

1
2
3
4
5
6
7
8
9
10
11

Data Mining in Space Physics: 1. The MineTool Algorithm

H. Karimabadi¹, T. B. Sipes¹, H. White², M. Marinucci³, A. Dmitriev⁴, J. K. Chao⁴, J. Driscoll¹, and N. Balac⁵

¹SciberQuest, Inc., Solana Beach, CA

²UCSD, San Diego, CA

³Universidad Complutense de Madrid, Madrid, Spain

⁴Institute of Space Science, National Central University, Jung-Li, Taiwan

⁵San Diego Supercomputer Center, La Jolla, CA

12

13 **Abstract**

14

15 A novel data mining method called MineTool is introduced which by virtue of
16 automating the modeling process and model evaluations, makes it more accessible to
17 non-experts. The technique aggregates the various stages of model building into a four-
18 step process consisting of (i) Data segmentation and sampling, (ii) Variable pre-selection
19 and transform generation, (iii) Predictive model estimation and validation, and (iv) Final
20 model testing. Optimal strategies are chosen for each modeling step. However, the
21 modular design of the MineTool enables the substitution of alternative strategies in any
22 of the four modeling steps. A notable feature of the technique is that the final model is
23 always in closed analytical form rather than “black box” form of most other techniques.
24 MineTool can be used for analysis of data (e.g., time series) as well as images. The utility
25 of the technique is illustrated through several examples based on synthetic data.
26 Application of the technique to analysis of spacecraft data will be presented in
27 subsequent papers.

28

29

30 **1. Introduction**

31 Space physics is data centric and relies heavily on the use of spacecraft data for further
32 advances in the field. Here we introduce a new technique in data mining as an aid to the
33 traditional data analysis approach based on visual inspection of data. Data mining is an
34 umbrella term and is used with varied meanings in a wide range of contexts. Here we
35 define data mining as algorithmic approach to data exploration and knowledge discovery.
36 Some of the immediate data mining functionalities of general interest in space physics,
37 applicable to both time series data as well as images/pixels, include event detection and
38 classification.

39 The adoption of data mining techniques in space physics has been slow partly due to the
40 steep learning curve of some of the techniques and/or the requirement to have a working
41 knowledge of statistics. Another factor is the existence of a plethora of data mining
42 approaches and it is often a daunting task for a scientist to determine the appropriate
43 technique. Our goal is to make data mining more accessible to non-experts. Here we
44 present a novel data mining technique, called MineTool, which provides a self-contained
45 step by step procedure for model building. Another key advantage of MineTool is that
46 the solution is always in an analytical form rather than a “black box” solution as in most
47 standard techniques.

48 The results and applications of MineTool to spacecraft data will be reported in
49 subsequent papers. Here we focus on describing the technique itself. The rest of the
50 paper is organized as follows. Data mining algorithms are deeply rooted in mathematics

51 and MineTool is no exception. Section 2 introduces the underlying algorithm of
52 MineTool. This section is by necessity highly mathematical and can be skipped for users
53 less interested in the details of the algorithms. Section 3 illustrates the utility of the
54 technique through several examples. Summary and conclusion are in Section 4.

55

56 **2. MineTool**

57 An important algorithmic issue in data mining is how to find the optimal complexity of
58 the model or the fitting function. Too much complexity in the model can result in overfit
59 whereas not enough complexity can result in underfit . Overfit can happen for example if
60 the algorithm was trained excessively to the point where it adjusts to specific random
61 features of the training data. In the process of overfitting, the performance on the training
62 set increases while the performance on the validation (unseen) data becomes worse.
63 Underfitting occurs when the algorithm performs too much of the generalization and
64 oversimplifies the model to the point where it does not capture the true underlying
65 problem structure. The model is too simplistic/biased to pick up on the important
66 features of the unseen data. As a result the model may work well on small training sets
67 but as the amount of training data increases, its performance suffers because it underfits
68 the data.

