Value Function Approximation using Hierarchical Aggregation for Multiattribute Resource Management

Abraham P. George    Warren B. Powell
Sanjeev R. Kulkarni
Department of Operations Research and Financial Engineering
Princeton University, Princeton, NJ 08544

September 2, 2004
Abstract

We consider the problem of estimating the value of a multiattribute resource, where the attributes are categorical in nature and the number of potential attribute vectors is very large. The problem arises in approximate dynamic programming when we need to estimate the value of a resource in the future using Monte Carlo estimates. We can represent the resources at different levels of aggregation, producing the classic tradeoff between statistical and structural error. We propose a method which estimates the value of a resource as a weighted combination of values at different levels of aggregation, and find the optimal weights which minimize the expected error, taking into account that the values are not independent. A simpler formula which assumes independence but accounts for biases is shown to produce similar results.
Consider the problem of managing a complex resource such as a human or equipment such as an aircraft or locomotive. We can represent the attributes of the resource using a multi-dimensional vector \( a \in \mathcal{A} \). Decisions which act on the resource produce a resource with a modified set of attributes, say \( a' \). Making the best decision now requires balancing the one-period reward plus the value of the resource in the future, which we denote by \( v_{a'} \). We can use statistical techniques to estimate \( v_{a'} \) using Monte Carlo methods, but we encounter the problem that the number of possible values of \( a \) can be extremely large. As a result, our estimate of \( v_{a'} \) may be quite poor (in fact, we may never have observed \( v_{a'} \) for a particular \( a' \)). Some examples of these problems include the following:

- Managing pilots for business jets - The attributes of a pilot include elements such as: home city, number of days away from home, the equipment that he is trained to fly, the number of hours he has been flying in a day, week or month, and the dates of the last three times he/she has landed at night. The last attribute is needed to determine whether the pilot is qualified to land at night. Sometimes, a pilot may be assigned to a flight at night to help maintain this status. Decisions about pilots can include assigning a pilot to a particular flight, or a decision to send a pilot for training on a new type of aircraft.

- Managing locomotives - The decision to assign a particular locomotive to a particular train has to consider attributes such as the type of locomotive, whether it is “leader qualified” (capable of being the lead locomotive out of the set of locomotives pulling a train), the number of days until it has to be maintained, its current location as well as its home maintenance shop, and the identity of the train that it pulled into a location. The last attribute identifies which other locomotives the locomotive is connected to (locomotives pull trains in a set called a “consist,” and the processing of coupling and uncoupling locomotives can be fairly time consuming). The decision to assign a locomotive to a train will change the attributes of the locomotive at the end of the move, and we need to know the value of the locomotive downstream in order to make this decision.

- A dry chemical bulk trailer is used to hold small pellets of a latex plastic of a particular
color used for manufacturing items. Although the trailer will generally be cleaned after each move, this cleaning process is not perfect, and as a result it is not possible to assign a trailer that has just pulled black pellets to move a load of light colored pellets until it has been cleaned several times. As a result, the attributes of the trailer have to include the precise set of colors that the trailer has pulled over the last five moves.

Our problem arises in any setting where we wish to estimate a parameter that depends on a variety of explanatory variables. We might, for example, wish to estimate the value of a suburban home that is being used as collateral in a home equity loan. The value might depend on the age, the location, the number of bedrooms, the size of the garage, and the recent prices of comparable homes. A model that depends on all these attributes simply will not have very many observations for each combination of these attributes (in fact, many combinations will not have any observations).

There are a variety of statistical strategies for estimating functions when \( a \) is made up of real-valued elements, but we are going to confine our attention to the general case where the elements of \( a \) are made up at least in part of categorical elements with no particular structure. Parametric methods, which fit a particular functional form for \( V(a) \) using a relatively small number of parameters, require making structural assumptions (such as separability or linearity) that are highly problem dependent (each application requires formulating, fitting and evaluating different parametric models). We do not even assume the existence of a metric \( \rho(a, a') \) that would provide a measure of the distance between two attribute vectors, which prevents the use of standard nonparametric methods.

These challenges arise in the field of approximate dynamic programming where Monte Carlo methods are used to estimate the value of being in a discrete state (see Bertsekas & Tsitsiklis (1996) and Sutton & Barto (1998) for introductions to this field). In this field, it has been common to use aggregation as a method for improving statistical reliability (at a cost of structural error). Strategies range from picking a fixed level of aggregation (Whitt (1978), Bean et al. (1987)), or using adaptive techniques that change the level of aggregation as the sampling process progresses (Bertsekas & Castanon (1989), Mendelssohn (1982), Bertsekas & Tsitsiklis (1996)), but which still use a fixed level of aggregation at any
given time.

In this paper, we focus purely on the problem of statistically estimating a function of an arbitrary, categorical attribute vector, with a predetermined set of aggregation functions. Although our problem arose in the context of estimating value functions for complex resource allocation problems, our techniques apply to any problem of estimating a quantity as a function of a set of attributes (or features) where aggregation offers an opportunity to reduce statistical error but at a cost of structural error. The important issues that arise in the context of dynamic programming are that new observations arrive sequentially, requiring updates of estimates of the function after each observation (in contrast with static estimation problems which seek to fit a function based on a given sample of observations). The rate of convergence is particularly important. While there are problems where it is possible to run thousands of iterations, we have encountered applications where it is only possible to run dozens of iterations.

We propose a hierarchical aggregation strategy which uses a weighted average of the estimate of the value of a resource at all levels of aggregation (for which there are observations) at the same time, using weights that change depending on statistics that are gathered as the algorithm progresses. The paper derives the optimal weights (under the assumption of known parameters), accounting for the lack of independence between statistics at different levels of aggregation. We also provide an approximation which assumes independence, but accounts for the bias that arises when aggregating a function. We experimentally compare a variety of weighting schemes under two settings: the idealized setting where the function parameters are known, and the more realistic setting of unknown parameters. Our tests are all done using a simple family of scalar functions which allows us to control the tradeoff between statistical and structural error. We found that the optimal weighting scheme (which accounts for correlations at different levels of aggregation) produces results that are very similar to an optimal weighting scheme that assumes independence and accounts for bias. This formula is easy to implement for large-scale, on-line learning applications that arise in approximate dynamic programming.

The paper is organized as follows. Section [1] provides the basic model and algorithmic
strategy to be used in the paper. Then, section 3 describes the hierarchical aggregation strategy in more detail. Section 4 derives the optimal weights from independent estimates when the estimates are biased, and then section 5 derives the optimal weights when the estimates exhibit the correlation that arises when using hierarchical aggregation. In section 6, we consider the special case where there are only two levels of aggregation, which allows the optimal weights to be found analytically. These expressions allow us to better compare the classical weighting formulas that assumes independence of estimates with the one that accounts for correlations. Section 7 summarizes a series of experiments analyzing different weighting schemes using a family of simple, scalar functions which allow us to exactly calculate our estimation errors. In section 8, we incorporate the techniques that we develop in an approximate dynamic programming application, namely, the nomadic trucker problem, in order to estimate the marginal values of individual states. In section 9, we provide our conclusions.

1 Setup

We assume that we are managing a single resource with multi-dimensional attribute $a$ consisting of arbitrary, categorical elements. The value of the resource at any stage is given by an unknown function $\nu_a$. We assume that at each iteration, we first sample the attribute vector $a$, and then sample the value of the resource, producing a sequence of observations $(\hat{a}^n, \hat{v}^n)$. We then wish to use this information to produce a statistically robust estimate of $\nu_a$.

We begin by defining the following:

$S = \text{A sample of observations } (\hat{a}^n, \hat{v}^n)_{n \in \mathcal{N}}, \text{ where } |\mathcal{N}| \text{ is the number of observations we have made so far.}$

$N_a = \text{The number of observations of attribute vector } a \text{ given our sample } S.$

$\hat{a}^n = \text{The attribute vector at observation } n.$

$\hat{v}^n = \text{The observation of the value of the resource with attribute } \hat{a}^n.$
\[1_{\{\hat{a}^n = a\}} = 1, \text{ if the } n\text{th observation is of attribute vector } a.\]

An estimate of \( \nu_a \) can be obtained as an average across all the observations of the response corresponding to \( a \). Our estimate of the value can be expressed as:

\[
\bar{v}_a = \frac{1}{N_a} \sum_{n \in \mathcal{N}} \hat{v}_n \hat{a}^n \{\hat{a}^n = a\}
\]

Throughout our presentation, we use the hat notation (as in \( \hat{v} \) and \( \hat{a} \)) to represent exogenous information, and bars (as in \( \bar{v} \)) to represent statistics derived from exogenous information. This notation will help distinguish what is random.

Since the attribute vector in these cases is multi-dimensional, there can be a huge number of possible states that it can attain. It might not always be possible to make enough observations so that the values of all the states may be estimated. In such a situation, we need to consider methods which cut down on the size of the attribute space. One way to do this is to ignore certain dimensions of the attribute vector, which will result in a much smaller attribute space. Another would be to model the problem such that it reflects some functional structure, like separability in the various attributes. In both cases, we are performing aggregation of the attribute space onto a smaller aggregate space, which can make the problem more tractable.

To illustrate, consider a case where the attribute vector has more than one dimension, with \( A_i \) denoting the number of possible values that attribute \( a_i \) can assume. The number of values that need to be estimated is \( \prod_i A_i \). Needless to say, as the attribute vector grows, such a model could require estimating millions of parameters which is generally infeasible. One strategy is to resort to aggregation (such as dropping one or more dimensions of \( a \)) which can quickly reduce the number of parameters but introduces structural error.

An alternative is to assume a structural property such as separability, which reduces the number of parameters to be estimated to \( \sum_i A_i \). This has fewer parameters, but requires that we introduce separability as an approximation. Since this will not be true as a general property, this strategy has the unfortunate byproduct of producing an approximation that does not converge in the limit to true values.
Regardless of the model we use, the dimensionality of $a$ will in general cause problems, and reducing dimensions of $a$ (which can be viewed as a form of aggregation) will reduce the number of parameters while introducing structural error. A traditional strategy is to choose the right level of aggregation by trading off statistical and structural errors to find a model with the least overall error. In our setting, the right tradeoff will change as we collect more observations. Furthermore, we generally do not control the sampling process of the attributes, and we will encounter instances where some regions of the attribute space $A$ will be sampled more than others.

