Two Algorithms for Inducing Causal Models from Data

Dawn E. Gregory and Paul R. Cohen

Computer Science Technical Report 94-42

Experimental Knowledge Systems Laboratory
Computer Science Department, Box 34610
Lederle Graduate Research Center
University of Massachusetts
Amherst, MA 01003-4610

Abstract

Many methods have been developed for inducing cause from statistical data. Those employing linear regression have historically been discounted, due to their inability to distinguish true from spurious cause. We present a regression-based statistic that avoids this problem by separating direct and indirect influences. We use this statistic in two causal induction algorithms, each taking a different approach to constructing causal models. We demonstrate empirically the accuracy of these algorithms.

This work is supported by ARPA/Rome Laboratory under contract #'s F30602-91-C-0076 and F30602-93-C-0100. To appear in *Proceedings of AAAI-94 Workshop on Knowledge Discovery in Databases.*

1 Causal Modeling

Causal modeling is a method for representing complex causal relationships within a set of variables. Often, these relationships are presented in a **directed, acyclic graph**. Each node in the graph represents a variable in the set, while the links between nodes represent direct, causal relationships that follow the direction of the link. Each link is annotated with the attributes of the relationship; for example, a numeric *weight value* is often used to indicate the strength of the (linear) relationship between the two variables. In addition, the annotations will indicate any relationships that are non-linear. An example of a causal model is shown in figure 1.

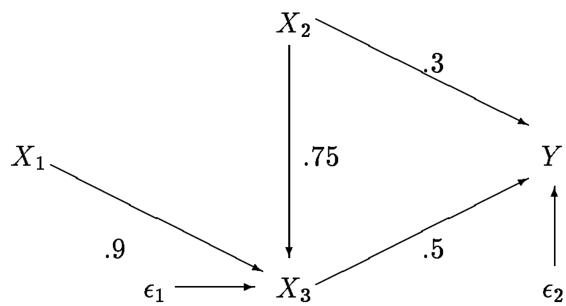

Figure 1: A simple causal model

This model shows direct causal influences between X_1 and X_3, X_2 and X_3, X_2 and Y, and X_3 and Y; as indicated by the coefficients of the links, these influences have strength .9, .75, .3, and .5 respectively. We refer to the variable Y as the *sink* variable, in that it has no outgoing links and thus "absorbs" all influences. Note that each predicted variable has an additional influence, ϵ_i; these *error terms* account for unexplained variance in the data, such as measurement error or unrecorded influences. We can also represent this model as a set of *structural equations*:

$$\begin{aligned} X_3 &= .9X_1 + .75X_2 + \epsilon_1 \\ Y &= .3X_2 + .5X_3 + \epsilon_2 \end{aligned}$$

Causal models are built in order to provide an explicit, understandable description of the causal influences within some system of variables. In the past, causal models were constructed manually and fine-tuned to reflect features found in the data. Recently, considerable effort has been directed towards automatically deriving a causal model from the probability distributions found in the data, a process called *causal induction.*[4, 5]

The problem of inducing cause from data alone is notoriously difficult. Suppes [6] established three reasonable criteria for a causal relationship: covariance, control, and temporal precedence. The *covariance*, or similarity, between two variables, can be measured through simple statistical tests. In order to show *control*, we must ensure that no other variables are responsible for this covariance; this feature can also be tested statistically, with one of many *conditional independence* tests.[1] Unfortunately, we meet a considerable obstacle when trying to show temporal precedence; the occurrence of one variable "before" another cannot be proven from post-hoc data alone. Thus, without additional knowledge or experimentation, any so-called causal relationship will remain a hypothesis.

Several methods have been developed for statistically hypothesizing temporal precedence. Many of these utilize the distributions of the statistics expected for certain configurations of variables in

[1] Conditional independence tests cannot establish control when the covariance is caused by a variable not included in the set.

a model. For example, the IC algorithm (see [4]) uses features of conditional independence to hypothesize the direction of a causal influence:

> Let $I(i, j \mid x)$ denote the independence of i and j given a variable x, and let $I(i, j \mid *)$ denote the independence of i and j given some other variable.[2] Then, for three variables a, b and c, when $I(a, c \mid *)$ is true, and all of $I(a, b \mid *)$, $I(c, b \mid *)$ and $I(a, c \mid b)$ are false, we have the configuration $a \rightarrow b \leftarrow c$.

