It is often convenient to refer to the strategy (pure or mixed) of player i separately from that of the remaining players. To accommodate this, we use s_{-i} to denote the joint strategy of all players other than i.

2.2 Nash equilibrium

In this paper, we limit attention to one-shot normal-form games, in which players make decisions about their strategies simultaneously and accrue payoffs, upon which the game ends. This single-shot nature may seem to preclude multi-stage games, but in fact this representation is quite general, since strategies can be arbitrary functions of history of past play. Thus, any multi-stage game may be converted to a one-shot game by making the strategy sets to be sets of functions of all possible histories of play. However, if we do not explicitly represent the dynamic elements of a game, we do lose fidelity in terms of equilibrium concepts (for example, subgame perfection can no longer be represented).

Game payoff data may be obtained from observations of other agents playing the game, or from simulations of hypothetical runs of the game. In any of these cases, learning is relevant despite the fact that the game is to be played only once.

Faced with a one-shot game, an agent would ideally play its best strategy given those played by the other agents. A configuration where all agents play strategies that are best responses to the others constitutes a *Nash equilibrium*.

Definition 1. A strategy profile $s = (s_1, \ldots, s_m)$ constitutes a (pure-strategy) *Nash equilibrium* of game $[I, \{S_i\}, \{u_i(s)\}]$ if for every $i \in I$, $s_i' \in S_i$,

$$u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i}).$$

A similar definition applies when mixed strategies are allowed.

Definition 2. A strategy profile $\sigma = (\sigma_1, \ldots, \sigma_m)$ constitutes a *mixed-strategy Nash equilibrium* of game $[I, \{\Delta(S_i)\}, \{u_i(s)\}]$ if for every $i \in I$, $\sigma_i' \in \Delta(S_i)$,

$$u_i(\sigma_i, \sigma_{-i}) \geq u_i(\sigma_i', \sigma_{-i}).$$

In this study we devote particular attention to games that exhibit symmetry with respect to payoffs.

Definition 3. A game $[I, \{\Delta(S_i)\}, \{u_i(s)\}]$ is *symmetric* if $\forall i, j \in I$,

- $S_i = S_j$, and
- $u_i(s_i, s_{-i}) = u_j(s_j, s_{-j})$ whenever $s_i = s_j$ and $s_{-i} = s_{-j}$.

Symmetric games have relatively compact descriptions and may present associated computational advantages. Given a symmetric game, we may focus on the subclass of symmetric equilibria, which are arguably most natural (Kreps, 1990), and avoid the need to coordinate on roles.[1] In fairly general settings, symmetric games do possess symmetric equilibria (Nash, 1951; Cheng et al., 2004).

[1] Contention may arise when there are disparities among payoffs in asymmetric equilibrium. Even for symmetric equilibria, coordination issues may still be present with respect to equilibrium selection.

polynomial approximation yields no pure-strategy Nash equilibrium, we randomly select a symmetric pure strategy profile from the joint strategy set.

Another difficulty arises when a polynomial of a degree higher than three has more than one Nash equilibrium. In such a case we select an equilibrium arbitrarily.

3.3 Local regression

In addition to polynomial models, we explore learning using two local regression methods: locally weighted average and locally weighted quadratic regression (Atkeson, Moore, & Schaal, 1997). Unlike model-based methods such as polynomial regression, local methods do not attempt to infer model coefficients from data. Instead, these methods weigh the training data points by distance from the query point and estimate the answer—in our case, the payoff at the strategy profile point—using some function of the weighted data set. We used a Gaussian weight function:

$$w = e^{-d^2},$$

where d is the distance of the training data point from the query point and w is the weight that is assigned to that training point.

In the case of locally weighted average, we simply take the weighted average of the payoffs of the training data points as our payoffs for a given strategy profile. Locally weighted quadratic regression, on the other hand, fits a quadratic regression to the weighted data set for each query point.

3.4 Support vector machine regression

The third category of learning methods we investigate is Support Vector Machines (SVMs). For details regarding this learning method, we refer an interested reader to Vapnik (1995). In our experiments, we used the *SVM light* package (Joachims, 1999), which is an open-source implementation of SVM classification and regression algorithms, and chose Gaussian radial basis function of the form $e^{-2||x-x'||^2}$ as the kernel.

3.5 Finding mixed-strategy equilibria

In the case of polynomial regression models, we were able to find either analytic or simple and robust numeric methods for computing pure Nash equilibria. With local regression and SVM learning we are not so fortunate, as we do not have access to a closed-form description of the function we have learned.

