Paper ST-10

An Exact Implicit Enumeration Algorithm for Variable Selection in Multiple Linear Regression Models Using Information Criteria

Dr. Dennis Beal, Science Applications International Corporation, Oak Ridge, Tennessee

ABSTRACT

For large multivariate data sets the data analyst often wants to know the best set of independent regressors to use in a multiple linear regression model. Akaike's Information Criteria (AIC) is one information criterion calculated in SAS® that is used to score a model. For a small number of independent variables *p*, an explicit enumeration of all possible 2^p models is possible. However, for large multivariate data sets where *p* is large, an explicit enumeration of all possible models becomes computationally intractable. This paper presents SAS code that implements the exact implicit enumeration algorithm authored by Bao (2005) that has been shown to always arrive at the globally optimal minimum AIC value when let run to completion. The number of models evaluated to determine the optimal model with the smallest AIC score is minimal and shown to be much more efficient than an explicit enumeration of all possible models. A large multivariate data set is simulated with a known true model to demonstrate how fast the exact implicit enumeration algorithm arrives at the true model. The number of models evaluated is compared to an explicit enumeration algorithm and the `REG` procedure in SAS. This paper is for intermediate SAS users of SAS/STAT who understand multivariate data analysis and SAS macros.

Key words: Akaike's Information Criteria, multiple linear regression, model selection, implicit enumeration

INTRODUCTION

Multiple linear regression is one of the classical statistical tools used for discovering relationships between variables. It can be used to find the linear model that best predicts the dependent variable Y from the independent X variables. A data set with *p* independent variables or regressors has 2^p possible subset models to consider since each of the *p* variables is either included or excluded from the intercept model, not counting interaction terms. Akaike's Information Criteria (AIC) is calculated for each model with the objective to identify the model that minimizes the information criteria. AIC is the monotonic nonlinear objective function to minimize. Information criteria balance the trade-off between a lack of fit term and a penalty term for the number of regressors in the model. SAS 9.2 currently displays AIC for all possible subset models (except the intercept only model) only for $p \leq 10$ in the `PROC REG` procedure in SAS/STAT. For $p > 10$, SAS displays AIC only for a subset of models. This paper shows SAS code that will implement the exact implicit enumeration algorithm by Bao (2005) which guarantees to find the optimal solution when allowed to run to completion. The SAS code presented in this paper uses the SAS System for personal computers version 9.2 running on a Windows XP Professional platform with Service Pack 2.

INFORMATION CRITERIA

Information criteria are measures of goodness of fit or uncertainty for the range of values of the data. In the context of multiple linear regression, an information criterion measures the difference between a given model and the "true" underlying model. Beal (2005, 2007) presents SAS code for using information criteria in variable selection.

Akaike (1973) introduced the concept of information criteria as a tool for optimal model selection. Akaike (1987) and Bozdogan (1987, 2000) discuss further developments of using information criteria for model selection. Akaike's Information Criteria (AIC) is a function of the number of observations *n*, the sum of squared errors (SSE), and the number of independent variables $k \leq p + 1$ where *k* includes the intercept, as shown in Eqn. (1).

$$AIC = n \cdot \ln\left(\frac{SSE}{n}\right) + 2k \qquad (1)$$

The first term in Eqn. (1) is a measure of the model lack of fit, while the second term ($2k$) is a penalty term for additional parameters in the model. Therefore, as the number of independent variables *k* included in the model increases, the lack of fit term decreases while the penalty term increases. Conversely, as variables are dropped from the model, the lack of fit term increases while the penalty term decreases. The model with the smallest AIC is deemed the "best" model since it minimizes the difference from the given model to the "true" model. AIC satisfies the monotonicity requirement for the objective function to be minimized.

IMPLICIT ENUMERATION ALGORITHM

A complete description and derivation of the exact implicit enumeration (IE) algorithm is described in Bao (2005). Therefore, this paper provides a brief overview of how the algorithm is implemented and terms used in Bao (2005).

The seven steps of the exact implicit enumeration algorithm are

1. Initialization,
2. Computing an upper bound,
3. Updating,
4. Computing a lower bound,
5. Fathoming,
6. Branching,
7. Backtracking.