In other words, we expect there are links between a and b and between c and b, but not between a and c. In addition, we know that b does not *separate*, or render independent, variables a and c. This fact excludes three configurations: $a \rightarrow b \rightarrow c$, $a \leftarrow b \leftarrow c$, and $a \leftarrow b \rightarrow c$, leaving $a \rightarrow b \leftarrow c$ as the only possible configuration. Thus, the statistic is used to induce the direction of the links from this known property of conditional independence.

The process of causal induction is complicated further by the introduction of *latent variables* into the model. A latent variable is one that is not included in the data, but has causal influence on one or more variables that are included. Whenever the latent variable influences two other variables, a causal induction algorithm may place a causal link between them when, in reality, there is no causal influence. This behavior has been cited as a major argument against the possibility of causal induction.

2 FBD and FTC

FBD and FTC are two causal induction algorithms we have recently implemented. Both utilize a set of statistical *filter conditions* to remove links from the model being constructed.[3] FBD constructs a model by applying the filter conditions to select a set of *predictors* for each variable. FTC constructs a model directly from the set of filtered links by sorting them according to a *precedence* function $S(x_i \rightarrow x_j)$. In the current implementation, both algorithms are deterministic; neither performs a search of the model space defined by the filtered set of links. However, this avenue is open for future research.

2.1 Filter Conditions

Both FBD and FTC rely on a set of methods for filtering links from the set of possible links. Each of these filters is based on some statistic $F(x_i \rightarrow x_j)$; when the value of $F(x_i \rightarrow x_j)$ falls outside of a specific range, the link $x_i \rightarrow x_j$ is removed from the set of links that can be included in the model.

2.1.1 Linear Regression

The first set of filters verifies that the result of linear regression indicates a sufficiently strong, linear relationship between variables x_i and x_j.[4] For any predictee x_j, x_j is regressed on $\{x_1 \ldots x_{j-1}, x_{j+1} \ldots x_n\}$, producing betas: $\{\beta_{1j} \ldots \beta_{(j-1)j}, \beta_{(j+1)j} \ldots \beta_{nj}\}$. Whenever any of these β_{ij} are close to 0, lower than a threshold T_β, the link $x_i \rightarrow x_j$ is discarded. In addition, the value of $R_j^2 = \sum_{i \neq j} r_{ij}\beta_{ij}$

[2] In truth, $I(i, j \mid *)$ should denote the independence of i from j given *any subset* of the other variables. In practice, we often use only the subsets of size 1.

[3] CLIP/CLASP [1] provides statistical support for FBD and FTC. In fact, each algorithm can be run within CLASP's Lisp-listener interface. FBD and FTC are available as part of the CLIP/CLASP package. If interested, please contact David Hart (dhart@cs.umass.edu).

[4] Causal relationships need not be linear; however, FTC and FBD assume they are. This drawback can often be avoided by applying appropriate transformations to the data prior to running the algorithm.

must be sufficiently high for each variable x_j (r_{ij} is the standardized correlation between x_i and x_j). R_j^2 measures the amount of variance in x_j that is captured by the regression. When this is low, below a threshold T_{R^2}, all links to x_j are discarded. These filters enforce Suppes' *covariance* condition for a causal relationship.

2.1.2 The ω Statistic

The primary contribution of the FBD and FTC algorithms is the introduction of the ω statistic:

$$\omega_{ij} = \left| \frac{r_{ij} - \beta_{ij}}{r_{ij}} \right| = \left| 1 - \frac{\beta_{ij}}{r_{ij}} \right|$$

The ω filter discards a link $x_i \rightarrow x_j$ whenever ω_{ij} is larger than a preset threshold T_ω.[5] Because the regression coefficient, β_{ij}, is computed with respect to other variables, ω_{ij} measures the fraction of x_i's influence on x_j that is *not* direct (i.e. goes through other variables). Thus, ω is also a means of enforcing Suppes' *control* condition: if x_i's direct influence on x_j is a small percentage of its total influence, the relationship between x_i and x_j is moderated by other variables.