Instead of focusing on choosing a single ideal level of aggregation, we could use different levels of aggregation at the same time. Let $\bar{\nu}_a^{(0)}$ be an estimate of $\nu_a$ at the most disaggregate level, and let $\bar{\nu}_a^{(1)}$ be an estimate at some level of aggregation. We could then estimate $\nu_a$ using:

$$\bar{\nu}_a = w_a^{(0)} \bar{\nu}_a^{(0)} + w_a^{(1)} \bar{\nu}_a^{(1)}$$

where $(w_a^{(0)}, w_a^{(1)})$ are a set of weights that sum to one. The weights will in general depend on both the attribute vector $a$ as well as the sample $S$, and will change as new observations are added. This strategy can be viewed as part of a broader statistical strategy for combining different estimates.

In the literature, there exist several techniques for combining estimates to improve accuracy. Bagging (Breiman (1996)) generates multiple versions of a predictor and combines these to get a better predictor. The aggregator averages over the versions when predicting a numerical outcome and does a plurality vote while predicting a class. Bagging can give substantial gains in accuracy, especially when the individual prediction methods are unstable and perturbations in the learning set can give significant changes in the predictor constructed. Another method called boosting (Freund & Schapire (1999)) sequentially applies a weak classification algorithm to repeatedly modified versions of the data, which produces a sequence of weak classifiers. The predictions from all of them are combined through a weighted majority to get a final prediction. The performance of this technique is, however, dependent on the data and the weak learning algorithm. LeBlanc & Tibshirani (1996) outlines a general
framework for how to combine a collection of general regression/classification fit vectors in order to obtain a better predictive model. The weights on the estimates from the individual predictors are computed by least squares minimization, stacked regression, generalized cross-validation and bootstrapping. Adaptive regression by mixing (Yang (1999)) assigns weights on candidate models that are combined after proper assessment of performance of the estimators, with the aim of reducing instability. The weights for combining the models are obtained as functions of the distributions of the error estimates and the variance of the random errors.

In all these methods, combining estimates has been proven to significantly improve the accuracy of predictions from individual models. In a hierarchical aggregation setting, we consider using the estimates from several levels of aggregation all at once to improve our prediction. In order to do this, we need to make an appropriate choice of weights, which is discussed in the later sections.

2 An Approximate Dynamic Programming Application: The Nomadic Trucker Problem

We consider a single resource version of the dynamic fleet management problem called the nomadic trucker, where there is a single trucker moving from city to city, either with a load or empty. When a load is moved, the trucker receives a reward. When moving empty, he incurs some cost. He also has the option of doing nothing, which would result in neither a cost nor a payment. This is illustrated in figure 1. The objective of the trucker is to maximize the profit over an indefinite time period.

The state of the resource, which is the trucker in this case, is defined by an attribute vector. We consider three attributes, namely, location, fleet type and driver domicile. The location attribute can be considered at three degrees of resolution - trade area, region or area. There are three different fleet types available with different trip length preferences - short, medium, and long distances. When the fleet type attribute is used, there is a penalty for trips outside the preferred range. The domicile attribute refers to the home location of
Figure 1: The Nomadic Trucker Problem - The trucker has to choose from a set of decisions to maximize the sum of current rewards and future value.

the trucker. Every time the trucker ends up at this location he receives a bonus. The state space, $A$, for this problem would consist of all possible combinations of the attributes of the trucker.

The trucker would be faced with a set of possible decisions ($D_a$) which is dependent on his current attribute vector, $a$. The decisions are where to move to and whether to move with a load or empty. From any given location, loads are available only to some of the locations, while the trucker can always choose to move empty anywhere. When acted upon by a decision, the attribute vector of the trucker is modified. The decision is evaluated based on the contribution $c$ (either payment of cost) that results from it and the value associated with being at the destination. The optimal value is obtained as the maximum of values resulting from all possible decisions $d$.

$$\hat{v}_a = \max_{d \in D_a} \{c(a, d) + \gamma v_{a'}\}$$  \hspace{1cm} (1)

This can be done iteratively where we update the value associated with the current attribute using a stepsize, $\alpha$, that could depend on the number of visits that the attribute vector has
Table 1: Examples of aggregations on the attribute space for the nomadic trucker problem. ‘*’ denotes that the attribute is used, ‘-’ indicates that the particular attribute is ignored

<table>
<thead>
<tr>
<th>Aggregation level</th>
<th>Location</th>
<th>Fleet type</th>
<th>Domicile</th>
<th>Size of state space</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Trade area</td>
<td>*</td>
<td>Region</td>
<td>$400 \times 3 \times 100 = 120,000$</td>
</tr>
<tr>
<td>1</td>
<td>Region</td>
<td>*</td>
<td>Area</td>
<td>$100 \times 3 \times 10 = 3,000$</td>
</tr>
<tr>
<td>2</td>
<td>Region</td>
<td>*</td>
<td>-</td>
<td>$100 \times 3 \times 1 = 300$</td>
</tr>
<tr>
<td>3</td>
<td>Area</td>
<td>-</td>
<td>-</td>
<td>$10 \times 1 \times 1 = 10$</td>
</tr>
</tbody>
</table>

received.

$$v^n_a = (1 - \alpha)v^{n-1}_a + \alpha\hat{v}^n_a$$ (2)

In order to have an estimate of the value of a state, it needs to be actually visited. However, if the state space becomes large, it might be the case that many of the attribute states are never visited, while others receive a few visits but not enough for a statistically reliable estimate. We are faced with the challenge of obtaining reliable estimates of their values.

Aggregation can be used as strategy for solving the problem of large state spaces, where we estimate the value of a state from those of similar states. We can use aggregation to create a hierarchy of state spaces, $\{A^{(g)}, g = 1, 2, \ldots, |G|\}$, with possibly fewer elements than the original state space. For example, there could be 400 possible locations at the trade area level which could be aggregated to 100 at a regional level. We could further aggregate these to 10 locations at the area level. Alternatively, we could perform aggregation by ignoring the fleet type of the truck or the domicile attribute or both. We may also aggregate on more than one attribute at a time. Table 1 lists some of the possible aggregations for this problem. Aggregation enables us to form estimates of the values of states with fewer observations. The idea is that when there is not much information available, observations made elsewhere in the state space can help us obtain a rough idea of the characteristics of the state under consideration although those other states may be different from it at a detailed level.

Our nomadic trucker application offers us enough richness to illustrate approximation methods that can be applied to larger and more complex applications. At the same time, it
Step 0. Initialize an approximation for the value function $\bar{v}_a^{(g),0}$ for all attribute vector states $a \in \mathcal{A}$ at all levels of aggregation $g \in \mathcal{G}$ and set $n = 1$.

Step 1. Iteration $n$:

Step 2. Randomly sample attribute vector $a$.

Step 3. Obtain the set of possible decisions, $\mathcal{D}_a$.

Step 4. Compute the value function estimates of the future states using some aggregation strategy.

$$\tilde{v}_{a'd}^{n-1} = f \left( \bar{v}_a^{(g),n-1}, g \in \mathcal{G} \right) \quad \forall d \in \mathcal{D}_a$$

Step 5. Solve for the optimal decision, given the current value function estimates.

$$d^n(a) = \arg \max_{d \in \mathcal{D}_a} \left\{ c(a,d) + \gamma \tilde{v}_{a'd}^{n-1} \right\}$$

$$\hat{v}_a^n = \max_{d \in \mathcal{D}_a} \left\{ c(a,d) + \gamma \tilde{v}_{a'd}^{n-1} \right\}$$

Step 6. Use the results to update the value function estimates and the associated statistics (number of observations and sample variance) at all levels of aggregation of the current attribute vector

$$\bar{v}_a^{(g),n} = (1 - \alpha)\bar{v}_a^{(g),n-1} + \alpha \hat{v}_a^n \quad \forall g \in \mathcal{G}$$

Step 7. Let $n = n + 1$. If $n < N$ go to step 1, else for each state $a$, $d^n(a)$ defines the near-optimal policy.

---

Figure 2: Forward dynamic programming algorithm using aggregation

is small enough that we can solve the problem exactly using classical methods such as value iteration.

We outline the steps for determining the approximate values of the various attribute states for the nomadic trucker problem in figure 2.

3 Hierarchical Aggregation

In general, aggregation of attribute vectors is performed using a collection of aggregation functions,

$$\mathcal{G}^g : \mathcal{A} \rightarrow \mathcal{A}^{(g)}$$

$\mathcal{A}^{(g)}$ represents the $g^{th}$ level of aggregation of the attribute space $\mathcal{A}$. We define the following:

$${a}'_a^{(g)} = \mathcal{G}^g(a), \text{ the } g^{th} \text{ level aggregation of the attribute vector } a.$$  

$\mathcal{G} = \text{ The set of indices corresponding to the levels of aggregation}$
It is usually the case that the $g$th level of aggregation acts on the $(g - 1)$st level as illustrated in Table 1. For example, the first-level aggregation function, $G^{(1)}$, involves aggregating the location to the regional level and the domicile to the area level. The second-level aggregation function, $G^{(2)}$, would be defined as aggregating out the driver domicile attribute.

