Two Algorithms for Inducing Structural Equation Models from Data

Paul R. Cohen, Dawn E. Gregory, Lisa Ballesteros, Robert St. Amant
Computer Science Department, LGRC
University of Massachusetts
Box 34610
Amherst, MA 01003-4610, (413) 545 3638
{cohen,gregory,balleste,stamant}@cs.umass.edu

Abstract

We present two algorithms for inducing structural equation models from data. Assuming no latent variables, these models have a causal interpretation and their parameters may be estimated by linear multiple regression. Our algorithms are comparable with PC [15] and IC [12,11], which rely on conditional independence. We present the algorithms and empirical comparisons with PC and IC.

1 Structural Equation Models

Given a dependent variable x_0 and a set of predictor variables $P = \{x_1, x_2, \ldots, x_k\}$, multiple regression algorithms find subsets $p \subset P$ that account for "much" of the variance in x_0. These are search algorithms, and they are not guaranteed to find the "best" p—the one that makes R^2 as big as possible, especially when p is a proper subset of P [10,3,7]. The question arises, what should be done with the variables in $T = P - p$, the ones that aren't selected as predictors of x_0? In many data analysis tasks, the variables in T are used to predict the variables in p. For instance, we might select x_1 and x_3 as predictors of x_0; and x_2, x_5, x_6 as predictors of x_1; and x_4 as a predictor of x_2 and x_3; and so on. We can write *structural equations*:

$$\begin{aligned} x_0 &= \beta_{0,1}x_1 + \beta_{0,3}x_3 + u \\ x_1 &= \beta_{1,2}x_2 + \beta_{1,5}x_5 + \beta_{1,6}x_6 + v \\ x_2 &= \beta_{2,4}x_4 + w \\ x_3 &= \beta_{3,4}x_4 + z \end{aligned}$$

The principal task for any modeling algorithm is to decide, for a given "predictee," which variables should be in p and which should be in T. Informally, we must decide where in a structural equation model a variable does most good. For example, parents' education (PE) and child's education (CE) could be used as predictors of a child's satisfaction when he or she takes a job (JS), but we might prefer a model in which PE predicts CE, and CE predicts JS (or a model in which PE predicts CE *and* JS). This paper presents two algorithms that build structural equation models.

There are clear parallels between building structural equation models and building causal models. Indeed, *path analysis* refers to the business of interpreting structural equation models

as causal models [9,16]. Path analysis has been heavily criticized (e.g., [13,8]) in part because latent variables can produce large errors in estimated regression coefficients throughout a model [15]. Recent causal induction algorithms rely not on regression coefficients but on conditional independence [11,15]. These algorithms use covariance information only to infer boolean conditional independence constraints; they do not estimate strengths of causal relationships, and, most importantly from our perspective, they don't use these strengths to guide the search for causal models.

Our algorithms, called FBD and FTC, use covariance information, in the form of estimated standardized regression coefficients, to direct the construction of structural equation models and to estimate the parameters of the models. Because latent variables can result in biased estimates, our algorithms might be misled when latent variables are at work. In practice, FBD and FTC are more robust than, say, stepwise multiple regression. They often discard predictors that are related to the predictee only through the presence of a latent variable [2]. We haven't yet shown analytically why the algorithms have this advantage. Until we do, our only claim for FBD and FTC is this: when latent variables are at work, our algorithms build multilevel regression models of heuristic value to analysts, just as ordinary regression algorithms build useful (but suspect) single-level models. If we can assume *causal sufficiency* [15]—essentially, no latent variables—these models may be interpreted as causal models [5].[1]

2 FBD and FTC

Structural equation models can be represented by directed acyclic graphs (DAGs) in which a link $x_1 \rightarrow x_0$ is interpreted to mean x_1 is a predictor of x_0. FBD and FTC rely on statistical *filter conditions* to remove insignificant links from the model being constructed. The two algorithms use the same filter conditions, differing only in how they are applied. The most important filter (and the only novel one) is the ω statistic:

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

Given a predictee, say x_i, and a set of predictors P, we first regress x_i on all the predictors in P. This yields a standardized regression coefficient β_{ij} for each predictor x_j. Now, β_{ij} is partial—it represents the effect of x_j on x_i when the influences of all the other variables are fixed. Following [9,14] we'll call β_{ij} an estimate of the *direct* influence of x_j on x_i. The correlation r_{ij} represents the *total* influence of x_j on x_i, so $r_{ij} - \beta_{ij}$ estimates the influence of x_j that is due to its relationships with other predictors—its *indirect* influence. Thus, ω_{ij} estimates the fraction of x_j's total influence on x_i that is indirect. If x_j has little direct influence on x_i relative to its indirect influence through, say, x_k, then x_k, not x_j, is heuristically more apt to be a direct cause of x_i. Thus we filter x_j when ω_{ij} exceeds a threshold T_ω.

Small values of ω are necessary but not sufficient to consider one variable a cause of another. This is because ω_{ji} will be very small if both r_{ij} and β_{ij} are small, but in this case the predictor x_j has little influence on x_i, direct or indirect, and should be filtered. Even if the β coefficients

[1] FBD and FTC run in CLIP/CLASP [1], a Common Lisp statistical package developed at UMass. For more information on CLIP/CLASP, please contact clasp-support@cs.umass.edu.

are relatively high, the regression of x_i on x_j and other variables might account for very little variance in x_i, so we filter x_j if it doesn't contribute enough (perhaps with other variables) to R^2 for x_i. Currently, the only other filter used by FBD and FTC is a test for *simple conditional independence* using the partial correlation coefficient. If x_k renders x_i and x_j independent, the partial correlation $r_{ij \cdot k}$ will be approximately 0, and we remove the link between x_i and x_j.

2.1 The FBD Algorithm

The FBD algorithm is told x_0, the sink variable, and works backwards, finding predictors for x_0, then predictors for those predictors, and so on.

1. Enqueue x_0 into an empty queue, Q.
2. Create M, an empty model (with n nodes and no links).
3. While Q is not empty, do:
   (a) Dequeue a variable x_j from Q
   (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

FBD performs well (see below) but it has two drawbacks: First, we must identify the sink variable x_0; most model induction algorithms do not require this information. Second, predictors of a single variable are enqueued on Q in an arbitrary order, yet the order in which variables are dequeued can affect the structure of the resulting model. For example, the link $x_i \rightarrow x_j$ will be reversed if x_i is dequeued before x_j. These issues led to the development of a second algorithm, called FTC.

2.2 The FTC Algorithm

FTC deals with these problems by inserting links into the model in order of precedence, rather than in order of selection. Precedence is determined by a sorting function $S(x_j \rightarrow x_i)$. The FTC algorithm is as follows:

1. Let $L = \{x_j \rightarrow x_i : i \neq j, 1 \leq i \leq n; 1 \leq j \leq n\}$; i.e. L is the set of all potential links in a model with n variables.
2. For each link $x_j \rightarrow x_i \in L$, test each filter condition for $x_j \rightarrow x_i$. If *any* condition fails, remove $x_j \rightarrow x_i$ from L.
3. Sort the links remaining in L by some precedence function $S(x_j \rightarrow x_i)$.
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 $x_j \rightarrow x_i$, of the highest precedence, from L.

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

Experiments with different sorting functions led us to the following simple procedure: for a link $x_j \rightarrow x_i$, its score for sorting is R^2 from the regression of x_i on all the other variables. Thus, a link's precedence is the R^2 of its dependent variable. We justify this policy with the following observation: variables with high values of R^2 are less likely to have latent influences, so they are preferred as dependent variables.

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.

3 Empirical Results

We have tested FBD and FTC under many conditions: on artificial data (using our own data generator and the TETRAD generator), on published data [15], and on data generated by running an AI planning system called PHOENIX [6]. We tested how well the ω heuristic selects predictors, measured in terms of variance accounted for in the dependent variable [4]. Also, we have compared FBD and FTC with other algorithms—PC for FBD [15] and IC for FTC [11].[2] Finally, to assess how latent variables affected its performance, we compared FBD with stepwise linear regression as implemented in MINITAB [2].