It should be noted that the *other variables* used in the regression can be any subset of the potential predictors, and the value of ω_{ij} may vary across subsets. In order to avoid the (exponential) exhaustive search of these subsets, FBD and FTC start with the largest subset (all potential predictors) for the initial regression. Whenever further regressions are computed, x_j is regressed on all variables x_i for which the link $x_i \rightarrow x_j$ has not been removed by the filters. Our empirical studies show that this approach provides excellent results while the algorithms remain polynomial in complexity.[6]

2.1.3 Other filters

Many other measurements can be used to filter links. Currently, the only other filter used by FBD and FTC is a test for *simple conditional independence*, similar to that used by the IC algorithm. In this test, we compute the partial correlation coefficient of x_i and x_j given some other variable x_k ($k \neq i$ and $k \neq j$). If x_k renders x_i and x_j independent, the partial correlation will be approximately 0, and we will discard the link between x_i and x_j. Like the ω filter, the conditional independence filter enforces the *control* condition. An experiment described below shows the difference between the effects of these control conditions.

2.2 The fbd Algorithm

The FBD algorithm was our first attempt at building causal models utilizing these filter conditions. FBD is told which variable is the sink variable, y, and proceeds as follows:

1. Enqueue y into an empty queue, Q.
2. Create M, an empty model (with n nodes and no links) to build on.
3. While Q is not empty, do:
   (a) Dequeue a variable x_j from Q

[5] We want the direct influence, β_{ij}/r_{ij}, to be close to 1; thus, $\omega_{ij} = | 1 - \beta_{ij}/r_{ij} |$ should be near 0.

[6] The complexity of these algorithms is $O(n^4)$, where n is the number of variables. Most of this is attributed to the linear regressions, which have complexity $O(n^3)$ in our implementation.

(b) Find a set of predictors $P = \{x_i \mid x_i \neq x_j$ and $x_i \rightarrow x_j$ passes all filter conditions and $x_i \rightarrow x_j$ will not cause a cycle$\}$

(c) For each $x_i \in P$, add the link $x_i \rightarrow x_j$ into M

(d) For each $x_i \in P$, Enqueue x_i into Q

Our pilot experiments indicated that good performance can be achieved with this approach[3]; however, two significant drawbacks were noted. The first is that we must provide FBD with knowledge not usually provided to causal induction algorithms: the identity of the sink variable. Although one expects algorithms perform better with additional knowledge, FBD does not work without it. The second problem is an effect of the order in which variables are predicted, called *premature commitment*. This problem surfaces in the following situation:

> Suppose we decide that variable y has two predictors, x_i and x_j. After adding the links $x_i \rightarrow y$ and $x_j \rightarrow y$ to the model, we proceed to put x_i and x_j onto the queue, in that order. Now, suppose the true state of the world is that $x_i \rightarrow x_j$. When we remove x_i from the queue and select its predictors, x_j will be one of them.[7] Thus, FBD will insert the link $x_j \rightarrow x_i$ into the model, which is incorrect.

These two drawbacks motivated the development of another causal induction algorithm, FTC.

2.3 The ftc Algorithm

FTC deals with the problems of FBD by inserting links into the model in order of precedence, rather than in order of selection. Precedence is determined by a sorting function $S(x_i \rightarrow x_j)$. Although FTC does not completely resolve the premature commitment problem, it does significantly reduce the number of reversed links, while eliminating the need for additional knowledge. The FTC algorithm is as follows:

1. Let $L = \{x_i \rightarrow x_j : i \neq j, 1 \leq i \leq n; 1 \leq j \leq n\}$; i.e. L is the set of all potential links in a model with n variables.

2. For each link $x_i \rightarrow x_j \in L$, test each filter condition for $x_i \rightarrow x_j$. If *any* condition fails, remove $x_i \rightarrow x_j$ from L.