69

70 Recently White [2006] described a new family of methods called QuickNet which
71 underlies MineTool. These methods aim directly at balancing the competing dangers of
72 underfit and overfit to identify the level of model complexity that guarantees the best out-

73 of-sample prediction performance without ad-hoc modifications to the fitting algorithms
74 themselves. While QuickNet provides a general method, it leaves open several choices in
75 specific implementation of the various modeling steps. Different specific
76 implementations of QuickNet exist, such as RETINA [Pérez-Amaral et al., 2005] and
77 RETINET [Marinucci, 2006]. MineTool uses QuickNet as the basis, draws from
78 RETINA and RETINET for its specific implementation schemes, and includes additional
79 customization features. The methodology improves the neural network-like modeling
80 approach, and is developed specifically to address the problem of identifying a model
81 architecture with many potential variables while avoiding overfitting. We accomplish
82 this by choosing the predictive model architecture to be linear in the parameters, yet built
83 on possibly highly non-linear transformations of the input.

84

85 **2.1 MineTool Algorithm – the Four-Step Modeling Process**

86

87 From a high-level point of view, the input to the MineTool algorithm is a set of input
88 vectors \mathbf{X}_j , and the output is the vector of predicted target values \mathbf{Y} , created using the
89 MineTool method. Here we denote vectors in bold. As an example, let us consider the
90 problem of identifying FTEs in time series data. In such an example, \mathbf{X}_j could consist of
91 components of magnetic field, plasma density, and plasma velocities and \mathbf{Y} could be a
92 binary tag assuming values of 0 and 1 with 0 indicating that a particular observation i is
93 not a FTE and with 1 indicating that it is a FTE. Table 1 illustrates the dataset that is
94 given as an input to the MineTool algorithm. The data matrix consists of K input
95 variables (\mathbf{X}_j), one output variable (\mathbf{Y}) and N observations or samples. Column \mathbf{Y}

96 represents the output or target values, one for each of the samples described by the input
97 variables X_j . The algorithm searches for a model M that best relates rows of the input
98 variable values X_{ij} to the appropriate target value y_i :

99

$$100 \quad y_i = M(X_{ij}), \quad \text{where } i = 1, \dots, N \text{ and } j = 1, \dots, K \quad (1)$$

101

102 To produce the model M (equation 1) using the MineTool approach, the algorithm
103 follows four major steps, as illustrated in Figure 1. After sampling the data, the algorithm
104 calculates the dependencies and correlations of the input X and the transformations of X ,
105 producing various candidate models which are evaluated, and the best predictive model is
106 chosen, creating an accurate forecast or prediction of the target values Y . Figure 1 also
107 shows step 1b, the detection of anomalous data that is planned to be implemented in the
108 near future, and hence is just suggested (in dashes) in the figure. We now give a more
109 detailed description of each of the modeling steps.

110

111 ***Step 1: Data Segmentation and Sampling.*** To guarantee the greatest success of the
112 models, MineTool requires data to be split at multiple stages in the modeling process so
113 that there are systematic tests of real-world performance throughout. To ensure that this
114 system works reasonably well, a percentage of the data are “held out”, or completely
115 reserved to serve as novel or unseen data. The records are selected into their respective
116 samples at random; the samples used for modeling are as follows:

- 117 • **Training and validation**— Used to estimate and validate candidate model
118 parameters. Used in the cross-validation of modeling results to verify performance of

119 selected variables based on out-of-sample R-squared measures. The validation
120 sample is critical to the model selection and variable transform generation process as
121 it allows measurement of forecast performance out of sample. This set is further
122 divided into three sub samples, each used for a different purpose, producing six
123 ordering combinations (Figure 3) that are varied in the algorithm to further increase
124 the accuracy, as described bellow.

- 125 • **Hold-out**—Held entirely outside the estimation and validation process, these data
126 are used to give the model some real-world exercise.