In order to estimate a function whose value is determined by the vector of values assumed by a set of attributes, we make several observations of the attribute vector-value pairs $(a, v)$, as defined in Section 1. The observations are indexed by $n \in N$. We let $\varepsilon^n$ denote the error in the $n$th observation with respect to the true value associated with $\hat{a}^n$. We assume that $\varepsilon^n$ has a mean value of zero. We can express,

$$\hat{v}^n = v_{\hat{a}^n} + \varepsilon^n$$

We define the following probability spaces,

\begin{align*}
\Omega^a &= \text{ The set of outcomes of observations of attribute vectors.} \\
\Omega^\varepsilon &= \text{ The set of outcomes of observations of the errors in the values.} \\
\Omega &= \text{ The overall set of outcomes.} \\
&= \Omega^a \times \Omega^\varepsilon. \\
\omega &= (\omega^a, \omega^\varepsilon) \\
&= \text{ An element of the outcome space.}
\end{align*}

An important component of our prediction error will be aggregation bias. Consider our most recent observed attribute vector $\hat{a}^n$ and some other attribute $a$, where $\hat{a}^n$ and $a$ may aggregate up to the same aggregated attribute at some level (for the moment, these are simply two attribute vectors). In our derivations below, it is useful to define a bias term

$$\mu_n^a = v_{\hat{a}^n} - v_a$$ (3)
We can use this notation to rewrite \( \hat{v}^n \) as follows:

\[
\hat{v}^n = \nu_a + (\nu_{a^n} - \nu_a) + \varepsilon^n
\]

\[
= \nu_a + \mu_a^n + \varepsilon^n
\]

We can use these relationships to obtain an estimate of the value associated with the attribute vector \( a \) at any level of aggregation. We define,

\[
\hat{A} = \text{The sequence of attribute vectors observed.}
\]

\[
= \{ \hat{a}^n \mid n \in N \}
\]

\( N_a^{(g)} \) = The set of indices that correspond to observations of the attribute vector \( a \) at the \( g \)th level of aggregation.

\[
= \{ n \mid G^g(\hat{a}^n) = G^g(a) \}
\]

\( N_a^{(g)} = |N_a^{(g)}| \)

\( v_a^{(g)} = \) The estimate of the value associated with the attribute vector \( a \) at the \( g \)th level of aggregation, given the sample, \( N \).

\( \delta_a^{(g)} = \) The error in the estimate, \( \bar{v}_a^{(g)} \), from the true value associated with attribute vector \( a \).

\[
= \bar{v}_a^{(g)} - \nu_a
\]

We compute the estimate, \( \bar{v}_a^{(g)} \), as follows,

\[
\bar{v}_a^{(g)} = \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} \tilde{v}^n
\]

\[
= \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} (\nu_a + \mu_a^n + \varepsilon^n)
\]

\[
= \nu_a + \left( \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} \mu_a^n \right) + \left( \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} \varepsilon^n \right)
\]
We let,

\[
\bar{\mu}_a^{(g)} = \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} \mu_a^n
\]

\[
\bar{\varepsilon}_a^{(g)} = \frac{1}{N_a^{(g)}} \sum_{n \in N_a^{(g)}} \varepsilon_n
\]

\(\bar{\mu}_a^{(g)}\) gives an estimate of the bias between the values of \(a\) at the \(g\)th level of aggregation and at the disaggregate level. \(\bar{\mu}_a^{(g)}\) is a random variable that is a function of the set of points sampled. \(\bar{\varepsilon}_a^{(g)}\) is an estimate of the random error that has zero expected value. The variability in \(\bar{\varepsilon}_a^{(g)}\) occurs because of the randomness of the set of points sampled as well as the statistical noise (which, in general could be dependent on the attribute vector sampled) in the observation of the value.

Therefore, we have,

\[
\bar{\delta}_a^{(g)} = \bar{\mu}_a^{(g)} + \bar{\varepsilon}_a^{(g)}
\]

(5)

\[
\bar{\nu}_a^{(g)} = \nu_a + \bar{\mu}_a^{(g)} + \bar{\varepsilon}_a^{(g)}
\]

(6)

Now, we have an estimate of the value, \(\nu_a\), for each level of aggregation.

The estimate at a higher level of aggregation, \(g'\), where \(g' \geq g\), would be given by,

\[
\bar{\nu}_a^{(g')} = \frac{1}{N_a^{(g')}} \sum_{n \in N_a^{(g')}} \hat{v}_a^n
\]

\[
= \nu_a + \frac{1}{N_a^{(g')}} \sum_{n \in N_a^{(g')}} \mu_a^n + \frac{1}{N_a^{(g')}} \sum_{n \in N_a^{(g')}} \varepsilon_n
\]

\[
= \nu_a + \bar{\mu}_a^{(g')} + \frac{1}{N_a^{(g')}} \left( \sum_{n \in N_a^{(g')}} \varepsilon_n + \sum_{n \in N_a^{(g')}} \varepsilon_n \right)
\]

Thus, the stochastic error term at the higher level of aggregation can be expressed as a combination of the one at the lower level of aggregation and some terms that are independent
of it:

\[
\tilde{\varepsilon}_a^{(g')} = \frac{1}{N_a^{(g')}} \left( \sum_{n \in \Lambda_a^{(g')}} \varepsilon^n + \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n \right) \\
= \frac{N_a^{(g')}}{N_a^{(g')}} \varepsilon_a^{(g)} + \frac{1}{N_a^{(g')}} \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n 
\]

(7)

The question arises as to which is the best level of aggregation. The right level of aggregation will generally be a function of the number of observations made. It is common in practice to choose a single level of aggregation that produces the lower overall error. However, it can be useful to combine estimates from different sources. In the next section, we address the issue of combining independent estimates in the presence of a bias term, after which we derive an optimal weighting strategy for hierarchically obtained estimates which are not independent.

4 Combining independent estimates

We begin with the problem with combining independent estimates of a quantity \( \nu \). Let \((\tilde{v}^{(g)})_{g \in \mathcal{G}}\) be a set of estimates of \( \nu \) which are statistically independent. We can form a weighted combination of these estimates using:

\[
\tilde{v} = \sum_{g \in \mathcal{G}} w^{(g)} \cdot \tilde{v}^{(g)}
\]

where \(w^{(g)}\), \(g \in \mathcal{G}\) are appropriately chosen weights. We would choose the weights to minimize \(\mathbb{E}\left[\frac{1}{2}(\tilde{v} - \nu)^2\right]\):

\[
\min_{w^{(g)}, g \in \mathcal{G}} \mathbb{E}\left[\frac{1}{2} \left( \sum_{g \in \mathcal{G}} w^{(g)} \cdot \tilde{v}^{(g)} - \nu \right)^2 \right]
\]

subject to the constraints:

\[
\sum_{g \in \mathcal{G}} w^{(g)} = 1 \\
w^{(g)} \geq 0, \quad g \in \mathcal{G}
\]
Bayesian theory (see Bernardo & Smith (1994)) gives the following simple formula for the optimal weights:

\[
w(g) = \frac{1}{\sigma(g)^2} \left( \sum_{g' \in G} \frac{1}{\sigma(g')^2} \right)^{-1} \quad \forall \ g \in G
\]  

(8)

In addition to independence, the previous result assumes that the estimates \(\bar{v}(g)\) are unbiased. We are going to be making estimates at different levels of aggregation which will introduce structural error, which biases our estimates. When biases may be present, equation (8) is no longer optimal. We can compensate for biases in a simple way. First define:

\[
\mu(g) = \text{Expected bias in the estimate, } \bar{v}(g)
\]

\[
= \mathbb{E} \left[ \bar{v}(g) - \nu \right]
\]

Proposition 1 The optimal weights are now given by,

\[
w(g) = \frac{1}{\sigma(g)^2 + \mu(g)^2} \left( \sum_{g' \in G} \frac{1}{\sigma(g')^2 + \mu(g')^2} \right)^{-1} \quad \forall \ g \in G
\]

Proof: The derivation follows the same reasoning as the case with unbiased estimates. However, in order to compute the mean squared errors of the estimates from the true value, we have to consider their variances as well as their biases from the true value. The variance of the estimate \(\bar{v}(g)\) is the expected squared deviation from its mean and can be written as follows:

\[
\sigma(g)^2 = \mathbb{E} \left[ (\bar{v}(g) - \nu - \mu(g))^2 \right]
\]

\[
= \mathbb{E} \left[ (\bar{v}(g) - \nu)^2 - 2\mu(g) (\bar{v}(g) - \nu) + \mu(g)^2 \right]
\]

\[
= \mathbb{E} \left[ (\bar{v}(g) - \nu)^2 \right] - 2\mu(g) \mathbb{E} [\bar{v}(g) - \nu] + \mu(g)^2
\]

\[
= \mathbb{E} \left[ (\bar{v}(g) - \nu)^2 \right] - 2\mu(g) \mu(g) + \mu(g)^2
\]

\[
= \mathbb{E} \left[ (\bar{v}(g) - \nu)^2 \right] - \mu(g)^2
\]
Thus, the mean squared error can be written as:

$$E \left[ (\bar{v}^{(g)} - \nu)^2 \right] = \sigma^{(g)^2} + \mu^{(g)^2}$$  \hspace{1cm} (9)$$

The revised weighting formula is then given by:

$$w^{(g)} = \frac{1}{\sigma^{(g)^2} + \mu^{(g)^2}} \left( \sum_{g' \in \mathcal{G}} \frac{1}{\sigma^{(g')^2} + \mu^{(g')^2}} \right)^{-1} \quad \forall \ g' \in \mathcal{G}$$  \hspace{1cm} (10)$$

□

5 Combining correlated estimates

We now consider estimates which are correlated by way of a tree structure as in the case of the aggregation network. We have observations at the most disaggregate level and these are used to get estimates at various levels of aggregation. We wish to find the weighting scheme that will optimally combine the estimates at the different levels of aggregation, that is, the weights which give a combined estimate with the least squared deviation from the true value associated with attribute vector $a$. The optimization formulation is as follows:

$$\min_{w_a^{(g)}, g \in \mathcal{G}} E \left[ \frac{1}{2} \left( \sum_{g \in \mathcal{G}} w_a^{(g)} \cdot \bar{v}_a^{(g)} - \nu_a \right)^2 \right]$$  \hspace{1cm} (11)$$

subject to:

$$\sum_{g \in \mathcal{G}} w_a^{(g)} = 1$$  \hspace{1cm} (12)$$

$$w_a^{(g)} \geq 0, \quad g \in \mathcal{G}$$  \hspace{1cm} (13)$$

We state the following theorem:

**Theorem 1** For a given attribute vector, $a$, the optimal weights, $w_a^{(g)}$, $g \in \mathcal{G}$, where the individual estimates are correlated by way of a tree structure, are given by solving the following
system of linear equations in \((w, \lambda)\):

\[
\sum_{g \in G} w_a^{(g)} \mathbb{E} \left[ \bar{\delta}_a^{(g)} \bar{\delta}_a^{(g')} \right] - \lambda = 0 \quad \forall \ g' \in G
\] (14)

\[
\sum_{g \in G} w_a^{(g)} = 1
\] (15)

\[
w_a^{(g)} \geq 0 \quad \forall \ g \in G
\] (16)