In the initialization step, all variables are assigned to be “free” variables. Free variables are free to enter the model during implementation of the algorithm. Once a variable enters the model, it becomes a “fixed” variable, which means it is fixed in the model solution. The algorithm is exhaustive, as it will either enumerate or eliminate by fathoming every possible solution. In addition, the algorithm is non-redundant, so it will enumerate any solution at most once. AIC is calculated for four models at each node of the enumeration tree:

1. Core solution where all free variables are excluded from the model,
2. Replete solution where all free variables are included in the model,
3. Augmented core solution where the best one free variable is added to the core solution,
4. Diminished replete solution where one free variable is deleted from the replete model.

The smallest AIC among these four models is then compared to the current upper bound AIC of the best model identified so far. If the smallest AIC is less than the current upper bound, then this smallest AIC becomes the new upper bound. If the smallest AIC is greater than or equal to the current upper bound, fathoming occurs on the diminished replete model. Since no models down that branch of the enumeration tree can have a lower AIC than the smallest AIC found at that node, IE does not need to evaluate models further down that branch. Fathoming continues until the model with the global minimum AIC is identified. Optimality is then proven using the monotonicity of the AIC objective function. IE will identify the globally minimum AIC if allowed to run to completion.

SIMULATED DATA

A multivariate data set with 16 independent regressor X variables was simulated from normal, lognormal, exponential, gamma, and uniform distributions with various means and variances. The following SAS code simulates 100 observations for these 16 independent X variables. Variables X5, X6, X7, X8, X13, X15, and X16 are correlated with other variables.

```
data dat;
  do i = 1 to 100;
    X1 = 10 + 3*rannor(0);
    X2 = 20 + 2*ranuni(0);
    X3 = 15 - exp(2*rannor(0));
    X4 = 35 + 2*ranexp(0);
    X5 = 3*x3 + 2*x1 - 2*rangam(0, 2);
    X6 = x1*x3 - x5 + 0.4*rannor(0);
    X7 = 2 + 3*x6 - 2*x1 + ranuni(0);
    X8 = x6 + ranuni(0)*ranexp(0);
    X9 = 10 + ranuni(0);
    X10 = 5 - 3*ranuni(0);
    X11 = 3 + 5*ranexp(0) + rannor(0);
    X12 = 5 + 2*exp(3*rannor(0));
    X13 = 53 + X5*X8 - 3*X10 + 2*ranuni(0);
    X14 = 50 + 7*rannor(0);
    X15 = X14 - 2*X1 + rangam(0, 3);
    X16 = X10 + 2*X3 + 2*rannor(0);
    output; end; drop i; run;
```

For each p ($2 \le p \le 16$) an explicit enumeration algorithm generated all possible 2^p models including the intercept term. One of these 2^p models was randomly selected as the true model so the dependent variable Y could be calculated. The coefficients and sign for each X variable were also randomly assigned for the true model.