127

128 ***Step 1b: Anomalous Data Detection.*** This step is needed to identify the data that might
129 have unusual influence on the model estimation routines, and therefore be potentially
130 detrimental for the model accuracy. After sampling, the data will be processed by the
131 outlier detector, a form of a clustering algorithm that allows multivariate outliers to be
132 identified among the data. An outlier is identified as a record that is distant from its k
133 nearest neighbors and which therefore lies in a region of low probability density. For
134 classification problems the data are separated between the target=0 sample and the
135 target=1 sample so that outliers can be identified relative to these separate groups.

136

137 ***Step 2: Variable Pre-Selection and Transform Generation.*** The final task before
138 beginning the model fitting process is the variable transform generation and a preliminary
139 elimination of non-predictive variables to reduce model complexity. The transforms
140 generated at this stage include the following for all of the variables on the input dataset:

141 • **Level one transforms**—Univariate continuous variables are grouped into cross-
142 products, squares, inversions and divisions. The level one transformations are
143 defined as:

144 •
$$\zeta(\mathbf{X}_i) = \{ X_{i1}^a X_{i1}^b, \dots, X_{ih}^a X_{ih}^b \}, \text{ where } a, b = -1, 0, 1 \text{ and } h, l = 1, \dots, K$$

145
146 giving a total of $1 + 2K^2 + 2K$ variable transforms of the original K inputs. Please
147 note that the original variables are a part of level one transforms (e.g. $\mathbf{X}_1^1 \mathbf{X}_2^0 = X_1$).

148

149 • **Level two transforms**—These are the interaction level transforms of the level one
150 transforms, allowing four-term transformations, such as $\mathbf{X}_1^2 \mathbf{X}_2^2$ or \mathbf{X}_3^4 . The
151 algorithm takes care of the redundancies, such as $(\mathbf{X}_1^2)(\mathbf{X}_1^{-2}) = 1$, and divisions by
152 zero.

153

154 • **User defined transforms**—User can include additional transforms.

155

156 Next, all of the variables generated up to this point are tested for performance on the
157 target variable. The main statistic for evaluating performance of the transforms $\zeta(\mathbf{X}_i)$ is
158 the univariate R-squared score. To measure the R-squared, we use the square of the
159 linear correlation coefficient, or Pearson's R: $R\text{-squared} = [\text{cov}(Y, X) / (\text{var}(Y) \text{var}(X))]^2$,
160 where $\text{cov}(X, Y)$ is the covariance of X and Y , and $\text{var}(X)$ is the variance of X . The R-
161 squared score is used to order the transformations according to their correlation to the
162 output Y .

163 **Step 3: Predictive Model Estimation and Validation.** As with most pattern recognition
164 problems [e.g. Ripley 1996], there are far more potential candidate variables for
165 inclusion, even after selection for univariate predictive power, than can actually be
166 accommodated in a predictive model. There are several significant risks in this situation.
167 One is that potentially useful candidates get overlooked simply because there are too
168 many variables to evaluate. Another is that if a systematic routine for evaluating and
169 including variables is used, it can lead to overfit. Finally, many candidate variables are
170 likely to be redundant, which can cause difficulties for the estimation routines.

171 MineTool deals with all of these issues by combining the theoretical nonlinear curve
172 fitting capability of the typical ANN with the stability of hierarchical techniques. The
173 predictive modeling phase of the algorithm consists of the following basic steps:

- 174 1 - Variable testing for inclusion into a candidate model
- 175 2 - Model estimation
- 176 3 - Non-linear input variable addition, if needed

177

178 **Step 4: Final Model Testing.** This step is used to ensure that whichever model is
179 selected as the final model, it is truly better than a simpler benchmark (linear regression
180 model created using the original variables) or other candidate models, on unseen data.
181 We use a holdout dataset for this purpose set aside specifically to test the final model(s).
182 We perform cross-validated testing (CVMSE) of the final model on this unseen data, to
183 get the estimate of the model's performance on novel data.