Proof: The Lagrangian for the problem formulated in (11)-(13) is,

\[
L(w, \lambda) = \mathbb{E} \left[ \frac{1}{2} \left( \sum_{g \in G} w_a^{(g)} \cdot \bar{v}_a^{(g)} - \nu_a \right)^2 \right] + \lambda \left( 1 - \sum_{g \in G} w_a^{(g)} \right)
\]

\[
= \mathbb{E} \left[ \frac{1}{2} \left( \sum_{g \in G} w_a^{(g)} \left( \bar{v}_a^{(g)} - \nu_a \right) \right)^2 \right] + \lambda \left( 1 - \sum_{g \in G} w_a^{(g)} \right)
\]

The first order optimality conditions are easily shown to be,

\[
\mathbb{E} \left[ \sum_{g \in G} w_a^{(g)} \left( \bar{v}_a^{(g)} - \nu_a \right) \left( \bar{v}_a^{(g')} - \nu_a \right) \right] - \lambda = 0 \quad \forall \ g' \in G
\] (17)

\[
\sum_{g \in G} w_a^{(g)} - 1 = 0
\] (18)

To simplify equation (17), we note that,

\[
\mathbb{E} \left[ \sum_{g \in G} w_a^{(g)} \left( \bar{v}_a^{(g)} - \nu_a \right) \left( \bar{v}_a^{(g')} - \nu_a \right) \right] = \mathbb{E} \left[ \sum_{g \in G} w_a^{(g)} \bar{\delta}_a^{(g)} \bar{\delta}_a^{(g')} \right]
\]

\[
= \sum_{g \in G} w_a^{(g)} \mathbb{E} \left[ \bar{\delta}_a^{(g)} \bar{\delta}_a^{(g')} \right]
\] (19)

Combining equations (17) and (19) gives us equation (14) which completes the proof.

We now turn to the problem of computing \(\mathbb{E} \left[ \bar{\delta}_a^{(g)} \bar{\delta}_a^{(g')} \right]\). In the remainder of the analysis, we constrain ourselves to the case where we are given the observed sequence of attribute vectors - that is, all expectations and probabilities are computed with respect to the probability space, \(\Omega^\varepsilon\).
Proposition 2. The coefficients of the weights in equation (15) can be expressed as follows:

\[ \mathbb{E} \left[ \tilde{\delta}_a^g \delta(g') \right] = \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\mu}_a^{(g')} \right] + \frac{N_a^{(g)}}{N_a^{(g')}} \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \right] \quad \forall g \leq g' \text{ and } g, g' \in \mathcal{G} \quad (20) \]

Proof: Using equation (5), we obtain the relation,

\[ \mathbb{E} \left[ \tilde{\delta}_a^g \delta(g') \right] = \mathbb{E} \left[ (\bar{\mu}_a^{(g)} + \bar{\varepsilon}_a^{(g)}) (\bar{\mu}_a^{(g')} + \bar{\varepsilon}_a^{(g')}) \right] = \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\mu}_a^{(g') + \bar{\varepsilon}_a^{(g)}} + \bar{\mu}_a^{(g')} \bar{\varepsilon}_a^{(g)} + \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] = \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\mu}_a^{(g')} \right] + \mathbb{E} \left[ \bar{\mu}_a^{(g')} \bar{\varepsilon}_a^{(g)} \right] + \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] + \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] \quad (21) \]

We notice that, \( \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] = \mu_a^{(g')} \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \right] = 0 \). By a similar argument, \( \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] = 0 \). This enables us to rewrite equation (21) as,

\[ \mathbb{E} \left[ \tilde{\delta}_a^g \delta(g') \right] = \mathbb{E} \left[ \bar{\mu}_a^{(g)} \bar{\mu}_a^{(g')} \right] + \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] \quad (22) \]

We consider the second term on the right hand side of equation (22). Using equation (7), we can rewrite this term as follows:

\[ \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] = \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \cdot \frac{N_a^{(g)}}{N_a^{(g')}} \bar{\varepsilon}_a^{(g)} \right] + \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \cdot \frac{1}{N_a^{(g')}} \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n \right] = \frac{N_a^{(g)}}{N_a^{(g')}} \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g)} \right] + \frac{1}{N_a^{(g')}} \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \cdot \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n \right] \quad (23) \]

The term \( I \) can be further simplified as follows,

\[ \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \cdot \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n \right] = \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \right] \mathbb{E} \left[ \sum_{n \in \Lambda_a^{(g')} \setminus \Lambda_a^{(g)}} \varepsilon^n \right] = 0 \]

This implies that,

\[ \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \bar{\varepsilon}_a^{(g')} \right] = \frac{N_a^{(g)}}{N_a^{(g')}} \mathbb{E} \left[ \bar{\varepsilon}_a^{(g)} \right] \quad (24) \]
Combining (22) and (24) gives us our result.

We can further simplify the expression in equation (20) if we assume that the statistical noise in the measurement of \( \hat{v}^n \) is independent of the attribute vector \( \hat{a}^n \). We let \( \sigma^2_\varepsilon \) denote the variance of the statistical noise in the observations.

**Corollary 1** For the special case where the statistical noise in the measurement of the values is independent of the attribute vector sampled, equation (20) reduces to,

\[
\mathbb{E} \left[ \bar{\delta}_{\hat{a}}(g) \bar{\delta}_{\hat{a}}(g') \right] = \mathbb{E} \left[ \bar{\mu}_{\hat{a}}(g) \bar{\mu}_{\hat{a}}(g') \right] + \frac{\sigma^2_\varepsilon}{N_a(g')}, \tag{25}
\]

**Proof:** The second term on the right hand side of equation (24) can be further simplified using,

\[
\mathbb{E} \left[ \bar{\varepsilon}_{\hat{a}}(g)^2 \right] = \mathbb{E} \left[ \left( \frac{1}{N_a(g)} \sum_{n \in N_a(g)} \varepsilon^n \right)^2 \right], \quad \forall \ g' \in \mathcal{G}
\]

\[
= \frac{1}{(N_a(g))^2} \sum_{m,n \in N_a(g)} \mathbb{E} \left[ \varepsilon^m \varepsilon^n \right] \\
= \frac{1}{(N_a(g))^2} \sum_{m,n \in N_a(g)} \mathbb{E} \left[ (\varepsilon^n)^2 \right] \\
= \frac{1}{(N_a(g))^2} N_a(g) \sigma^2_\varepsilon \\
= \frac{\sigma^2_\varepsilon}{N_a(g)} \tag{26}
\]

Combining equations (20), (24) and (26) gives us the result in equation (25). 

For the case where \( g = 0 \), we can use the result, \( \mathbb{E} \left[ \bar{\mu}_{\hat{a}}(0) \bar{\mu}_{\hat{a}}(g') \right] = 0 \) (which follows from the property: \( \mu_{\hat{a}}^{(0)} = 0 \)), to further simplify (25) and obtain the following result:

\[
\mathbb{E} \left[ \bar{\delta}_{\hat{a}}^{(0)} \bar{\delta}_{\hat{a}}(g') \right] = \frac{\sigma^2_\varepsilon}{N_a(g')} \tag{27}
\]
6 Special case: Two levels of aggregation

In this section, we discuss the special case where we combine only two correlated estimates and the statistical noise is independent of the attribute vector sampled. We assume that we know the probability distributions of the sampling of the attribute vectors and their values. These enable us to solve the optimality equations to obtain a solution explicitly.

We perform the analysis as follows. Suppose we have a particular realization of the value function estimates at the different levels of aggregation. The mean squared error of the combined estimate from the true value associated with the attribute vector \(a\) can be depicted as follows,

\[
F(w_a, \omega) = \frac{1}{2} \left( \sum_{g \in G} w_a^{(g)} \cdot \tilde{v}_a^{(g)}(\omega) - \nu_a \right)^2
\]  

(28)

We analytically compute weights that minimize \(E[F(w_a)]\). We obtain the optimal weights by solving the following system of equations:

\[
E\left[\tilde{\delta}_a^{(0)} \tilde{\delta}_a^{(1)}\right] w_a^{(0)} + E\left[\tilde{\delta}_a^{(1)} \tilde{\delta}_a^{(1)}\right] w_a^{(1)} - \lambda = 0 
\]  

(29)

\[
E\left[\tilde{\delta}_a^{(0)} \tilde{\delta}_a^{(1)}\right] w_a^{(0)} + E\left[\tilde{\delta}_a^{(0)} \tilde{\delta}_a^{(0)}\right] w_a^{(0)} + E\left[\tilde{\delta}_a^{(1)} \tilde{\delta}_a^{(1)}\right] w_a^{(1)} - \lambda = 0
\]  

(30)

\[
w_a^{(0)} + w_a^{(1)} = 1
\]  

(31)

Since we are dealing with getting the estimates for a particular attribute vector, we drop the index \(a\) in the following analysis. We obtain the value of \(w^{(0)}\) as,

\[
w^{(0)} = \frac{E\left[\tilde{\delta}^{(1)}\tilde{\delta}^{(0)}\right] - E\left[\tilde{\delta}^{(0)}\tilde{\delta}^{(0)}\right]}{E\left[\tilde{\delta}^{(0)}\tilde{\delta}^{(0)}\right] + E\left[\tilde{\delta}^{(1)}\tilde{\delta}^{(1)}\right] - 2E\left[\tilde{\delta}^{(0)}\tilde{\delta}^{(1)}\right]}
\]  

(32)

We call this weighting scheme WOPT. If we assume that the estimates at the two levels of aggregation are independent, that is, \(E[\tilde{\delta}^{(0)}\tilde{\delta}^{(1)}] = 0\), then we can express \(w^{(0)}\) as,

\[
w^{(0)} = \frac{E\left[\tilde{\delta}^{(1)}\right]}{E\left[\tilde{\delta}^{(0)}\right] + E\left[\tilde{\delta}^{(1)}\right]}
\]  

(33)
We name this strategy of weighting the estimates WIND, which is optimal when the estimates are independent but may be a good approximation when the estimates are not independent (as would occur in our case).