3. Sort the links remaining in L by some precedence function $S(x_i \rightarrow x_j)$.

4. Create M, an empty model (with n nodes and no links) to build on.

5. While L is not empty, do:

   (a) Remove the link $x_i \rightarrow x_j$, of the highest precedence, from L.

   (b) If $x_i \rightarrow x_j$ does not cause a cycle in M, add $x_i \rightarrow x_j$ to M. Otherwise, discard $x_i \rightarrow x_j$.

The models constructed by this algorithm depend greatly on the statistic used as the precedence function. For example, note that FBD is a special case of FTC, where the sort function is: $S(x_i \rightarrow x_j) = \beta_{ij} + Order(j) * n$, where n is the number of variables, and

$$Order(j) = \begin{cases} n & \text{when } j \text{ is the sink variable} \\ \max_{x_j \rightarrow x_k}(Order(k)) - 1 & \text{otherwise} \end{cases}$$

[7]This is true because the link from x_i and x_j guarantees the conditions of covariance and control, so all filters will be passed.

So the links to the sink variable have the highest precedence $(n^2 + \beta_{ij})$, and the predictors of the sink variable will be rated by their respective betas $(n(n-1) + \beta_{ij})$. In the next section, we describe an experiment in which we tried several different sorting statistics in order to determine which would be an appropriate precedence function.

3 Empirical Results

In spite of the simplicity of these algorithms, their empirical performance is very good. We compared FBD and FTC with two other causal induction algorithms, IC[4] and PC[5]. Both of these take a least-commitment approach to causal induction, conservatively assigning direction to very few links in order to avoid misinterpretation of potential *latent* influences. FBD and FTC, on the other hand, commit to a direction in all cases.

3.1 Input to the Experiments

In these initial experiments, we worked with a set of 60 *artificially generated* data sets: 20 data sets for each of 6, 9, and 12 variables. These were generated from the structural equations of 60 randomly selected *target models*. The advantage of this approach is that the model constructed by each algorithm can be evaluated against a known target.

The target models were constructed by randomly selecting m links from the set of potential links $L = \{x_i \to x_j \mid i \neq j\}$. For each model of n variables, m is chosen from the range $1.0(n-1)\ldots3.0(n-1)$; thus the target models have an *average branching factor* between 1 and 3.

As each link is selected, it is inserted into the target model. With probability 0.3, the link will be a *correlation* link, indicating the presence of a latent variable.[8] Although neither FBD or FTC can detect correlation links, their presence is critical if we are to believe the artificial data are similar to real data.

Once the structure of each target model has been determined, the *structural equations* are created for each dependent variable x_j. For directed links $x_i \to x_j$, a path coefficient is randomly selected from the range $-1.0\ldots1.0$. For correlation links, a latent variable l_{ij} is created (these variables are not included in the final data set), and a path coefficient is selected for the links $l_{ij} \to x_i$ and $l_{ij} \to x_j$.

Finally, the data are generated from the structural equations. For each independent and latent variable, a set of 50 data points are sampled from a Gaussian distribution with mean of 0 and standard deviation of 1. Sample values for the dependent variables are computed from the structural equations, and a Gaussian error term is added to each (also with mean 0 and standard deviation 1). The resulting data set can now be used as input to the causal induction algorithms.

3.2 Evaluating the Output

To measure the performance of each causal induction algorithm, we use several types of evaluation; each is intended to capture a different aspect of the causal model.

The R^2 statistic measures the amount of variance that can be attributed to the predictors of each variable. In theory, the best set of predictors for a variable will produce the highest possible value of R^2; thus, the strength of any predictor set can be evaluated through its R^2 value. When evaluating model performance, the R^2 value of the dependent variable, called *Dependent*R^2, is of primary interest; however, we also want to consider the other variables in the model. Specifically,

[8] This is called a *correlation* link because the latent variable induces a correlation between the variables.

we compute a ΔR^2 score by computing the mean of the absolute differences in R^2 between the dependent variables in the target model and the model being evaluated; ideally, this value will be 0. These measures indicate how well the model accounts for variance in each variable.