184

185 **2.2 Putting It All Together**

186 **2.2.1 Specifics of the MineTool Method**

187 MineTool creates a predictive model architecture that is linear in the parameters.
188 The model parameters are either linear combinations of the input, linear transformations
189 of the input variables ($\zeta(\mathbf{X}_i)$), or highly non-linear transformations of the input ($\Psi(\mathbf{X}_i, \gamma)$).
190 Equation 2 describes the general form of a MineTool model:

191
$$y_i = \mathbf{X}_i' \boldsymbol{\alpha} + \sum_{p=1}^P \zeta(\mathbf{X}_i)' \boldsymbol{\delta}_p + \sum_{q=1}^Q \Psi(\mathbf{X}_i, \gamma_q)' \boldsymbol{\beta}_q \quad (2)$$

192

193 Here prime indicates transpose of the vector, index i refers to the i^{th} observation, P
194 is the number of linear transformations (which does not include the original variables as
195 they are already included in the first part of the Eq. 2), and Q is the number of
196 neurons/hidden layers. $\boldsymbol{\beta}$ is a vector coefficient of the nonlinear transformations and is to
197 be determined from the modeling process and Ψ is a given activation function. In its
198 simplest form, the model would be a linear combination of the input parameters (i.e. a
199 linear regression model). MineTool goes beyond a simple linear model by introducing
200 the linear (such as level-1 and level-2 transformations producing cross-products, ratios,
201 squares, cubes etc.) and non-linear transformation of the input variables, if their addition
202 increases the model accuracy. The non-linear transforms Ψ are single hidden layer
203 feedforward Artificial Neural Net (ANN)-like transforms, just like the ANNs of the same
204 architecture, with the difference that the non-linear transformed inputs are combined into
205 a linear model (i.e., added, instead of combined by a non-linear function μ).

206

207 Figure 2 graphically illustrates the architecture of a MineTool model, as defined
208 in Eq. 2. As depicted in the figure, we see that the power of a MineTool method comes
209 from the fact that the model it produces is linear in the parameters, allowing for highly
210 non-linear parameters, only if needed. In the next sections we discuss the
211 implementation details of the MineTool method.

212

213 ***2.2.2 Candidate Model Building and Testing***

214

215 In this section we elaborate on the candidate model building and the model testing and
216 selection processes (Steps 3 and 4 in Figure 1). We illustrate the details of the model
217 building and testing steps in Figure 3. In the MineTool methodology, after the linear
218 input transforms are created, they (the original inputs are a part of them) are considered
219 for inclusion into the current candidate model architecture. To improve accuracy,
220 MineTool accomplishes this by dividing the training data into three sub-samples (S_1 , S_2
221 and S_3). The three sub-samples are ordered into six possible orderings ($S_1S_2S_3$, $S_1S_3S_2$, ...
222 , $S_3S_2S_1$). For each sub-sample combination or ordering, we perform the following:

223

- Using the data from the first sub-sample S_1 :

224

- Order the candidate transforms $\zeta(\mathbf{X})$, start with the one most highly
225 correlated to \mathbf{Y} : This step produces an ordered list of $\zeta(\mathbf{X})$'s, with the
226 $\zeta_h(\mathbf{X})$ most correlated to the output at the top. For example, we would
227 have a list that looks like: \mathbf{X}_3^4 , $\mathbf{X}_3^2\mathbf{X}_2^2$, $\mathbf{X}_3/\mathbf{X}_1^2$, $1/\mathbf{X}_3^2$, \mathbf{X}_3 , ...