By assumption, the estimate at the disaggregate level is unbiased, that is, $\mu^{(0)} = 0$. We define,

\[ m^2 = \text{The expected value of the square of the bias term at the aggregate level} \]
\[ = E\left[\tilde{\mu}^{(1)^2}\right] \quad (34) \]

For obtaining exact solutions, we use the following expressions,

\[ E\left[\delta^{(0)}\delta^{(1)}\right] = \sigma^2 \varepsilon \frac{N(1)}{m^2} \]
\[ E\left[\delta^{(0)^2}\right] = \sigma^2 \varepsilon \frac{N(0)}{m^2} \]
\[ E\left[\delta^{(1)^2}\right] = E\left[\tilde{\mu}^{(1)^2}\right] + E\left[\varepsilon^{(1)^2}\right] = m^2 + \sigma^2 \varepsilon \frac{N(1)}{m^2} \]

We denote by $w^{opt}$ and $w^{ind}$, the weights on the disaggregate estimate as computed using the WOPT and WIND schemes respectively.

Equations (32) and (33) reduce to,

\[ w^{opt} = \frac{1}{1 + \left(\frac{1}{N(0)} - \frac{1}{N(1)}\right) \frac{\sigma^2 \varepsilon}{m^2}} \quad \text{(WOPT)} \quad (35) \]
\[ w^{ind} = \frac{1 + \frac{1}{N(1)} \frac{\sigma^2 \varepsilon}{m^2}}{1 + \left(\frac{1}{N(0)} + \frac{1}{N(1)}\right) \frac{\sigma^2 \varepsilon}{m^2}} \quad \text{(WIND)} \quad (36) \]

We define the parameter $g_i, i = 0, 1$ as follows,

\[ g_i = \frac{1}{N(i)} \frac{\sigma^2 \varepsilon}{m^2} \quad i = 0, 1 \]

**Proposition 3** The upper bound for the difference in the estimates by WOPT and WIND
is given by,

$$\Delta v \leq \frac{(1 - g_0 + g_1)g_1}{(1 + g_0)^2 - g_1^2} (\bar{v}^{(1)} - \bar{v}^{(0)})$$  \hspace{1cm} (37)

**Proof:** We denote by $\tilde{v}^{opt}$ and $\tilde{v}^{ind}$ the estimates computed using the two weighting schemes.

$$\tilde{v}^{opt} = w^{opt}\bar{v}^{(0)} + (1 - w^{opt})\bar{v}^{(1)}$$ \hspace{1cm} (38)

$$\tilde{v}^{ind} = w^{ind}\bar{v}^{(0)} + (1 - w^{ind})\bar{v}^{(1)}$$ \hspace{1cm} (39)

The difference between the two estimates,

$$\Delta v = \tilde{v}^{opt} - \tilde{v}^{ind}$$

$$= (w^{opt}\bar{v}^{(0)} + (1 - w^{opt})\bar{v}^{(1)}) - (w^{ind}\bar{v}^{(0)} + (1 - w^{ind})\bar{v}^{(1)})$$

$$= (w^{opt} - w^{ind}) (\bar{v}^{(1)} - \bar{v}^{(0)})$$

$$= \Delta w (\bar{v}^{(1)} - \bar{v}^{(0)})$$ \hspace{1cm} (40)

We can evaluate the difference ($\Delta w$) between the weights given by the two schemes as follows:

$$\Delta w = w^{opt} - w^{ind}$$

$$= 1 + \frac{1}{N^{(0)}} - \frac{1}{N^{(1)}} \frac{\sigma^2}{m^2} - \frac{1 + \frac{1}{N^{(0)}} + \frac{1}{N^{(1)}}}{1 + g_0 + g_1} \frac{\sigma^2}{m^2}$$

$$= \frac{1 + g_0 - g_1}{1 + g_0 + g_1}$$

$$= \frac{(1 - g_0 + g_1)g_1}{(1 + g_0)^2 - g_1^2}$$ \hspace{1cm} (41)

Substituting this result in equation (40) gives us the proposed bound. \hfill \Box

7 **Experiments on Scalar Functions**

In this section, we evaluate the performance of the competing hierarchical weighting strategies. The central questions concern the quality of the solution as a function of the number
of observations. The rate of convergence is of particular interest in some applications of approximate dynamic programming where a single iteration can take up to an hour. In section 7.1 we describe the experimental design. In section 7.2 we compute the weights for combining the estimates at different levels of aggregation assuming that the parameters of the underlying function are known. These experiments are used to evaluate the accuracy of the optimal weighting strategy and compare it to the simpler formula which assumes that the estimates are independent. We also compare these to the strategy which uses optimal weights computed from sample values. In section 7.3 we conduct experiments where we require that all parameters be estimated from observations.

7.1 Experimental Design

In order to measure the accuracy of the various weighting strategies using aggregation, we apply them to the estimation of several classes of known scalar functions. The functions are chosen so that we can control the relative levels of statistical noise and structural variation. We use known functions so that we can exactly evaluate the error. We assume that our most disaggregate representation is fine-grained, which means that there is no structural error (structural errors at the most disaggregate level complicate the analysis without adding insight). To achieve this, we discretized the domain of the function into a set of discrete cells at the disaggregate level.

Our sampling process occurs in two steps: the sampling of the cell, and then the sampling of the observed value of the cell. The probability law for sampling the cells could be uniform or arbitrary, but the sampling of the values assumed a normal distribution for the errors. After each observation, for each cell at each level of aggregation, we compute estimates of the value at each level of aggregation as an average of the values sampled for the observations corresponding to that particular cell. Thus after each update, we have estimates of the values at the different levels of aggregation and their variances.

In order to compare the effectiveness of different weighting strategies, we developed a measure of the degree to which a weighting strategy reduced the variance of an estimate.
We define the following:

\[ \tilde{v}_a^s = \text{The value of the attribute vector } a \text{ as estimated by strategy } s. \]

\[ \epsilon^s = \text{The sum of squared errors as estimated by strategy } s. \]

\[ = \sum (\tilde{v}_a^s - \nu_a)^2 \]
\(G\) denotes the static aggregation strategy which treats the function as a constant over its domain.

\(\theta^s\) measures the degree of variability explained by a particular weighting strategy relative to using a single constant which can be thought of as a default strategy where all observations are aggregated together. \(\theta^s\) is analogous to an \(R^2\) measure commonly used in statistics.

A major factor in the performance of a weighting strategy is the relative size of the structural variation compared to the statistical noise. For this purpose, we define the index \(\rho\) as follows:

\[
\mu^2 = \text{The average bias in the estimates for the aggregation network} \\
= \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} (v_a(g) - v_a)^2 \\
\rho = \text{The noise-to-bias ratio} \\
= \frac{\sigma_\varepsilon}{\mu}
\]

We found it useful to group sets of statistics based on the magnitude of \(\rho\). For problems with low noise \((\rho \approx 1)\), we expect to see higher weights on the disaggregate level, whereas problems with larger values of \(\rho\) should weight the aggregate level more. Typical observations of function values for different values of \(\rho\) are shown in figure 3.

All the runs were conducted using a series of scalar functions chosen for their ability to illustrate the techniques using different levels of structural error and noise. We performed experiments on both linear functions, which exhibit a fixed and easy to adjust level of structural error, and a series of nonlinear functions, depicted in figure 4. For each function, we vary the level of statistical noise, allowing us to control the noise to bias index \(\rho\). We vary the number of levels of aggregation used and the number of nodes that are clustered to form an aggregate node.

### 7.2 Experimental Results - Parameters Known

Our analysis of functions characterized by known parameters (which effectively requires that we know the actual function) is intended only as a demonstration of the effectiveness of
Figure 5: A piecewise constant function with its aggregate approximation. Estimates of values of each attribute vector are computed at both the aggregate and disaggregate levels. A weighted averaging is done to improve the estimates.

our optimal weighting strategy, as well as to serve as a benchmark for the strategy which assumes independence of estimates at different levels of aggregation.

In the experiments in this section, we only consider two levels of aggregation, allowing us to use equations (35) and (36) for the weights. We observe that the weights given by either method are functions of the bias in the value at the aggregate level, the variance of the statistical noise in the observation of the values and the number of observations at either level. In order to compare the values of the weights from the competing strategies, we create scenarios with different combinations of the parameters that would produce significant changes in the weights. We then analyze how the variations in the weights given by WOPT and WIND affect the actual function estimates computed using the two schemes.

We consider the piecewise constant function in figure 5 that is monotone and has varying slopes in different regions in its domain. Also shown is an aggregate version of the function. We note that there are distinct regions in the domain where the bias is high, intermediate and zero - we expect the relative weights to be very different in these three regions. Figure 6

27
Figure 6: A comparison of the weights over the function domain for $\rho = 2$

illustrates the weights produced by the optimal formula, WOPT, and the formula assuming independence, WIND. As we would expect, the optimal weights at the disaggregate level are zero when there is no structural error, in contrast to WIND. When the structural error is highest, the optimal weights on the disaggregate level are slightly higher than the WIND weights. The results show that the difference in the weights increases as the noise increases, and is greatest when the bias is smallest.

We have illustrated the difference in the weights produced by the two strategies, but less obvious is the difference in the estimates of the underlying function. Figure 7 compares the performances of the two weighting strategies, for the three levels of noise. We observe that the performance of WOPT and WIND are almost identical even though there were situations where the weights given by the two schemes were significantly different.

The similarity in the function estimates from the two strategies is explained as follows. From our analysis of the weights in section 6, we observed that the difference in the estimates computed by the two schemes was a product of the bias and the difference in the weights, $\Delta w$, and could be bounded from above (equations (37) and (40)). Our experimental results show that the difference is most pronounced when the variance of the statistical noise is
Figure 7: Comparison of the performance, as measured by $\theta^*$, of WOPT and WIND in estimating the piecewise constant function large relative to the bias in the estimate at the aggregate level. Therefore, the weights are most different when the quantities we combine are most similar and in those cases where $v^{(0)}$ and $v^{(1)}$ are most different, the difference in the weights is negligible. Thus, in either of the extreme cases, a negligible bias would offset a large difference in the weights or vice versa, causing the difference in the estimates given by the two schemes - WOPT and WIND, to be marginal. From this analysis, we conclude that WIND, which combines estimates assuming independence, will generally be a very accurate approximation, with the added benefit of being easier to use.