We also compare models directly to the target models from which the data were generated. We are concerned with the percentage of the links (directed) in the target model that were correctly identified ($Correct\%$), the ratio of wrong links found for every correct one ($Wrong/Correct$), the number of links that were identified but had the wrong direction ($WrongReversed$), and the number of links that were completely wrong ($WrongNotRev$). These measures indicate how close the model is to the model that generated the data.

3.3 Evaluation of Sort Heuristics

In the first experiment, we wanted to determine an appropriate precedence function, $S(x_i \rightarrow x_j)$, for the third step of the FTC algorithm. We compared several statistical measures, including ω_{ij}, β_{ij}, and R_j^2. One additional set of trials used unrelated pseudo-random numbers to establish a baseline for comparison. The results of this experiment are shown in Table 1.

$Measure$	$SortFunction$	$6vars$	$9vars$	$12vars$
$DependentR^2$	$Random$	0.562 (0.330)	0.405 (0.336)	0.566 (0.299)
	ω_{ij}	0.442 (0.364)	0.570 (0.315)	0.472 (0.326)
	β_{ij}	0.366 (0.302)	0.333 (0.363)	0.305 (0.331)
	R^2	0.547 (0.350)	0.689 (0.194)	0.640 (0.337)
ΔR^2	$Random$	0.240 (0.101)	0.249 (0.072)	0.238 (0.067)
	ω_{ij}	0.271 (0.085)	0.267 (0.085)	0.257 (0.070)
	β_{ij}	0.291 (0.111)	0.299 (0.085)	0.281 (0.071)
	R^2	0.209 (0.112)	0.140 (0.102)	0.186 (0.066)
$Correct\%$	$Random$	0.406 (0.170)	0.339 (0.125)	0.387 (0.136)
	ω_{ij}	0.401 (0.229)	0.385 (0.146)	0.297 (0.141)
	β_{ij}	0.268 (0.179)	0.232 (0.119)	0.233 (0.131)
	R^2	0.564 (0.214)	0.630 (0.223)	0.470 (0.173)
$Wrong/Correct$	$Random$	1.649 (1.723)	2.505 (1.510)	1.626 (0.658)
	ω_{ij}	2.124 (2.121)	2.058 (1.087)	2.627 (1.219)
	β_{ij}	3.529 (2.357)	5.537 (4.923)	4.489 (4.381)
	R^2	0.698 (0.549)	0.789 (0.577)	1.123 (0.608)
$WrongReversed$	$Random$	2.750 (1.446)	5.150 (1.899)	5.900 (1.917)
	ω_{ij}	2.850 (1.461)	4.850 (1.899)	8.000 (2.865)
	β_{ij}	4.000 (1.376)	6.650 (1.954)	9.450 (3.034)
	R^2	1.600 (1.501)	1.900 (1.683)	3.900 (2.198)
$WrongNotRev.$	$Random$	1.300 (1.031)	3.700 (2.105)	6.350 (3.083)
	ω_{ij}	1.350 (0.988)	3.800 (2.093)	6.250 (3.093)
	β_{ij}	1.350 (0.988)	3.950 (2.235)	6.200 (3.302)
	R^2	1.150 (0.988)	3.450 (2.114)	5.950 (3.052)

Table 1: Means and (Standard Deviations) of scores for several precedence functions

Overall, the best results are obtained with the R^2 statistic. In some sense, this is to be expected: we would like to give precedence to the links that predict well predicted variables. Although it seems as though more information about precedence could be obtained from pairs of variables, it is possible to construct the correct model by ordering the variables rather than the links.

In addition, notice the low variance in the *Wrong Not Reversed* category. This effect has a logical explanation: since most of the wrong links are discarded by the filter conditions, the differences between these scores indicate links that were removed to avoid cycles.