SAS CODE FOR IMPLICIT ENUMERATION

The SAS code for implementing the implicit enumeration algorithm follows. PROC IML from SAS/STAT is used to calculate AIC values.

```
%let P = 16;                        ** P = number of independent regressors;
%let N_MODELS = %eval(2**(&P.));  ** assume each model has intercept;
%let INPDS = alldat_p&P;          ** with Y variable calculated;
%let DS_USE = anydsname;
%let MAX_IT = 10000;            ** maximum number of iterations the model will run;
%let LAST_IT = 0;                  ** stores how many iterations IE used;
%let MAX_VARNAME_LENGTH = 8;
%let MIN_VAR = AIC;
%let YVAR  = y;

data &DS_USE.1; set &INPDS;  INT = 1;  ** for intercept;  keep y x1-x&P int; run;
data &DS_USE; set &INPDS;  keep y x1-x&P; run;

******  DEFINE VARIABLES  ********;
data incumbent;  input VARNAME $1-8 VARNUM W FF $20-24 UNDERLINE $27 ITERATION;
  cards;
int          0   1 fixed  N  0
;
run;

%macro obsnvars(ds);  %* stores the number of observations into a macro variable;
   %global dset nvars nobs;
   %let dset=&ds;
   %let dsid = %sysfunc(open(&dset));
   %if &dsid %then %do;
     %let nobs =%sysfunc(attrn(&dsid,NOBS));
     %let rc = %sysfunc(close(&dsid));
     %end;
   %else
   %put Open for data set &dset failed - %sysfunc(sysmsg());
%mend obsnvars;

%macro assignvar;
 data addvars;  length VARNAME $8;
  %do i = 1 %to &P;
   VARNAME = "X&i."; VARNUM = &i.; W = 0; FF = 'free'; UNDERLINE = 'N';
   ITERATION = .; output; %end;  run;

  %do i = 1 %to &P; %global XVAR&i.; %let XVAR&i. = X&i.; %end;

 data incumbent; length VARNAME $8; set incumbent addvars; run;
%mend assignvar;
%assignvar

data current0; set incumbent; run;
data current; set incumbent; run;

%macro ds_names(name, num);  %* resolves macro variables containing X variables;
  %do zz = 1 %to &num;
    &&&name&zz.
  %end;
%mend ds_names;
```

```
 %macro ds_names2(name, num);  %*  this macro is used in SET statements;
   %do zz = 1 %to &num;
     &name&zz.
   %end;
 %mend ds_names2;

 %obsnvars(&DS_USE); %let N_OBS = &nobs; %let N_PAR = &nvars;

 data _null_;   p = &N_PAR - 1;  call symput('N_XPAR', p); run;

%macro calc_ss2(dsname, var, num);
%*  calculates the extra sums of squares (SS2) given other variables in the model;
 data fixed_with1;  set &DSNAME;  where w=1 and varname ^in ('int', "&VAR");
    call symput('FIXED_VAR'||left(_N_), left(trim(varname))); run;

  %obsnvars(fixed_with1); %let N_FIXED = &nobs;

 %if &N_FIXED > 0 %then %do;
  proc reg data=&DS_USE outest=est_fixed_with1 noprint;
    model &YVAR = &VAR %ds_names(FIXED_VAR, &N_FIXED) / selection=none sse aic; run;

  proc reg data=&DS_USE outest=est_fixed_without1 noprint;
    model &YVAR = %ds_names(FIXED_VAR, &N_FIXED) / selection=none sse aic; run;
  %end;
 %else %do;
  proc reg data=&DS_USE outest=est_fixed_with1 noprint;
    model &YVAR = &VAR / selection=rsquare sse aic; run;

  proc reg data=&DS_USE outest=est_fixed_without1 noprint;
    model &YVAR = / selection=none sse aic; run;
  %end;

 data with1; set est_fixed_with1; rename _sse_=SSE_WITH1; keep _sse_; run;
 data without1; set est_fixed_without1; rename _sse_=SSE_WITHOUT1; keep _sse_; run;

 data calc_ss&num;
  length VARNAME $ &MAX_VARNAME_LENGTH;
  set without1;
  if _N_=1 then set with1;
  SS2 = sse_without1 - sse_with1;
  VARNAME = "&VAR";
  VARNUM = &num;
  FF = 'free';
  run;
 %mend calc_ss2;

 %macro calc_ss2_all(dsname);  %*  This macro calculates all SS2 for a given model;

 data free;  set &DSNAME;  where ff='free';
   call symput('FREE_VAR'||left(_N_), left(trim(varname))); run;

   %obsnvars(free); %let N_FREE = &nobs;

 %do i = 1 %to &N_FREE;  %calc_ss2(&DSNAME, &&FREE_VAR&i., &i.);  %end;