- 228 ○ Start building the candidate model using the first transform (i.e., the one
229 most highly correlated with Y) from the list (in our example that would
230 be \mathbf{X}_3^4).
- 231 ○ Consider other variables from the list if they satisfy the colinearity
232 threshold λ : In our example, we would start with $\mathbf{X}_3^2\mathbf{X}_2^2$ and check
233 whether it should be included in the candidate model. We only include a
234 transform $\zeta_i(\mathbf{X})$ if the R-squared of the regression of $\zeta_i(\mathbf{X})$ already
235 included in the model is less than or equal of λ . We calculate the R-
236 squared of $\zeta_i(\mathbf{X})$ by, in our example, inputting \mathbf{X}_3^4 into a linear regression
237 model and predicting $\mathbf{X}_3^2\mathbf{X}_2^2$. We then calculate the R-squared of the
238 actual and the predicted value of $\mathbf{X}_3^2\mathbf{X}_2^2$ and if it satisfies the λ criterion,
239 it gets included into the model. For the sake of our example, let's say
240 that \mathbf{X}_3^4 and $\mathbf{X}_3^2\mathbf{X}_2^2$ have both been selected for the model inclusion, and
241 we are proceeding with deciding whether to include $\mathbf{X}_3/\mathbf{X}_1^2$ (the next
242 variable transform from our list). We decide whether to include $\mathbf{X}_3/\mathbf{X}_1^2$
243 by inputting \mathbf{X}_3^4 and $\mathbf{X}_3^2\mathbf{X}_2^2$ into a linear regression model and predicting
244 $\mathbf{X}_3/\mathbf{X}_1^2$. We then calculate the R-squared of the actual and the predicted
245 value of $\mathbf{X}_3/\mathbf{X}_1^2$ and if it satisfies the λ criterion, it gets included into the
246 model. This is repeated for all the variable transformations on the sorted
247 list of variable transforms, and we end up with the transforms to be
248 included into the current candidate model (for the given subset ordering,
249 and the given value of λ).

250 ○ We then increment λ and repeat the process of the candidate model
251 creation, for each of the values of λ (by default, $\lambda = 0.1, 0.2, \dots, 1.0$). For
252 each value of λ we end up with a different set of variables and a different
253 model. For the default value range of λ , we end up with 10 different
254 models, one for each increment of λ .

255

256 • Using the data from sub-samples S_1 and S_2 :

257 ○ Estimate each candidate model consisting of the variables chosen in the
258 previous step, using the data from S_1 only, and computing the CVMSE

259 on S_2 where $CVMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (E_i^{S_2})^2}$, where N is the number of

260 instances, and $E_k^{S_2}$ is the modeling error of the instance k when the latter
261 is taken from the validation subset S_2 . This is done by inputting all the
262 chosen transforms into a linear regression model and predicting the
263 output Y. We then compute the CVMSE of the predicted value of Y and
264 the actual value of Y, on the sub sample S_2 .

265 • Using the data from sub-samples S_2 and S_3 :

266 ○ Finally, we perform an evaluation of all the chosen candidate models
267 (one for each λ) out-of-sample, using the data from S_3 .

268

269 This process is repeated six times, each time using a different ordering of the sub-
270 samples. To summarize the models, and possibly find an even more accurate model, we
271 follow this step by a creation of the union model [Perez-Amaral et al., 2003]. The union
272 model takes all the variable transforms selected by the previous six steps and creates a
273 new (union) model, which is then evaluated out-of-sample. In the final step, we perform
274 the evaluation of all the seven models (six models created from the different iterations
275 through the sub-sample ordering, and the seventh, union model) and produce the best
276 predictive model, offering the highest accuracy (as measured by the CVMSE) on the
277 hold-out data.

278 The next step is to check the adequacy of the obtained evaluation results, and if
279 higher scores are expected or needed, we proceed with the generation of the non-linear
280 variable transforms ($\Psi(\mathbf{X}, \gamma)$), as described in the following section and illustrated in
281 Figure 6. Once the non-linear transforms are created, they are merged with the linear
282 transformations, and processed using the sub sampling technique we just described in this
283 section (Figure 5). The final model consequently contains both linear (original inputs \mathbf{X}
284 and the linear transformations $\zeta(\mathbf{X})$) and non-linear terms ($\Psi(\mathbf{X}, \gamma)$), and often offers a
285 higher accuracy than just the linear-term model.