3.4 Comparative Performance

Next, we compared each algorithm with another causal induction algorithm. FBD was compared with the PC algorithm [5], which has the ability to deal with external constraints, such as knowledge of the sink variable. FTC was compared to the IC algorithm, since neither uses external knowledge about the data. The results follow:

Measure	*Algorithm*	*6vars*	*9vars*	*12vars*
Dependent R^2	*FBD∗*	0.734 (0.187)	0.787 (0.146)	0.728 (0.278)
	PC∗	0.301 (0.396)	0.403 (0.307)	0.363 (0.320)
ΔR^2	*FBD∗*	0.118 (0.087)	0.134 (0.077)	0.179 (0.130)
	PC∗	0.333 (0.151)	0.321 (0.102)	0.372 (0.100)
Correct%	*FBD∗*	0.653 (0.186)	0.651 (0.184)	0.528 (0.248)
	PC∗	0.284 (0.181)	0.273 (0.172)	0.167 (0.109)
Wrong/Correct	*FBD∗*	0.685 (0.542)	0.932 (0.391)	1.396 (1.082)
	PC∗	0.458 (0.534)	0.617 (0.921)	1.276 (1.315)
WrongReversed	*FBD∗*	1.200 (1.056)	2.200 (1.824)	4.100 (3.323)
	PC∗	0.650 (0.813)	0.950 (1.146)	1.950 (1.146)
WrongNotRev.	*FBD∗*	1.900 (1.165)	4.900 (1.997)	9.100 (4.388)
	PC∗	0.250 (0.444)	0.300 (0.470)	0.950 (0.945)

Table 2: Means and (Standard Deviations) of scores using additional knowledge.

Measure	*Algorithm*	*6vars*	*9vars*	*12vars*
Dependent R^2	*FTC*	0.547 (0.350)	0.689 (0.194)	0.640 (0.337)
	IC	0.326 (0.389)	0.451 (0.338)	0.303 (0.363)
ΔR^2	*FTC*	0.209 (0.112)	0.140 (0.102)	0.186 (0.066)
	IC	0.347 (0.128)	0.345 (0.137)	0.348 (0.095)
Correct%	*FTC*	0.564 (0.214)	0.630 (0.223)	0.470 (0.173)
	IC	0.293 (0.18)	0.272 (0.19)	0.165 (0.11)
Wrong/Correct	*FTC*	0.698 (0.549)	0.789 (0.577)	1.123 (0.608)
	IC	1.48 (0.944)	1.50 (1.11)	2.17 (2.10)
WrongReversed	*FTC*	1.600 (1.501)	1.900 (1.683)	3.900 (2.198)
	IC	1.55 (1.05)	1.35 (1.27)	2.0 (1.30)
WrongNotRev.	*FTC*	1.150 (0.988)	3.450 (2.114)	5.950 (3.052)
	IC	1.2 (1.06)	2.5 (2.09)	2.9 (1.8)

Table 3: Means and (Standard Deviations) of scores without additional knowledge.

These results show no unexpected differences between comparable algorithms. First, since FBD and FTC are based on linear regression, better R^2 scores are expected for these algorithms. Second, PC and IC assign direction to very few links. Since only directed links are included in the scores, the differences in the *Correct%*, *Wrong Reversed*, and *Wrong Not Reversed* are to be expected; FBD and FTC will find more correct links at the cost of finding more incorrect links. Note that the ratio *Wrong/Correct* is slightly better (lower) for FTC than for IC, although this difference is not statistically significant.

3.5 The Effects of the ω Filter

We also wanted to determine if the ω filter is the key to the success of these algorithms, so we ran FBD and FTC without this filter condition. The results are shown below:

Again, there are no statistically significant differences, so we cannot say that ω is wholly responsible for the performance of FBD and FTC. There is, however, a definite effect on the models

Measure	*Algorithm*	*6vars*	*9vars*	*12vars*
DependentR^2	*FBDwith*	0.734 (0.187)	0.787 (0.146)	0.728 (0.278)
	FBDwithout	0.749 (0.173)	0.811 (0.117)	0.755 (0.225)
ΔR^2	*FBDwith*	0.118 (0.087)	0.134 (0.077)	0.179 (0.130)
	FBDwithout	0.101 (0.067)	0.119 (0.065)	0.157 (0.069)
Correct%	*FBDwith*	0.653 (0.186)	0.651 (0.184)	0.528 (0.248)
	FBDwithout	0.696 (0.168)	0.696 (0.139)	0.573 (0.182)
Wrong/Correct	*FBDwith*	0.685 (0.542)	0.932 (0.391)	1.396 (1.082)
	FBDwithout	0.746 (0.635)	1.009 (0.438)	1.445 (0.608)
WrongReversed	*FBDwith*	1.200 (1.056)	2.200 (1.824)	4.100 (3.323)
	FBDwithout	1.350 (1.089)	2.400 (1.847)	4.550 (2.800)
WrongNotRev.	*FBDwith*	1.900 (1.165)	4.900 (1.997)	9.100 (4.388)
	FBDwithout	2.200 (1.673)	6.100 (1.861)	11.300 (3.404)