We tested the relative performance of the two methods for other function classes such as linear, concave non-monotone, sinusoidal and arbitrary functions. We summarize the results in Table 2. The figures denote the performance measure as a function of the average number of observations per disaggregate cell. We observe that there is very little statistical difference between the performance of the two methods. We conclude from our experiments that the two schemes, WOPT and WIND, are almost equivalent for all practical purposes.
Table 2: Comparison of WOPT and WIND for various function types using expected values of weights. We use a moderate value of $\rho (= 2)$. The entries in the table denote the average performance measure ($\theta_s$) over 100 samples, along with its standard deviation.

| Avg. num. of obs. per cell | Linear | Conca
tve | Sinusoidal | Random |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WOPT</td>
<td>WIND</td>
<td>WOPT</td>
<td>WIND</td>
</tr>
<tr>
<td></td>
<td>WOPT</td>
<td>WIND</td>
<td>WOPT</td>
<td>WIND</td>
</tr>
<tr>
<td></td>
<td>WOPT</td>
<td>WIND</td>
<td>WOPT</td>
<td>WIND</td>
</tr>
<tr>
<td>2</td>
<td>0.9529</td>
<td>0.9531</td>
<td>0.7880</td>
<td>0.8419</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0002</td>
<td>0.0002</td>
<td>0.0010</td>
<td>0.0010</td>
</tr>
<tr>
<td>4</td>
<td>0.9761</td>
<td>0.9762</td>
<td>0.8862</td>
<td>0.8860</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0005</td>
<td>0.0005</td>
</tr>
<tr>
<td>6</td>
<td>0.9839</td>
<td>0.9839</td>
<td>0.9272</td>
<td>0.9274</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0003</td>
<td>0.0003</td>
</tr>
<tr>
<td>10</td>
<td>0.9899</td>
<td>0.9899</td>
<td>0.9532</td>
<td>0.9532</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>12</td>
<td>0.9915</td>
<td>0.9915</td>
<td>0.9612</td>
<td>0.9612</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>16</td>
<td>0.9932</td>
<td>0.9931</td>
<td>0.9719</td>
<td>0.9717</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
<tr>
<td>20</td>
<td>0.9945</td>
<td>0.9945</td>
<td>0.9775</td>
<td>0.9774</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
</tbody>
</table>

7.3 Experimental Results - Parameters Unknown

We now turn our attention to the more realistic setting where the underlying function is not known, requiring us to compute all quantities from observed data. We still retain our assumption that the statistical noise in the observations is independent of the attribute vector observed. We define the following:

\[
s_{a}^{(g)}^2 = \text{The sample variance of the observations corresponding to the estimate } \hat{v}_{a}^{(g)}.
\]

\[
= \frac{1}{N_a^{(g)} - 1} \sum_{n \in N_{a_i}^{(g)}} (\hat{v}_n - \hat{v}_{a}^{(g)})^2
\]

\[
\hat{\mu}_{a}^{(g)} = \text{An estimate of the bias in the estimate, } \hat{v}_{a}^{(g)}, \text{ with respect to the true value.}
\]

\[
= \hat{v}_{a}^{(g)} - \hat{v}_{a}^{(0)}
\]

\[
\bar{\sigma}^2 = \text{An estimate of the variance in the statistical noise, } \sigma_{\varepsilon}^2.
\]

\[
= \frac{\sum_{a \in A} \sigma_{a}^{(0)^2} N_a^{(0)}}{\sum_{a \in A} N_a^{(0)}}
\]
Table 3: Comparison of performance of WOPT and WIND for different values of $\rho$ for a linear function. The entries denote the average $\theta^*$ over 100 samples, using estimates of weights given by the two strategies.

<table>
<thead>
<tr>
<th>Avg. num.of obs.per cell</th>
<th>$\rho = 1$</th>
<th>$\rho = 2$</th>
<th>$\rho = 5$</th>
<th>$\rho = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WOPT</td>
<td>WIND</td>
<td>WOPT</td>
<td>WIND</td>
</tr>
<tr>
<td>2</td>
<td>0.9654</td>
<td>0.9655</td>
<td>0.9385</td>
<td>0.9396</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0007</td>
<td>0.0007</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>4</td>
<td>0.9891</td>
<td>0.9891</td>
<td>0.9673</td>
<td>0.9678</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
<tr>
<td>6</td>
<td>0.9936</td>
<td>0.9936</td>
<td>0.9782</td>
<td>0.9784</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
<tr>
<td>10</td>
<td>0.9963</td>
<td>0.9963</td>
<td>0.9875</td>
<td>0.9875</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
<tr>
<td>12</td>
<td>0.9969</td>
<td>0.9969</td>
<td>0.9896</td>
<td>0.9896</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>16</td>
<td>0.9978</td>
<td>0.9978</td>
<td>0.9919</td>
<td>0.9919</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>20</td>
<td>0.9983</td>
<td>0.9983</td>
<td>0.9935</td>
<td>0.9935</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

We use these estimates to form approximations of the weights given by equations (35) and (36). We first test these strategies on a simple linear function with varying levels of statistical noise (linear functions allow us to more directly control the noise to structure index $\rho$). We later compare the strategies for a moderate value of $\rho = 2$ for several non-linear functions. The results are summarized in tables 3 and 4.

Given the fairly close results between WOPT and WIND found in this and the previous sections and the computational burden involved in using WOPT, we no longer consider the WOPT strategy, but turn to approximations of WIND as well as some heuristic strategies which are described in the next section.

7.4 Heuristic strategies

In this section, we compare a series of heuristic strategies, some of which are approximations of WIND. We test them on the same types of functions, but now consider multiple levels ($|G| > 2$) of aggregation.

- WIMSE - weighting the estimates at different levels of aggregation by the inverse of
Table 4: Comparison of average $\theta^*$ for $\rho = 2$ across different nonlinear functions, using estimates of weights given by WOPT and WIND strategies

<table>
<thead>
<tr>
<th>Avg. num. of obs. per cell</th>
<th>Concave monotone</th>
<th>Concave non-monotone</th>
<th>Sinusoidal</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WOPT</td>
<td>WIND</td>
<td>WOPT</td>
<td>WIND</td>
</tr>
<tr>
<td>2</td>
<td>0.9173</td>
<td>0.9186</td>
<td>0.7022</td>
<td>0.7876</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0004</td>
<td>0.0004</td>
<td>0.0011</td>
<td>0.0010</td>
</tr>
<tr>
<td>4</td>
<td>0.9562</td>
<td>0.9568</td>
<td>0.8437</td>
<td>0.8459</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0002</td>
<td>0.0002</td>
<td>0.0007</td>
<td>0.0007</td>
</tr>
<tr>
<td>6</td>
<td>0.9738</td>
<td>0.9741</td>
<td>0.9045</td>
<td>0.9273</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0003</td>
<td>0.0003</td>
</tr>
<tr>
<td>10</td>
<td>0.9853</td>
<td>0.9854</td>
<td>0.9390</td>
<td>0.9396</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>12</td>
<td>0.9878</td>
<td>0.9879</td>
<td>0.9509</td>
<td>0.9512</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>16</td>
<td>0.9909</td>
<td>0.9909</td>
<td>0.9635</td>
<td>0.9638</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
<tr>
<td>20</td>
<td>0.9925</td>
<td>0.9925</td>
<td>0.9711</td>
<td>0.9713</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0001</td>
</tr>
</tbody>
</table>

the estimates of their mean squared deviations (obtained as the sum of the variances and the biases) from the true value.

$$w_a^{(g)} = \frac{1}{\sum_{g' \in G} \frac{1}{N_a^{(g')}} + \mu_a^{(g')^2}} \left( \sum_{g' \in G} \frac{1}{N_a^{(g')}} + \mu_a^{(g')^2} \right)^{-1}$$

(42)

- WISAV - weighting the estimates in inverse proportion to the variances of the samples.

$$w_a^{(g)} = \frac{1}{s_a^{(g)^2}} \left( \sum_{g' \in G} \frac{1}{s_a^{(g')^2}} \right)^{-1}$$

(43)

- WENUM - weighting the estimates by the corresponding number of observations.

$$w_a^{(g)} = \frac{N_a^{(g)}}{\sum_{g' \in G} N_a^{(g')}}$$

(44)

- MINV - dynamically choosing the estimate with the least sample variance.

$$\bar{v}_a = \bar{v}_a^{(g*)}$$

where, $g* = \arg \min_{g' \in G} s_a^{(g')^2}$

(45)
Figure 8: Comparison of the various strategies in estimating a linear function with $\rho = 5$

Figure 9: Comparison of the various strategies in estimating a linear function with $\rho = 2$

Figure 8 shows a plot of the performance measure, $\theta^*$, as a function of the average number of observations per cell for $\rho = 5$. Here, we considered four levels of aggregation, each aggregate node being formed by clustering two nodes at a more disaggregate level. We
observe that for fewer observations per cell, the method WENUM that weights the estimates by the number of observations outperforms the rest. However, it soon converges to a suboptimal value. The strong performance of WENUM for high noise and few observations reflects the challenge of weighting based on statistics when the data is extremely noisy. However, the poor performance in the limit makes this approach undesirable as a general purpose strategy. The method WISAV, which uses weights inversely proportional to the sample variances (rather than the variances of the statistics, which requires dividing by the number of observations), outperforms the remaining methods, although WIMSE and WIND are catching up in the limit. WISAV implicitly puts a higher weight on the more aggregate level, which tends to work better when the noise level is very high.