3.1 Artificial Models and Data

We worked with a set of 60 artificially generated data sets: 20 data sets for each of 6, 9, and 12 measured 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 $L = \{x_i \rightarrow x_j \mid i \neq j\}$. For each model of n variables, m is chosen from the range $1.0(n-1) \ldots 3.0(n-1)$; 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.

Once the structure of each target model has been determined, the *structural equations* are created for each dependent variable x_j. For directed links $x_i \rightarrow x_j$, a path coefficient is randomly selected from the range $-1.0 \ldots 1.0$. 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 $l_{ij} \rightarrow x_i$ and $l_{ij} \rightarrow x_j$.

Finally, data are generated from the structural equations. For each independent and latent variable, a set of 50 data points is sampled from a Gaussian distribution with mean of 0 and standard deviation of 1 (we have also run experiments with variates selected from uniform

[2] We are very grateful to Professors Spirtes, Glymour and Scheines, and Professor Pearl, for making the code for their algorithms available.

distributions). Sample values for the dependent variables are computed from the structural equations, and a Gaussian error term is added to each (with mean 0 and standard deviation 1).

3.2 How Good Is ω at Selecting Predictors?

Suppose we have a dataset with one dependent variable x_0 and a set P of predictor variables, and we want to find the best subset *p–best*(k) of k predictors. Here, "best" means no other subset p' of k predictors accounts for more variance in x_0 than *p–best*(k) does. We can find *p–best*(k) by exhaustive search of all subsets of k predictors. Or, we can regress x_0 on all the predictors in P, and select the k predictors with the lowest ω scores. How well does the latter method perform in comparison with the former, exhaustive method?

For each of the 60 target models described in the previous section, we found by exhaustive search the best subsets of $k = 3, 4, 5$ predictors of the "sink" variable, x_0. Next we selected subsets of k predictors using ω scores. The *batch discarding* method regresses x_0 on $x_1, x_2, \ldots$, calculates ω for each of these predictors, and selects the k predictors with the best ω scores. This set is denoted *p–batch*(k). The *iterative discarding* method repeatedly regresses x_0 on a set of predictors, discarding the predictor with the worst ω score, until only k predictors remain, denoted *p–iter.*(k). We chose to try both methods since one can expect β coefficients to change substantially even if one predictor is removed from a regression.

If ω scores select good predictors, then *p–best*(k), *p–batch*(k), and *p–iter.*(k) should contain the same predictors; if they don't, we would like *p–batch*(k) and *p–iter.*(k) to account for nearly as much of the variance in x_0 as *p–best*(k).

Table 1 shows how many predictors *p–batch*(k) and *p–iter.*(k) have in common with *p–best*(k). For 12-variable models and $k = 5$, the mean number of predictors shared by *p–batch*(k) and *p–best*(k) is 3.15, and, for *p–iter.*(k) and *p–best*(k) this number is 3.375. Thus, when batch discarding selects five variables, roughly two of them (on average) are not the best variables to select.

k	*Vars.*	$\lvert p-batch(k) \cap p-best(k) \rvert$	$\lvert p-iter.(k) \cap p-best(k) \rvert$
5	12	3.15 (1.16)	3.375 (.86)
5	9	3.65 (.49)	3.7 (.52)
5	6	5.0 (0)	5.0 (0)
4	12	2.3 (.57)	2.3 (.74)
4	9	3.05 (.36)	3.08 (.53)
4	6	3.33 (.23)	3.33 (.23)
3	12	1.48 (.61)	1.68 (.58)
3	9	2.18 (.46)	2.18 (.51)
3	6	2.28 (.36)	2.33 (.33)

Table 1: Means and (Standard Deviations) of the size of the intersections

On the other hand, Table 2 shows that the variables in *p–batch*(k) and *p–iter.*(k) account for almost as much of the variance in x_0 as those in *p–best*(k). Let $R^2_{p-best(k)}$ be the variance in x_0 that is "available" to predictors in *p–batch*(k) and *p–iter.*(k). For instance, if the best k predictors account for only 50% of the variance in x_0, then we want to express the predictive power of *p–batch*(k) as a fraction of 50%, not 100%. Table 2 therefore contains the ratios