Table 4: Means and (Standard Devations) of scores for FBD with and without ω filtering

that are constructed; notably, the number of wrong links decreases quite a bit while the number of correct links decreases a little when ω is used. Thus, we can say that ω occasionally discards a correct link, but more often it discards an incorrect link.

4 Conclusions

Our empirical results show that ω does remarkably well as a heuristic for selecting predictors. In fact, it performs so well that very simple model construction algorithms achieve comparable quality to models constructed by very sophisticated algorithms.

Admittedly, neither FBD nor FTC infers the presence of latent variables, which may be a significant drawback for some applications. However, other experiments have shown that both FBD and FTC will often avoid predictors that are connected to the variables they predict via a common but latent cause (see [2]).

Finally, FBD and FTC are simple, polynomial-time algorithms that construct models without searching the entire space of models. We believe it possible to obtain even better results while maintaining this level of complexity by integrating these techniques with others.

Measure	*Algorithm*	*6vars*	*9vars*	*12vars*
DependentR^2	*FTCwith*	0.734 (0.187)	0.787 (0.146)	0.728 (0.278)
	FTCwithout	0.749 (0.173)	0.790 (0.148)	0.736 (0.270)
ΔR^2	*FTCwith*	0.118 (0.087)	0.107 (0.087)	0.147 (0.086)
	FTCwithout	0.124 (0.087)	0.115 (0.091)	0.146 (0.086)
Correct%	*FTCwith*	0.658 (0.203)	0.682 (0.223)	0.566 (0.211)
	FTCwithout	0.675 (0.204)	0.690 (0.208)	0.569 (0.214)
Wrong/Correct	*FTCwith*	0.467 (0.627)	0.696 (0.496)	0.809 (0.288)
	FTCwithout	0.738 (0.618)	0.833 (0.402)	1.037 (0.417)
WrongReversed	*FTCwith*	0.600 (0.681)	1.450 (1.849)	2.650 (1.981)
	FTCwithout	1.000 (1.124)	1.950 (2.350)	3.350 (2.739)
WrongNotRev.	*FTCwith*	1.350 (0.988)	3.500 (1.821)	6.400 (2.798)
	FTCwithout	2.400 (1.465)	4.700 (1.976)	8.150 (3.150)

Table 5: Means and (Standard Devations) of scores for FTC with and without ω filtering

Acknowledgments

This research is supported by ARPA/Rome Laboratory under contract #'s F30602-91-C-0076 and F30602-93-C-0100. The US Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

References

[1] Scott D. Anderson, Adam Carlson, David L. Westbrook, David M. Hart, and Paul R. Cohen. Clip/clasp: Common lisp analytical statistics package/common lisp instrumentation package. Technical Report 93-55, Department of Computer Science, University of Massachusetts, Amherst, MA, 1993.

[2] Lisa Ballesteros. Regression–based causal induction with latent variable models. In *Proceedings of the Twelvth National Conference on Artificial Intelligence*. Morgan Kaufmann, 1994.

[3] Paul R. Cohen, Lisa Ballesteros, Dawn Gregory, and Robert St. Amant. Regression can build predictive causal models. Technical Report 94-15, Dept. of Computer Science, University of Massachusetts, Amherst, 1994.

[4] Judea Pearl and T.S. Verma. A theory of inferred causation. In J. Allen, R. Fikes, and E. Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference.*, pages 441–452. Morgan Kaufman, 1991.

[5] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search.* Springer-Verlag, 1993.

[6] P.C. Suppes. *A Probabilistic Theory of Causality.* North Holland, Amsterdam, 1970.