 data calc_ss; length VARNAME $8; set %ds_names2(calc_ss, &N_FREE); run;

  %obsnvars(calc_ss); %let N_ROWS = &nobs;

  proc sort data=calc_ss; by ss2;run;

  data calc_ss; set calc_ss; if _N_=1 then SMALLEST_SS2='*';
   if _N_=&N_ROWS then LARGEST_SS2='*'; run;
```

```
 %mend calc_ss2_all;

%macro calc_icomp(dsname);  %*  calculates the AIC for a given model;

data x_without_y;
  set &DSNAME;  ** this assumes the intercept column of 1s is in it;
  drop &YVAR; run;

data y_only; set &DSNAME; keep &YVAR; run;

 proc iml;
 pi = 3.141592654;
 use x_without_y;
 read all var _num_ into x;   * convert data set X_WITHOUT_Y into matrix X in IML;
 close x_without_y;
 use y_only;
 read all var _num_ into y;      * convert data set Y_ONLY into matrix Y in IML;
 close y_only;
 n=nrow(x);                      * number of observations ;
 k=ncol(x);                      * number of variables including the intercept;
 xpx=x`*x;                       * cross-products ;
 xpy=x`*y;
 xpxinv=inv(xpx);                * inverse crossproducts ;
 b=xpxinv*xpy;                   * parameter estimates ;
 yhat=x*b;                       * predicted values ;
 resid=y-yhat;                   * residuals ;
 sse=resid`*resid;               * sum of squared errors ;
 sigma2_hat = sse/n;             * sigma squared hat      ;
 dfe=n-k;                        * degrees of freedom error ;
 mse=sse/dfe;                    * mean squared error ;
 rmse=sqrt(mse);                 * root mean squared error ;
 tr=trace(sigma2_hat#xpxinv);
 d=det(sigma2_hat#xpxinv);
 AIC = n*log(sigma2_hat) + 2*k;
 create icomp var{sse aic mse rmse n k}; * creates SAS data set ICOMP;
 append;
 quit; run;
%mend calc_icomp;

%macro select_xvars(dsn, outpds);
  %*  selects the variables from the full data set to calculate AIC in the model;

 data a; set &DSN; where w=1;
  call symput('N_VAR'||left(_N_), left(trim(varname))); run;

 proc transpose data=a out=at; var w; id varname; run; %obsnvars(a);
  %let N_VARS = &nobs;

 data keepem; set &DS_USE.1; keep &YVAR %ds_names(N_VAR, &N_VARS); run;
  %calc_icomp(keepem);

 data &OUTPDS; set icomp; if _N_=1 then set at; drop _name_; run;

 %mend select_xvars;

 ****  do upper and lower bound initializations  ********;

 %global ITERATION Z_UB Z_LB;
 %let ITERATION = 0;
 %let Z_UB = 1E20;  ** initial upper bound on information criteria;
 %let Z_LB = 0;     ** initial lower bound on information criteria;
 %let IT_Z_UB = &Z_UB;  ** local upper bound on actual model information criteria;
 %let IT_Z_LB = &Z_LB;  ** local lower bound on fathomed information criteria;
```

```
%let OPTIMAL = N;    ** set optimality to N at first;

%macro iterate;  %* this is the main macro that implements IE;
 %do ITERATION = 1 %to &MAX_IT;
   %if &OPTIMAL = N %then %do;

    ***  GET AIC FOR CORE REGRESSION MODEL  ***;

   %calc_ss2_all(current); run;
   %select_xvars(current, z1_core); run;

     ***  GET AIC FOR AUGMENTED CORE REGRESSION MODEL  ***;

 data max_ss; set calc_ss;
   where largest_ss2='*';  ** keep largest SS to add variable to core model;
   keep varname ss2 varnum ff largest_ss2; run;

  proc sort data=max_ss; by varname;
  proc sort data=current; by varname;

data aug_core; merge current max_ss; by varname; if largest_ss2 ='*' then w=1; run;

%select_xvars(aug_core, z2_aug_core);

  ***  GET AIC FOR REPLETE REGRESSION MODEL  ***;

data replete; set current; if ff='free' then w=1; run;

%select_xvars(replete, z3_replete);

  ***  GET AIC FOR DIMINISHED REPLETE REGRESSION MODEL  ***;

%calc_ss2_all(replete);

data min_ss; set calc_ss;
  where smallest_ss2='*';  ** keep smallest SS to drop variable from replete model;