$R^2_{p-batch(k)}/R^2_{p-best(k)}$ and $R^2_{p-iter.(k)}/R^2_{p-best(k)}$. For example, for 12-variable models and $k = 5$, the predictors selected by batch discarding account for 85% of the available variance in x_0, on average, and those selected by iterative discarding account for 86%.

k	*Vars.*	$R^2_{p-batch(k)}/R^2_{p-best(k)}$	$R^2_{p-iter.(k)}/R^2_{p-best(k)}$
5	12	.845 (.03)	.86 (.023)
5	9	.94 (.006)	.94 (.005)
5	6	1.0 (0)	1.0 (0)
4	12	.80 (.04)	.82 (.03)
4	9	.94 (.008)	.94 (.005)
4	6	.91 (.01)	.91 (.01)
3	12	.73 (.06)	.80 (.04)
3	9	.91 (.02)	.92 (.01)
3	6	.88 (.03)	.89 (.03)

Table 2: Means and (Standard Deviations) of the R^2 ratios

Batch and iterative discarding do not find exactly the same predictors as exhaustive search for the best predictors, but the ones they find account for much of the available variance in the dependent variable. Bear in mind that exhaustive search for the best k of N variables requires N-choose-k multiple regressions with k predictors, whereas iterative discarding requires $N - k$ multiple regressions with between N and $k + 1$ predictors, and batch discarding requires just one regression with N predictors. Thus, batch discarding finds predictors that are nearly as good as exhaustive search, with a fraction of the effort.

We wondered whether something simpler than ω scores, such as sorting by beta coefficients, would perform equally well. In most cases, beta coefficients selected the same predictors as ω scores, but sometimes they recommended different sets with bad R^2 scores. We can see why (and also why ω is preferable) by considering four cases:

High r_{0i}, high β_{0i}: In this case ω is small in absolute value and x_i will be accepted as a predictor of x_0. Similarly, β is high, so by this criterion x_i will also be accepted as a predictor. Since x_i should be accepted in this case, both ω and β do the right thing.

High r_{0i}, low β_{0i}: Here, ω is large in absolute value and β small, so both statistics will reject x_i, which is the right thing to do.

Low r_{0i}, high β_{0i}: Here, ω is large in absolute value so x_i will be rejected. However, x_i's β score suggests accepting it. The correct action is to reject x_i because the only way to get, say, $r_{0i} = 0$ and $\beta_{0i} = .8$ is for x_i's direct influence (.8) to be cancelled by its indirect influence (-.8) through other predictors. We want predictors that have large direct influence and small indirect influence, so we ought to discard x_i. In this case, β coefficients make the wrong recommendation.

Low r_{0i}, low β_{0i}: In this case, ω is small but FBD and FTC will discard x_i as a predictor because β is small.

3.3 Comparative Studies

FBD and FTC were compared to the PC [15] and IC [12,11] algorithms, respectively. We chose these algorithms for comparison due to their availability and strong theoretical support. Both PC and IC build models from constraints imposed by *conditional independencies* in the data.

In order to provide a comprehensive evaluation of the algorithms, we used a variety of performance measures for each model. The measures we used are shown in table 3. See [5,4] for details of these and other dependent measures.

Measure	Meaning
Dependent R^2	The variance accounted for in x_0, the sink variable.
ΔR^2	The mean of the absolute differences in R^2 between all the predictees in the target model and the model being evaluated.
Correct %	The percentage of the directed links in the target model that were correctly identified.
Wrong/Correct	The ratio of wrong links found for every correct one.
Wrong Reversed	The number of links $x_i \rightarrow x_j$ that should have been reversed, $x_i \leftarrow x_j$.
Wrong Not Reversed	The number of links $x_i \rightarrow x_j$ that should not have been drawn in either direction.

Table 3: Dependent measures and their meanings.