This performance is reversed in figure [9] for the lower noise case $\rho = 2$, where we considered three levels of aggregation, with three nodes at a more disaggregate level being used to form

### Table 5: Comparison of the various strategies as an average over several aggregations for a linear function

<table>
<thead>
<tr>
<th># of obs per cell</th>
<th>MINV</th>
<th>WENUM</th>
<th>WISAV</th>
<th>WIMSE</th>
<th>WIND</th>
<th>MINV</th>
<th>WENUM</th>
<th>WISAV</th>
<th>WIMSE</th>
<th>WIND</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho = 1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.9584</td>
<td>0.9219</td>
<td>0.9617</td>
<td>0.9626</td>
<td><strong>0.9695</strong></td>
<td>0.9117</td>
<td>0.9090</td>
<td><strong>0.9300</strong></td>
<td>0.9252</td>
<td>0.9251</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0010</td>
<td>0.0009</td>
<td>0.0009</td>
<td>0.0009</td>
<td>0.0008</td>
<td>0.0022</td>
<td>0.0016</td>
<td>0.0017</td>
<td>0.0018</td>
<td>0.0019</td>
</tr>
<tr>
<td>4</td>
<td>0.9804</td>
<td>0.9301</td>
<td>0.9823</td>
<td>0.9832</td>
<td><strong>0.9851</strong></td>
<td>0.9423</td>
<td>0.9226</td>
<td><strong>0.9582</strong></td>
<td>0.9549</td>
<td>0.9570</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0004</td>
<td>0.0003</td>
<td>0.0002</td>
<td>0.0004</td>
<td>0.0004</td>
<td>0.0013</td>
<td>0.0008</td>
<td>0.0009</td>
<td>0.0010</td>
<td>0.0010</td>
</tr>
<tr>
<td>10</td>
<td>0.9926</td>
<td>0.9341</td>
<td>0.9911</td>
<td><strong>0.9945</strong></td>
<td>0.9944</td>
<td>0.9731</td>
<td>0.9313</td>
<td>0.9783</td>
<td>0.9806</td>
<td><strong>0.9809</strong></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0006</td>
<td>0.0004</td>
<td>0.0004</td>
<td>0.0004</td>
<td>0.0004</td>
</tr>
<tr>
<td>16</td>
<td>0.9950</td>
<td>0.9355</td>
<td>0.9923</td>
<td><strong>0.9965</strong></td>
<td>0.9964</td>
<td>0.9822</td>
<td>0.9336</td>
<td>0.9820</td>
<td><strong>0.9878</strong></td>
<td><strong>0.9878</strong></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0004</td>
<td>0.0003</td>
<td>0.0003</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td>20</td>
<td>0.9958</td>
<td>0.9358</td>
<td>0.9925</td>
<td><strong>0.9972</strong></td>
<td>0.9971</td>
<td>0.9850</td>
<td>0.9344</td>
<td>0.9831</td>
<td><strong>0.9899</strong></td>
<td><strong>0.9899</strong></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0001</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0001</td>
<td>0.0003</td>
<td>0.0003</td>
<td>0.0003</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th># of obs per cell</th>
<th>MINV</th>
<th>WENUM</th>
<th>WISAV</th>
<th>WIMSE</th>
<th>WIND</th>
<th>MINV</th>
<th>WENUM</th>
<th>WISAV</th>
<th>WIMSE</th>
<th>WIND</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho = 5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.6689</td>
<td><strong>0.8342</strong></td>
<td>0.7610</td>
<td>0.7303</td>
<td>0.6749</td>
<td>-0.0704</td>
<td><strong>0.6768</strong></td>
<td>0.2223</td>
<td>0.1611</td>
<td>0.0575</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0094</td>
<td>0.0042</td>
<td>0.0065</td>
<td>0.0075</td>
<td>0.0117</td>
<td>0.1812</td>
<td>0.0283</td>
<td>0.0976</td>
<td>0.1119</td>
<td>0.0902</td>
</tr>
<tr>
<td>4</td>
<td>0.7510</td>
<td><strong>0.8834</strong></td>
<td>0.8447</td>
<td>0.8072</td>
<td>0.8092</td>
<td>0.1964</td>
<td><strong>0.7613</strong></td>
<td>0.4986</td>
<td>0.3893</td>
<td>0.3355</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0069</td>
<td>0.0026</td>
<td>0.0045</td>
<td>0.0055</td>
<td>0.0060</td>
<td>0.1310</td>
<td>0.0225</td>
<td>0.0762</td>
<td>0.0910</td>
<td>0.0567</td>
</tr>
<tr>
<td>10</td>
<td>0.8756</td>
<td>0.9156</td>
<td><strong>0.9352</strong></td>
<td>0.9141</td>
<td>0.9180</td>
<td>0.6161</td>
<td><strong>0.8461</strong></td>
<td>0.7841</td>
<td>0.7187</td>
<td>0.7510</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0030</td>
<td>0.0013</td>
<td>0.0017</td>
<td>0.0022</td>
<td>0.0020</td>
<td>0.0821</td>
<td>0.0161</td>
<td>0.0404</td>
<td>0.0547</td>
<td>0.0366</td>
</tr>
<tr>
<td>16</td>
<td>0.9142</td>
<td>0.9226</td>
<td><strong>0.9540</strong></td>
<td>0.9414</td>
<td>0.9421</td>
<td>0.8080</td>
<td><strong>0.8985</strong></td>
<td>0.8978</td>
<td>0.8483</td>
<td>0.8484</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0019</td>
<td>0.0011</td>
<td>0.0011</td>
<td>0.0015</td>
<td>0.0015</td>
<td>0.0120</td>
<td>0.0064</td>
<td>0.0086</td>
<td>0.0145</td>
<td>0.0158</td>
</tr>
<tr>
<td>20</td>
<td>0.9270</td>
<td>0.9261</td>
<td><strong>0.9600</strong></td>
<td>0.9514</td>
<td>0.9518</td>
<td>0.8293</td>
<td>0.9045</td>
<td><strong>0.9122</strong></td>
<td>0.8692</td>
<td>0.8687</td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0017</td>
<td>0.0009</td>
<td>0.0010</td>
<td>0.0013</td>
<td>0.0013</td>
<td>0.0145</td>
<td>0.0075</td>
<td>0.0105</td>
<td>0.0202</td>
<td>0.0229</td>
</tr>
</tbody>
</table>
Table 6: Comparison of the various strategies for different non-linear function classes. The performance measures are averages over values of $\rho = 1, 2$ and 5, and aggregation trees with 2, 3 and 4 levels, and with 2, 3 and 4 nodes clustered to form an aggregate node.

<table>
<thead>
<tr>
<th>Avg. num. of obs per cell</th>
<th>Concave monotone</th>
<th>Function type</th>
<th>Concave non-monotone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MINV WENUM WISAV WIMSE WIND</td>
<td>MINV WENUM WISAV WIMSE WIND</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.8027 0.8589 0.8521 0.8405 0.8251</td>
<td>0.3530 0.5313 0.5126 0.4704 0.4189</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0045 0.0023 0.0033 0.0035 0.0040</td>
<td>0.0132 0.0065 0.0094 0.0104 0.0143</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.8626 0.8913 0.9112 0.8947 0.9000</td>
<td>0.5296 0.6245 0.6946 0.6373 0.6546</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0032 0.0012 0.0019 0.0025 0.0024</td>
<td>0.0097 0.0035 0.0056 0.0071 0.0071</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.9031 0.9020 0.9406 0.9284 0.9337</td>
<td>0.6613 0.6601 0.7926 0.7459 0.7608</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0020 0.0008 0.0012 0.0015 0.0013</td>
<td>0.0081 0.0024 0.0037 0.0053 0.0047</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.9392 0.9103 0.9625 0.9564 0.9578</td>
<td>0.7839 0.6852 0.8636 0.8441 0.8486</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0012 0.0006 0.0006 0.0008 0.0008</td>
<td>0.0042 0.0018 0.0022 0.0027 0.0026</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.9481 0.9123 0.9670 0.9633 0.9641</td>
<td>0.8162 0.6926 0.8804 0.8712 0.8740</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0010 0.0005 0.0005 0.0007 0.0007</td>
<td>0.0034 0.0017 0.0019 0.0023 0.0022</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.9595 0.9148 0.9724 0.9719 0.9723</td>
<td>0.8558 0.7014 0.8993 0.9003 0.9010</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0008 0.0004 0.0004 0.0005 0.0005</td>
<td>0.0023 0.0014 0.0015 0.0016 0.0016</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.9667 0.9160 0.9752 0.9771 0.9773</td>
<td>0.8805 0.7078 0.9106 0.9202 0.9205</td>
<td></td>
</tr>
<tr>
<td>std. dev.</td>
<td>0.0006 0.0004 0.0004 0.0004 0.0004</td>
<td>0.0022 0.0012 0.0013 0.0014 0.0014</td>
<td></td>
</tr>
</tbody>
</table>

| Sinusoidal Random | MINV WENUM WISAV WIMSE WIND | MINV WENUM WISAV WIMSE WIND |                     |
|-------------------|-----------------------------|-----------------------------|                     |
| 2                 | 0.5375 0.6676 0.6504 0.6224 0.5835 | -0.6533 -0.0812 -0.3919 -0.3867 -0.4638 |                     |
| std. dev.         | 0.0106 0.0056 0.0081 0.0085 0.0140 | 0.0346 0.0163 0.0259 0.0282 0.0344 |                     |
| 4                 | 0.6887 0.7420 0.7891 0.7558 0.7699 | -0.1802 0.1234 0.1730 0.0919 0.1408 |                     |
| std. dev.         | 0.0064 0.0029 0.0045 0.0052 0.0050 | 0.0225 0.0069 0.0143 0.0168 0.0175 |                     |
| 6                 | 0.7609 0.7663 0.8505 0.8224 0.8337 | 0.1515 0.1943 0.4275 0.3806 0.4159 |                     |
| std. dev.         | 0.0054 0.0022 0.0032 0.0039 0.0028 | 0.0152 0.0052 0.0096 0.0116 0.0127 |                     |
| 10                | 0.8544 0.7881 0.9062 0.8943 0.8974 | 0.4516 0.2526 0.6079 0.6189 0.6306 |                     |
| std. dev.         | 0.0031 0.0016 0.0019 0.0024 0.0024 | 0.0088 0.0033 0.0050 0.0061 0.0057 |                     |
| 12                | 0.8755 0.7919 0.9157 0.9095 0.9118 | 0.5277 0.2684 0.6468 0.6793 0.6868 |                     |
| std. dev.         | 0.0028 0.0014 0.0016 0.0019 0.0019 | 0.0070 0.0028 0.0040 0.0047 0.0045 |                     |
| 16                | 0.9021 0.7990 0.9294 0.9300 0.9310 | 0.6143 0.2861 0.6866 0.7458 0.7495 |                     |
| std. dev.         | 0.0020 0.0011 0.0012 0.0014 0.0014 | 0.0055 0.0021 0.0032 0.0036 0.0034 |                     |
| 20                | 0.9173 0.8038 0.9373 0.9430 0.9436 | 0.6734 0.2978 0.7104 0.7884 0.7904 |                     |
| std. dev.         | 0.0016 0.0010 0.0010 0.0011 0.0011 | 0.0047 0.0019 0.0027 0.0031 0.0030 |                     |

an aggregate node. In this case, WISAV does not appear to be catching up to WIMSE or WIND in the limit. We also note for this case that WENUM is virtually equivalent to the other methods when there are relatively few observations, while performing very poorly in the limit.