  keep varname ss2 varnum ff smallest_ss2; run;

 proc sort data=min_ss; by varname;
 proc sort data=replete out=replete_sub; by varname;

data dim_replete; merge replete_sub min_ss; by varname;
 if smallest_ss2='*' then w=0; run;

%select_xvars(dim_replete, z4_dim_replete);

data z;
  length MODEL $12;
  set z1_core z2_aug_core z3_replete z4_dim_replete;
  ITERATION = &ITERATION;
  if _N_=1 then MODEL='Core';
    else if _N_=2 then MODEL='Aug Core';
    else if _N_=3 then MODEL='Replete';
    else if _N_=4 then MODEL='Dim Replete'; run;

data aug_core_k; set z; where model='Aug Core'; call symput('AUG_CORE_K', k); run;

data dim_replete_k; set z; where model='Dim Replete';
 call symput('DIM_REPLETE_K', k); run;

data nums; do k = &AUG_CORE_K + 1 to &DIM_REPLETE_K - 1; output;   end; run;

 proc summary data=nums; var k;
```

```
     output out=kminmax(drop=_type_ _freq_) min=MINK max=MAXK;

  proc sort data=z; by &MIN_VAR;
data zminmax; set z; if _N_=1; rename aic=MIN_AIC sse=MIN_SSE mse=MIN_MSE
 rmse=MIN_RMSE; run;

***  ALWAYS FATHOM DIMINISHED REPLETE MODEL ;

data check_ub;
  set zminmax;
  %global NEW_INCUMBENT Z_UB;
  if round(min_&MIN_VAR., 0.0001) < round(&Z_UB, 0.0001) then do;
     call symput('Z_UB', min_&MIN_VAR.);
    NEW_INCUMBENT = 'Y';
    end;
   else NEW_INCUMBENT = 'N';
   call symput('NEW_INCUMBENT', left(trim(new_incumbent))); run;

 ** use this for calculating fathom AIC;

 data min_aic; set z; where model='Dim Replete';
   ** always use diminished replete SSE for calculating fathom AIC and ICOMP;
  rename k=DIM_REPLETE_K sse=DIM_REPLETE_SSE; keep k sse; run;

 %macro new_incumbent;

 %if &NEW_INCUMBENT = Y %then %do;

  proc sort data=z; by &MIN_VAR;
  data best_sofar; set z; if _N_=1; keep int %ds_names(XVAR, &N_XPAR); run;

  proc transpose data=best_sofar out=best_sofart(rename=(_name_=VARNAME)); run;

  data misc; set z; if _N_=1; keep aic icomp n sse mse rmse k; run;

  proc sort data=best_sofart; by varname;
  proc sort data=current out=cur_sub(keep=varname ff w); by varname;

  data best_sofar;
   merge best_sofart cur_sub; by varname;
   if _N_=1 then set misc;
   if col1=. then col1=0;
   if col1=w then ff='fixed';  w=col1;  drop col1; run;
  %end;
 %mend new_incumbent;
 %new_incumbent

 %global Z_UB Z_LB FATHOM;

 data it&ITERATION.;
  length FATHOM $1;
  merge zminmax kminmax min_aic;
  FATHOM_AIC = n*log(dim_replete_sse/n) + 2*mink;
  IT_Z_UB = min_&MIN_VAR.;  ** minimum of 4 actual models;
  IT_Z_LB = min(fathom_&MIN_VAR., min_&MIN_VAR.); ** min of 4 models and fathomed;
  if &Z_UB > it_z_ub then do;
   call symput('Z_UB', min_&MIN_VAR.);
   Z_UB = it_z_ub;
   end;
  else Z_UB = &Z_UB;
 if it_z_lb >= Z_UB then fathom='Y';
   else fathom='N';
 if &Z_LB < fathom_&MIN_VAR. < &Z_UB and fathom = 'N' then do;
```

```
   call symput('Z_LB', fathom_&MIN_VAR.);
   Z_LB = fathom_&MIN_VAR.;
   end;
  else Z_LB = &Z_LB;
  call symput('FATHOM', left(trim(fathom))); run;

 %macro keep_best_model;

 %global MODEL;

 %if &FATHOM = N %then %do;
data best_iteration&ITERATION;
 set it&ITERATION;
 call symput('MODEL', left(trim(model))); run;

  proc sort data=best_sofart out=best_sofart2; by varname;  run;

data current&ITERATION.;  ** keep copy of current before it gets overwritten;
  set current; run;

%calc_ss2_all(replete); * calculate new SS with best model before updating current;
proc sort data=calc_ss out=calc_ss_sub(keep=varname ss2 smallest_ss2 largest_ss2);
    by varname;
%if &ITERATION > 1 %then %do;
 proc sort data=current(drop=smallest_ss2 largest_ss2); by varname;
 %end;
%else %do; proc sort data=current; by varname; %end;
proc sort data=best_sofart2; by varname;