When a particular learned model is not amenable to a closed-form solution, we use it to approximate a Nash equilibrium of the underlying game as follows. First, we restrict the learned function to a finite strategy subset. Since this restriction produces a finite game, we can thereafter apply any generic finite game solver to find an approximate Nash equilibrium of the *learned* game. For this task, we employed replicator dynamics (Fudenberg & Levine, 1998), which searches for a symmetric mixed equilibrium using an iterative evolutionary algorithm. We treated the result after a fixed number of iterations as an approximate Nash equilibrium of the learned game.[4]

[4] Replicator dynamics does not necessarily converge, but when it does reach a locally asymptotically stable fixed point the result is a Nash equilibrium (Friedman, 1991). For cases where replicator dynamics fails to converge, we still treat the final result as an approximate Nash equilibrium of the game.

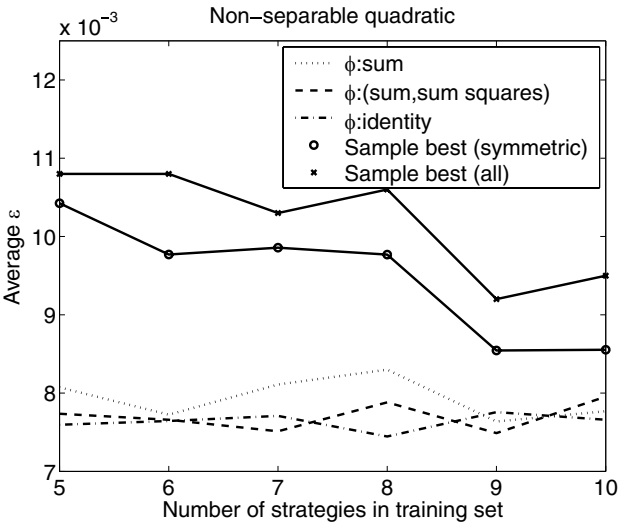
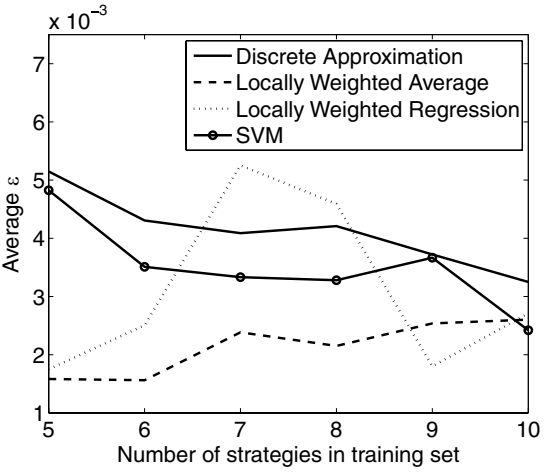


Fig. 6 Performance comparison of discrete and non-separable quadratic function approximation models with several forms of strategy aggregation

Fig. 7 Performance comparison of discrete approximation (with replicator dynamics used to find an equilibrium) and local and SVM regression methods with strategy aggregation of the form $\phi(s_{-i}) = (\phi_{sum}, \phi_{ss})$



our experiments on four of the six data set sizes we considered, and SVM consistently beat discrete approximation for all six data set sizes.[9]

It is somewhat surprising to see how irregular our results appear for the local regression methods. We cannot explain this irregularity, although of course there is no reason for us to expect otherwise: even though increasing the size of the training data set may improve

[9] Note that we do not compare these results to those for the polynomial regression methods. Given noise in the data set, mixed-strategy profiles with larger supports may exhibit lower ϵ simply due to the smoothing effect of the mixtures. Thus, when any profile with the lowest ϵ is desired, mixed-strategy profiles would likely be preferred. However, when pure strategy profiles are preferred, polynomial methods are more desirable as they produce pure strategy approximate equilibria.

the quality of fit, improvement in quality of equilibrium approximation does not necessarily follow.

In order to investigate the effect of noise on the function approximation methods as we had done in the FPSB setting, we ran another set of experiments in which we used the original data set of 300,000 samples per profiles as the training data set, and the data set of 2.5 million samples per profile as the evaluation data set. The remainder of the setup was as above.

The results in Figs. 8 and 9 show the performance of separable and non-separable quadratic models with different forms of strategy aggregation, comparing them to simply selecting the

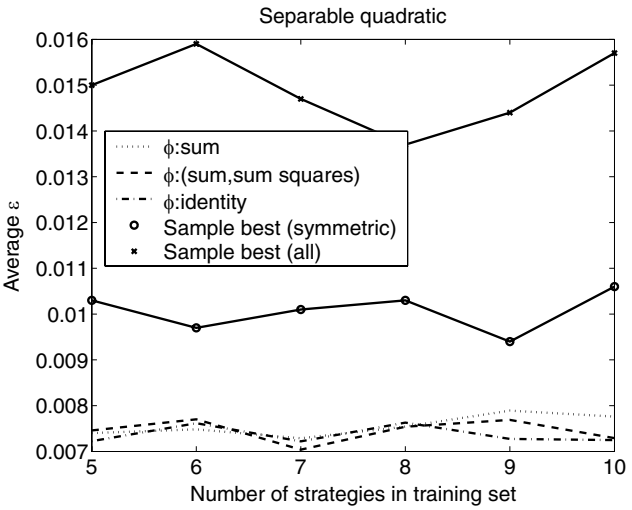


Fig. 8 Performance comparison of discrete and separable quadratic function approximation models with several forms of strategy aggregation when noise is present in training data