Irrespective of the other parameters that are varied, the relative performance of the various strategies seem to be a function of the noise-to-bias ratio. In order to examine how
significant the results are, we take the average of the performance measures over several problems for each strategy while varying values of the noise-to-bias ratio and tabulate the results in table 5. We observe that WIMSE and WIND outperform the other strategies in most of the cases. The exceptions are the runs with high noise and relatively few observations per cell.

We summarize our results for concave monotone, sinusoidal and random function types in Table 6. The values denote the average performance measure over several problems with different values of the noise-to-bias ratio. The relative performance of the various strategies seems to be independent of the function class considered. Here again, we see that it is a toss-up between WIMSE and WISAV, with WISAV dominating when there are fewer observations and WIMSE emerging as the best strategy in the remaining cases, especially when the function does not exhibit any structure.

Since the use of the WIND strategy as an approximation of equation (36) is conditional on the statistical noise being independent of the attribute vector sampled and also because the heuristic WIMSE strategy produces almost identical results, we propose WIMSE as a suitable alternative to WIND.

8 Experiments in an approximate dynamic programming application

We applied the heuristic strategies proposed in section 7.4 to the nomadic trucker problem described in section 2. We consider a problem with 40 locations at the trade area level which can be aggregated to 35 regions and 10 areas. The other attributes that we consider are the trailer type and the day of week. The trailer type attribute can take on one of three values

<table>
<thead>
<tr>
<th>Day of week</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>1.0</td>
</tr>
<tr>
<td>Tuesday</td>
<td>0.8</td>
</tr>
<tr>
<td>Wednesday</td>
<td>0.6</td>
</tr>
<tr>
<td>Thursday</td>
<td>0.7</td>
</tr>
<tr>
<td>Friday</td>
<td>0.9</td>
</tr>
<tr>
<td>Saturday</td>
<td>0.2</td>
</tr>
<tr>
<td>Sunday</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 7: Parameters for the nomadic trucker problem
Table 8: Aggregations for the nomadic trucker problem. A ‘∗’ corresponding to a particular attribute indicates that the attribute is included in the attribute vector, and a ‘−’ indicates that it is aggregated out.

<table>
<thead>
<tr>
<th>Aggregation level</th>
<th>Location</th>
<th>Trailer type</th>
<th>Day of week</th>
<th>Size of state space</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Trade area</td>
<td>*</td>
<td>*</td>
<td>$40 \times 3 \times 7 = 840$</td>
</tr>
<tr>
<td>1</td>
<td>Region</td>
<td>-</td>
<td>*</td>
<td>$35 \times 1 \times 7 = 245$</td>
</tr>
<tr>
<td>2</td>
<td>Region</td>
<td>-</td>
<td>-</td>
<td>$35 \times 1 \times 1 = 35$</td>
</tr>
<tr>
<td>3</td>
<td>Area</td>
<td>-</td>
<td>-</td>
<td>$10 \times 1 \times 1 = 10$</td>
</tr>
</tbody>
</table>

- small, medium and large. The day of week attribute of the resulting state is dependent on the time taken by the trucker to reach his destination, which is a function of the distance between his origin and destination. The day of week attribute, combined with the origin-destination pair, determines the probability that a load will appear on that particular day, to be moved to the destination city. We use a load probability distribution according to which loads are more likely to appear during the beginning of the week (Mondays) and towards the end (Fridays). The load probability dips during the middle of the week and is lower over the weekends. We list the parameters that we use in table 7.

In table 8, we list the aggregations that we use for the problem and the number of attribute states at each level of aggregation.

In order to compute the true values associated with each attribute vector, we use a standard value iteration algorithm where we loop over all the attribute vectors, updating the values using the best combination of current reward and future value, until the values converge. We use our aggregation strategies to form estimates of these values, by incorporating them in the forward dynamic programming algorithm outlined in figure 2. The generic estimation formula given in step 4 of this algorithm is replaced by the following equation:

$$
\bar{v}_{a_d}^{n-1} = \sum_{g \in G} w_{a_d}^{(g),n-1} v_{a_d}^{(g),n-1} \quad \forall d \in D_a
$$

(46)

where $w_{a_d}^{(g),n-1}$ denotes the weight, at iteration $(n-1)$, on the estimate of the value of attribute vector $a_d$ at the $g$th level of aggregation. We point out that the algorithm is one that does pure exploration, that is, at each iteration, the new attribute vector state is generated at
random from the space of all possible states. This way, we focus purely on the problem of statistical estimation of the values.

We compare the strategies WIMSE, WISAV and WENUM against the static aggregation strategy that uses only the disaggregate estimate. For the static aggregation scheme, in the absence of visits to a particular attribute vector state, we use an average of all the observations as an estimate of the value. For the strategies that use aggregation, there will be an estimate of the value for every attribute vector at some level of aggregation, even though some of the states might not have been visited at all at the most disaggregate level.

We use a normalized error measure ($\eta$) to compare the performance. The performance measure for strategy $s$ is computed using the following formula,

$$
\eta^s = \frac{\sqrt{\epsilon^s}}{\sqrt{\sum_{a \in A} \nu_a^2}}
$$

(47)

where $\epsilon^s$ is the sum of squared errors defined in section 7.1 and $\nu_a$ is the true value associated with attribute vector $a$.

We tested our strategies for both myopic and non-myopic assignments of the resource. We considered cases where the rewards were deterministic and also ones where the rewards were stochastic with a Gaussian distribution. In the myopic experiments, the downstream value of the resource is ignored while choosing the best possible assignment. Thus the problem is reduced to one where, after each observation, a decision is made based on the current rewards alone. For this set of experiments, we can safely assume that the observations of values corresponding to each state form a stationary series. Therefore, it would be appropriate to use a stepsize of $1/n$, where $n$ denotes the number of observations corresponding to a particular attribute vector.

In the non-myopic case, we take into account the downstream value of the resource. We used a factor of 0.9 for discounting the future rewards. Adding in the downstream value would cause the observation of the value of the current state to be biased. We can no longer assume stationarity in the observations. Hence, we use a stepsize rule that addresses nonstationarity in order to update the value estimates.
A comparison of the weights, generated by WISAV and WIMSE strategies, on the estimates at different levels of aggregation is shown in figure 10. The results are taken as an average over several myopic runs of the forward dynamic programming algorithm with deterministic rewards. The non-myopic runs give similar, but noisier values of the weights. This illustrates how WISAV puts a steady weight on the aggregate estimates (Levels 1, 2 & 3) irrespective of the number of observations, whereas the WIMSE weights keep tending towards the disaggregate estimate (Level 0) with more observations.
Figure 11: Percentage error for myopic runs as a function of the number of observations.

Figure 11 illustrates the performance of the competing schemes for myopic runs. For a small number of observations, the performance of the various methods is indistinguishable. As expected, the static aggregation scheme does not work best initially, but soon outperforms the other methods with more observations. The WENUM scheme is seen to stabilize at a very suboptimal solution, the reason being that it weights the most aggregate level highly. Both WISAV and WIMSE outperform the disaggregate strategy initially. WISAV levels off to a larger error value since it implicitly weights the more aggregate estimates which, in
Figure 12: Percentage error for non-myopic runs as a function of the number of observations.

This case, are of poorer quality than the disaggregate estimate. WIMSE, however, performs comparably to the disaggregate strategy irrespective of the number of observations and is seen to be the best overall.

The results from the runs where we consider the downstream value of the resource are shown in figure 12. The behavior is very similar to the myopic case as in all the weighted strategies give better values initially as compared to the disaggregate scheme. The fast con-
vergence of the weighted strategies (as compared to the myopic case) is as a result of the stepsizes used for updating the estimates. The stepsizes take into account the nonstationarity in the observations and adjust themselves accordingly so as to give updated estimates that best approximates the true value. WENUM and WISAV are found to tend towards suboptimal solutions. WIMSE once again outperforms the remaining methods, giving very small error values. The relative behavior of the various methods seem to be unaffected by the nature of the rewards - it is seen to be very similar whether the rewards are deterministic or stochastic.

9 Conclusion

We have studied a variety of strategies for combining estimates obtained at different levels of aggregation. We derived an optimal weighting strategy, but found that it did not produce noticeably different results than a simpler weighting scheme that assumed that the observations were independent. A simple analysis of a two-level aggregation scheme explained this behavior. We then compared several strategies, and identified two that were promising. The strategy WISAV, which weights estimates in inverse proportion to estimates of the sample variances at each level of aggregation, was found to do well, especially when the noise-to-bias ratio was relatively high and there were only a few observations per cell. The WISAV strategy has the effect of artificially inflating the weights on the more aggregate estimates, which also produces consistent errors in the limit. The WIMSE strategy, which weights statistics according to the variance of the estimate, correcting using an estimate of the bias, produces the best results in the limit and good initial results, although not as good as WISAV in high noise cases.

The weighting schemes were made use of in an approximate dynamic programming application for the nomadic trucker problem, where values associated with individual attribute vectors had to be estimated. The weighting strategies were incorporated into the forward dynamic programming algorithm and the estimates compared to the true values. The weighting schemes were found to produce better results initially, with the static aggregation scheme at the disaggregate level catching up in the limit. The weighting scheme WIMSE is found
to be very robust and gives estimates of good quality as compared to the other methods irrespective of the number of observations.

There may be applications in approximate dynamic programming where it is known roughly how many observations will be made per cell, and the level of noise. In such applications, it may be possible to preselect a strategy which will work the best. If this is not the case, it seems desirable to choose a strategy such as WIMSE which will work well in the limit, even if it underperforms a strategy such as WISAV initially.

References


43