data cur&ITERATION;
%if &ITERATION > 1 %then %do; set current(drop=new_incumbent); %end;
 %else %do; set current; %end;
if _N_=1 then set check_ub(keep=new_incumbent); run;

data current;
  merge cur&ITERATION best_sofart2 calc_ss_sub; by varname;
  %if &MODEL = Dim Replete %then %do;
      if new_incumbent='Y' and col1=. and ff='free' then do;
        ff='fixed'; w=0; ITERATION = &ITERATION; end;
      if new_incumbent='N' and smallest_ss2='*' and ff='free' then do;
         ff='fixed'; w=0; ITERATION = &ITERATION; end;
    %end;
  %if &MODEL = Aug Core %then %do;
      if new_incumbent='Y' and col1^=. and ff='free' then do;
         ff='fixed'; w=1; ITERATION = &ITERATION; end;
      if new_incumbent='N' and largest_ss2='*' and ff='free' then do;
         ff='fixed'; w=1; ITERATION = &ITERATION; end;
    %end;
  %if &MODEL = Replete %then %do;
      if new_incumbent='Y' and col1=. and ff='free' then do;
         ff='fixed'; w=0; ITERATION = &ITERATION; end;
      if new_incumbent='N' and smallest_ss2='*' and ff='free' then do;
         ff='fixed'; w=0; ITERATION = &ITERATION; end;
    %end; run;
 %end;

%else %do;  ***  FATHOM HERE;

data fixed; set current;  where ff='fixed'; run;

  %obsnvars(fixed); %let N_FIXED = &nobs;

 %do i = 1 %to &N_FIXED;
```

```
    %if &i = 1 %then %do;
       proc sort data=current; by descending iteration;%end;
    %else %do;
       proc sort data=current&ITERATION; by descending iteration; %end;

data current&ITERATION;
  %if &i = 1 %then %do; set current; done='N';  %end;
  %else %do; set current&ITERATION; %end;
  if _N_=1 and ff='fixed' and underline='Y' and iteration>0 and done='N' then do;
    underline='N'; ff='free'; ITERATION_KEEP=iteration; iteration=.;
    if w=0 then w=1; else w=0; end; run;

proc sort data=current&ITERATION; by descending iteration;
data current&ITERATION;
 set current&ITERATION;
 if _N_=1 and ff='fixed' and underline='N' and iteration>0 then do;
   ITERATION_KEEP = iteration;
   underline='Y';
   DONE = 'Y';
   if w=0 then w=1; else w=0;
   end; run;
  %end;

data current; set current&ITERATION; run;
  %end;

%mend keep_best_model;
%keep_best_model;

 *** CHECK IF WE HAVE FOUND PROVEN OPTIMALITY ***;

 proc sort data=current0; by varnum;

 data newcur; set current0;
  rename w=CUR0_W ff=CUR0_FF underline=CUR0_UNDERLINE iteration=CUR0_ITERATION;

 data cur_sub; set current; keep varnum varname w ff underline iteration; run;

 proc sort data=cur_sub; by varnum; run;

data chek; merge cur_sub newcur; by varnum;
  if w^=cur0_w or ff^=cur0_ff or underline^=cur0_underline or
     iteration^=cur0_iteration then DIF=1; else DIF=0; run;

 proc summary data=chek; var dif;
     output out=chekout(drop=_type_ _freq_) sum=DIFSUM; run;

 data _null_; set chekout; call symput('DIF', difsum); run;

 %if &DIF = 0 %then %do;  %let OPTIMAL = Y; %let LAST_IT = &ITERATION; %end;
 %end;  * OPTIMAL;
%end;

data showem; set models_evaluated; where y^=.;
  _rmse_ = round(_rmse_, 0.001);
  intercept = round(intercept, 0.001);
  _sse_ = round(_sse_, 0.001);
  _rsq_ = round(_rsq_, 0.001);
  _aic_ = round(_aic_, 0.001);
  %do k = 1 %to &P;
    X&k = round(x&k., 0.001);
    %end;
  keep _rmse_ intercept _sse_ x1-x&P _rsq_ _aic_; run;
```

```
proc sort data=showem out=showem2 nodupkey; by _aic_; run;

%global N_MODELS_RUN;
  %obsnvars(showem2);
  %let N_MODELS_RUN = &nobs;

data _null_;
  %put LAST ITERATION = &LAST_IT  OPTIMAL = &OPTIMAL ITERATION = &ITERATION ; run;

proc print data=showem2;
  title "&N_MODELS_RUN models evaluated out of &N_MODELS models possible";
  var _aic_ _rmse_ _rsq_ _sse_ intercept x1-x&P.; run;

%mend iterate;
%iterate
```