We compared FBD with the PC algorithm [15], because PC can be given exogenous knowledge about causal order, specifically, it can be told which is the sink variable x_0.[3] FTC was compared to the IC algorithm, since neither uses external knowledge about the data. Both PC and IC 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. Hence it was necessary to evaluate statistics like $DependentR^2$ for PC and IC models in two ways—by interpreting undirected links as directed (choosing the most favorable direction, always), or simply ignoring undirected links. In Table 4 the last two rows for the R^2 measures, denoted *w/undirected*, give scores obtained by interpreting undirected links as directed in the most favorable way.

FBD and FBD attained significantly higher $DependentR^2$ and ΔR^2 scores than PC and IC (much higher, when we ignore undirected links). This is to be expected: FBD and FTC are driven by covariance information and PC and IC use covariance only to find conditional independence relationships. (Significance was tested with paired-sample t tests, $p < .05$.) FBD and FTC have significantly higher *Correct%* scores than PC and IC. FBD has a higher $Wrong/Correct$ score than PC, although not significantly so, because it commits to more directed links than PC. As the models become larger, FTC becomes significantly better than IC on $Wrong/Correct$ scores. Although FBD and FTC include more links than the other algorithms, FTC maintains a low rate of incorrect identifications. Roughly 72% of PC's wrong links are wrong because they are backwards (i.e., $WrongReversed$); conversely, only 28% of PC's links should not have been drawn in either direction. For IC, 44% of the wrong links are backwards, and for FBD and FTC, the numbers are 33% and 30%, respectively. Clearly, when FBD and FTC draw an

[3] We thank Professor Glymour for suggesting this comparison.

incorrect link, it generally shouldn't be in the model pointing in either direction, whereas PC and to a lesser extent IC err by drawing links backwards. Keep in mind, though, that this is an analysis of wrong links, only, and that FTC has better *Wrong/Correct* performance than the other algorithms.

Measure	*Algorithm*	*6vars*	*9vars*	*12vars*
DependentR^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)
	FTC	0.734 (0.187)	0.787 (0.146)	0.728 (0.278)
	IC	0.326 (0.389)	0.451 (0.338)	0.303 (0.363)
w/correlations	*PC**	0.607 (0.337)	0.685 (0.266)	0.684 (0.269)
	IC	0.603 (0.31)	0.811 (0.12)	0.739 (0.216)
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)
	FTC	0.118 (0.087)	0.107 (0.087)	0.147 (0.086)
	IC	0.347 (0.128)	0.345 (0.137)	0.348 (0.095)
w/correlations	*PC**	0.185 (0.084)	0.166 (0.101)	0.193 (0.051)
	IC	0.199 (0.104)	0.124 (0.063)	0.172 (0.063)
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)
	FTC	0.658 (0.203)	0.682 (0.223)	0.566 (0.211)
	IC	0.293 (0.18)	0.272 (0.19)	0.165 (0.11)
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)
	FTC	0.467 (0.627)	0.696 (0.496)	0.809 (0.288)
	IC	1.48 (0.944)	1.50 (1.11)	2.17 (2.10)
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)
	FTC	0.600 (0.681)	1.450 (1.849)	2.650 (1.981)
	IC	1.55 (1.05)	1.35 (1.27)	2.0 (1.30)
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)
	FTC	1.350 (0.988)	3.500 (1.821)	6.400 (2.798)
	IC	1.2 (1.06)	2.5 (2.09)	2.9 (1.8)

Table 4: Means and (Standard Deviations) of scores for several causal induction algorithms (* = uses additional knowledge)

3.4 Performance with Latent Variables

Regression coefficients are unstable, especially when unmeasured or *latent* variables influence them. Selecting variables by their ω scores lessens this problem. In our lab, Ballesteros [2] has obtained good results with models Spirtes et al. [15, page 240] show are difficult for ordinary regression. For these models, regression often chooses predictors whose relationships to the dependent variable are mediated by latent variables or common causes (we refer to these variables as proxy variables). Ballesteros generated 12 different sets of coefficients for the structural equations for each of the four Spirtes et al. models, and data sets for each having 100 to

1000 variates. FBD's performance on these data was measured by the number of times it chose the correct predictors and number of times it incorrectly chose a proxy variable. The data from these experiments are given in Table 5. FBD correctly chose true predictors 90% of the time, and correctly rejected proxy variables 80% of the time.