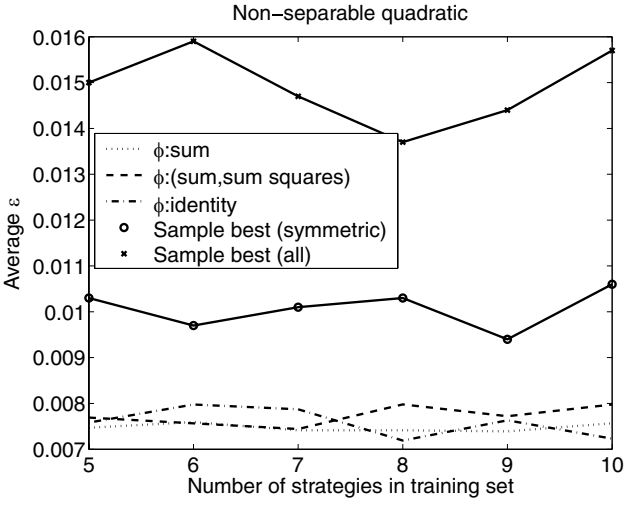


Fig. 9 Performance comparison of discrete and non-separable quadratic function approximation models with several forms of strategy aggregation when noise is present in training data

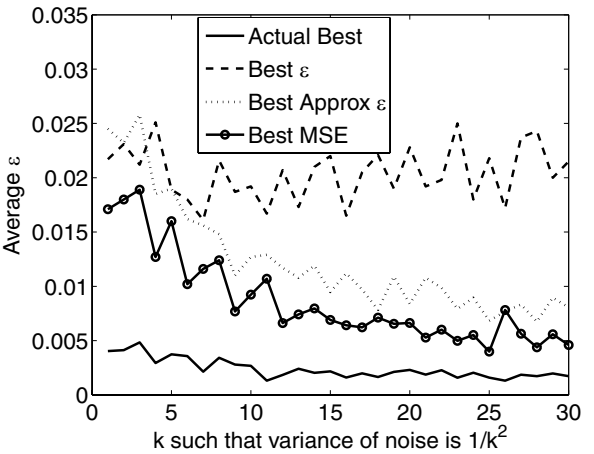


Fig. 13 Comparison of model selection criteria in FPSB as variance of the noise in the data decreases

of the learned game, as we discussed in some detail in Section 5. We refer to this method as "Best Approximate ϵ".

Figure 13 compares the three methods for model selection, varying the parameter, k, of the noise distribution, $N(0, 1/k^2)$, and keeping the size of the training set fixed at 25 profiles, representing all profiles for a random draw of 5 strategies. The model choices are restricted to separable and non-separable quadratics, as well as second- and third-degree separable polynomials. As a baseline for comparisons, we chose the model with the lowest actual ϵ in addition to applying the above selection criteria. This is referred to as "Actual Best" in the figure.

As another comparison between the same model selection methods, we fixed the distribution of noise to be $N(0, 1)$ and varied the number of strategies in the training data set between 5 and 10. The results of this comparison are shown in Fig. 14.

Both plots show that mean-squared error selection criterion consistently outperformed the others. Additionally, we can see from Fig. 13 that "Best Approximate ϵ" method tended to outperform the other ϵ-based method in the experiments. This provides some justification

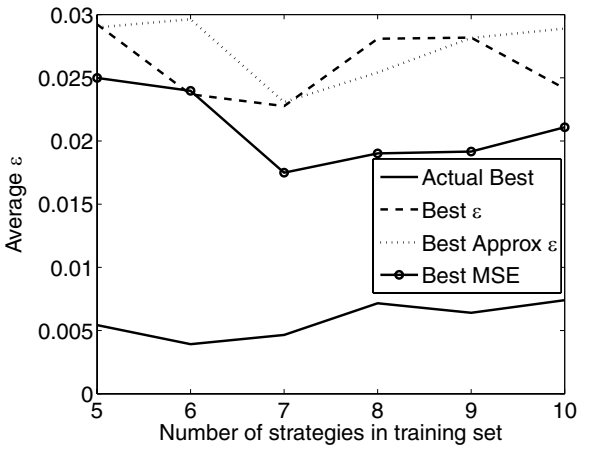


Fig. 14 Comparison of model selection criteria for different training data set sizes, fixing variance of noise at 1