RESULTS FROM THE IMPLICIT ENUMERATION

The results from IE were compared to PROC REG with the `selection=adjrsq` option which evaluates all possible subset models (except for the intercept only model) for $p \leq 10$. Table 1 compares the number of models evaluated by both PROC REG and IE for $2 \leq p \leq 16$. The number of all possible subset models is 2^p. Table 1 shows PROC REG evaluates all possible subset models (excluding the intercept only model) for $p \leq 10$. However, when $p > 10$ PROC REG evaluates a rapidly decreasing percentage of all subset models. On the other hand, IE evaluates all possible subset models (including the intercept only model) for $p \leq 4$ before arriving at the optimal model with the smallest AIC score. For $p > 4$ IE evaluates a generally decreasing percent of all subset models to arrive at the optimal model and proves optimality.

Figure 1 shows the results from Table 1 in a line graph where the number of independent variables p is on the horizontal axis and the percent of all possible subset models evaluated is on the vertical axis for both IE and PROC REG. Figure 1 shows IE evaluates fewer models than PROC REG for $5 \leq p \leq 10$, while PROC REG evaluates fewer models than IE for $11 \leq p \leq 16$. Both PROC REG and IE identified the optimal model with the smallest AIC score. The optimal models were confirmed by calculating AIC for all possible subset models using an explicit enumeration algorithm by Beal (2008).

Table 1. Number of subset models evaluated by IE and PROC REG

Number of Variables	Number of Subset Models	IE Models Evaluated	Percent of Models Evaluated by IE	Models Proc REG Evaluated	Percent of Models Evaluated by REG
2	4	4	100	3	75
3	8	8	100	7	87.5
4	16	16	100	15	93.8
5	32	20	62.5	31	96.9
6	64	40	62.5	63	98.4
7	128	46	35.9	127	99.2
8	256	92	35.9	255	99.6
9	512	280	54.7	511	99.8
10	1024	224	21.9	1023	99.9
11	2048	599	29.2	111	5.4
12	4096	855	20.9	133	3.2
13	8192	583	7.1	157	1.9
14	16,384	349	2.1	183	1.1
15	32,768	3744	11.4	211	0.6
16	65,536	4492	6.9	241	0.4