	chosen	rejected	Total
True Predictor	108	12	120
Proxy Variable	12	48	60
Total	120	60	180

Table 5: Contingency Table FBD Choices

As a comparison, Ballesteros used Minitab to run stepwise regressions (significance = .05) on each latent variable dataset to find out how often the algorithm chose a proxy variable as a predictor. The data are given in Table 6.

	chosen	rejected	Total
True Predictor	118	2	120
Proxy Variable	42	18	60
Total	160	20	180

Table 6: Contingency Table Stepwise Choices

MINITAB correctly selected 98% of the true predictors, but it correctly rejected proxies only 30% of the time. We are trying to understand why stepwise regression performs so poorly in the presence of latent variables and why FBD is less susceptible. Our best guess is that stepwise regression is a "step up" procedure that starts with no predictors and adds them one at a time, whereas FBD starts with all predictors, computes regression coefficients, and then removes some predictors with filter conditions. We think that a regression coefficient for a predictor is less likely to be biased when it is one of several variables in a regression 0 , than when it is one of very few predictors. If so, step up procedures will be more susceptible to bias than, say, backward elimination and FBD. Ballesteros has some evidence to support this conjecture.

4 Conclusions

FBD and FTC are simple, polynomial-time algorithms that construct models without searching the entire space of models. Our empirical results show that ω does remarkably well as a heuristic for selecting predictors. In fact, it performs so well that FBD and FTC build models that are in some respects superior to those constructed by PC and IC. Admittedly, neither FBD nor FTC infers the presence of latent variables, which may be a significant drawback for some applications. However, we have shown that FBD will often avoid predictors that are connected to the variables they predict via a common but latent cause.

Acknowledgments

This work is supported by ARPA/Rome Laboratory under contract #'s F30602-91-C-0076 and F30602-93-C-0010, and by a NASA GSRP Training Grant, NGT 70358. The U.S. government is authorized to reproduce and distribute reprints for governmental purposes not withstanding 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. This document is available under http://eksl-www.cs.umass.edu/publications.html.

[2] Lisa Ballesteros. Regression–based causal induction with latent variable models. In *Proceedings of the Twelfth National Conference on Artificial Intelligence.* Morgan Kaufmann, 1994. This document is available under http://eksl-www.cs.umass.edu/publications.html.

[3] Jacob Cohen and Patricia Cohen. *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences.* Hillsdale, NJ, 1975.

[4] Paul R. Cohen, Lisa Ballesteros, Dawn Gregory, and Robert St. Amant. Experiments with a regression–based causal induction algorithm. EKSL memo number 94-33, Dept. of Computer Science, University of Massachusetts/Amherst. This document is available under http://eksl-www.cs.umass.edu/publications.html, 1994.

[5] 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. This document is available under http://eksl-www.cs.umass.edu/publications.html.

[6] Paul R. Cohen, David M. Hart, Robert St. Amant, Lisa Ballesteros, and Adam Carlson. Path analysis models of an autonomous agent in a complex environment. In *Selecting Models from Data: AI and Statistics IV.*

[7] N. R. Draper and H. Smith. *Applied Regression Analysis.* New York, NY, 1966.

[8] Paul Holland. Statistics and causal inference. *Journal of the American Statistical Association*, 81:945–960, 1986.

[9] C.C. Li. *Path analysis–A primer.* Boxwood Press, 1975.

[10] Frederick Mosteller and John W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison Wesley, Reading, MA, 1977.

[11] Judea Pearl and T.S. Verma. A statistical semantics for causation. *Statistics and Computing*, 2:91–95, 1991.

[12] 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.

[13] Juliet Popper Shaffer, editor. *The Role of Models in Nonexperimental Social Science: Two Debates.* American Educational Research Association and American Statistical Association, 1992.

[14] Robert R. Sokal and F. James Rohlf. *Biometry: the Principles and Practice of Statistics in Biological Research.* W.H. Freeman and Co., New York, second edition, 1981.

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

[16] Sewall Wright. Correlation and causation. *Journal of Agricultural Research*, 20:557–585, 1921.