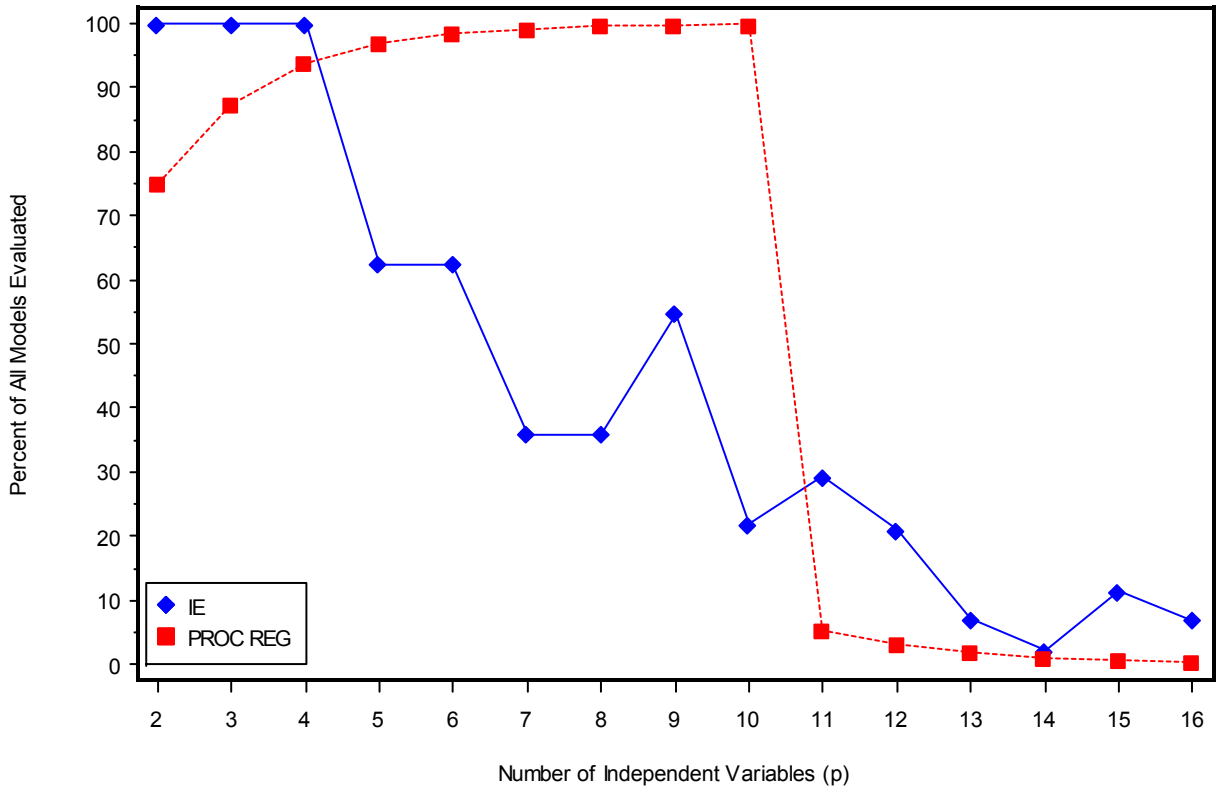

Figure 1. Line graph comparing models evaluated by IE and PROC REG

CONCLUSION

SAS is a powerful tool that utilizes AIC to evaluate all possible subsets of multiple linear regression models (except the intercept only model) to determine the best model for $p \leq 10$ independent variables using PROC REG. However, for $p > 10$ variables, only a small fraction of all possible models are evaluated using the `selection=adjrsq` option when identifying the best subset model with the smallest AIC score. SAS code was presented using Bao's (Bao 2005) exact implicit enumeration (IE) algorithm to determine the global optimal solution that minimizes the monotonic AIC nonlinear objective function. The number of models evaluated using IE was compared with an explicit enumeration of all possible subset models and PROC REG using simulated data. IE identified the global optimal solution and proved optimality in fewer model evaluations compared to PROC REG for $5 \leq p \leq 10$. However, for $p > 10$ PROC REG evaluated fewer models than IE to arrive at the optimal model. Both IE and PROC REG are much more efficient than explicitly enumerating all possible models for large p. PROC REG has the advantage of utilizing less SAS code with the `selection=adjrsq` option, but there is no guarantee the global optimal model will be obtained for $p > 10$. IE requires more SAS code than PROC REG, but optimality can be proven using AIC if IE is allowed to run to completion.

REFERENCES

Akaike, H. 1973. "Information theory and an extension of the maximum likelihood principle". In B.N. Petrov and F. Csaki (Eds.), *Second international symposium on information theory*, 267-281. Budapest: Academiai Kiado.

Akaike, H. 1987. "Factor analysis and AIC". *Psychometrika* 52:317-332.

Bao, X. 2005. "An implicit enumeration algorithm for mining high dimensional data". *International Journal of Operational Research* 1:123-143.

Beal, D. J. 2005. "SAS code to select the best multiple linear regression model for multivariate data using information criteria". *Proceedings of the Thirteenth Annual Conference of the SouthEast SAS Users Group*, Portsmouth, VA.

Beal, D. J. 2007. “Information criteria methods in SAS® for multiple linear regression models”. *Proceedings of the Fifteenth Annual Conference of the SouthEast SAS Users Group*, Hilton Head, SC.

Beal, D. J. 2008. “SAS® Code for Variable Selection in Multiple Linear Regression Models Using Information Criteria Methods with Explicit Enumeration for a Large Number of Independent Regressors,” *Proceedings of the Sixteenth Annual Conference of the SouthEast SAS Users Group*, St. Pete Beach, FL.

Bozdogan, H. 1987. “Model selection and Akaike’s information criterion (AIC): the general theory and its analytical extensions”. *Psychometrika* 52:345-370.

Bozdogan, H. 2000. “Akaike’s information criterion and recent developments in informational complexity”. *Journal of Mathematical Psychology* 44:62-91.

CONTACT INFORMATION

The author welcomes and encourages any questions, corrections, feedback, and remarks. Contact the author at:

Dennis J. Beal, Ph.D.
Senior Statistician / Risk Scientist
Science Applications International Corporation
P.O. Box 2501
151 Lafayette Drive
Oak Ridge, Tennessee 37831
phone: 865-481-8736
fax: 865-481-4757
e-mail: beald@saic.com