286

287 ***2.2.3 Non-Linear Variable Transform Generation***

288

289 As depicted in Figure 3, if the evaluation scores of the model containing only the
290 linear transforms of the input variables are not as high as the user expected, or the user
291 wishes to check the model accuracy when the non-linear terms are used, MineTool

292 proceeds to building and then adding the non-linear transformations to the model (Fig. 4)
 293 which may lead to increase in accuracy. Once the non-linear transformations $\Psi(\mathbf{X}, \gamma)$ of
 294 the input are created, they are merged with the linear transforms selected in the previous
 295 steps of the algorithm, and a new set of models with the combined linear and non-linear
 296 parameters is created. The process to create this mixture model is identical to the process
 297 described in the previous section, and the result is six predictive models and one union
 298 model to be evaluated on the hold-out data.

299 The goal of introducing a number of nonlinear transformations is to improve the
 300 approximation to our target variable Y . Broadly speaking, the procedure is iterative and
 301 besides some user defined parameters, it needs an estimated error vector as a starting
 302 point. In our case this is obtained by the previous step as $\varepsilon = Y - M(0)$ where

303 $M(0) = \mathbf{X}_1' \boldsymbol{\alpha} + \sum_{p=1}^P \zeta(\mathbf{X}_i)' \boldsymbol{\delta}_p$ is the non-ANN part of the model involving selected level 1

304 and 2 transforms of the original X data matrix. In the first iteration a user established
 305 number c of non-linear transforms is selected such that $M(1) = M(0) + \Psi(\mathbf{X}_{s(I)} \gamma) \hat{\boldsymbol{\beta}}$
 306 where γ is a suitable parameters vector chosen randomly by exploiting Stinchcombe and
 307 White's (1998) results on model misspecification. Here $\mathbf{X}_{s(I)}$ is a matrix which does
 308 include the same number of observations of \mathbf{X} but a *subset* of original inputs (why this is
 309 not the complete data matrix \mathbf{X} will be made clear further). The terms $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$ are
 310 estimated by OLS while keeping γ fixed. Finally a new error
 311 vector $\hat{\varepsilon} = Y - M(0) - \Psi(\mathbf{X}_{s(I)} \gamma) \hat{\boldsymbol{\beta}}$ is computed. This process is repeated until a user-
 312 defined maximum number of iterations I is achieved, resulting in $Q = c * I$ non-linear terms
 313 added to the initial $M(0)$ model which is then processed through the main loop in Figure 5

314 for the final model determination. Non-linear estimations procedures are avoided since
315 the parameter vectors γ of the hidden units are generated randomly.

316 Although the algorithm presented here has Quicknet [White, 2006] as its core, it
317 does differ in its detail in several important respects. For example, here we use a subset
318 $X_s \subseteq X$ of the inputs to build the hidden units instead of the all the available X . We also
319 allow the users to choose among a mix of three different types Ψ of squashing functions,
320 namely Logistic, Radial Basis and Ridgelets ANN. The user also has the option to add
321 other powerful approximation methods that are special cases of ANN such as Fourier
322 Transforms, Bernstein, Hermite or Chebycheff polynomials, among others. By allowing
323 the algorithm to choose among several squashing functions, we typically obtain more
324 accurate results.

325 To create the non-linear transformations Ψ of the input variables X , we use a
326 simple, yet very effective iterative process (as illustrated in Figure 4). At each iteration (i
327 $= 1, \dots, I$), the method randomly selects a subset of the input variables X , creates $m = 100$
328 potential non-linear transformations with the use of a different basis function (such as
329 logistic, radial basis and ridgelets), and selects the $c = 5$ most correlated non-linear
330 transforms Ψ with the residuals (or, the error) of the model created in the previous step.
331 At the beginning or step 0 of the process, model $M(0)$ is the best model created using just
332 the linear transforms of the input. The error is the difference between the actual output Y
333 and the predicted value given by $M(0)$. The goal of including the non-linear transforms is
334 to add new parameters to the model, to fill in the difference between the actual output
335 value Y and the output value given by the best model created by that point in time. This
336 iterative process is repeated until we reach the predetermined number of $Q = 50$ non-

337 linear potential terms to be added to the model, in $I = 10$ increments of $c = 5$. At each of
338 the iteration steps, the new model is created by adding the non-linear terms to the existing
339 linear terms, and entering all of the transformations into a linear regression model. This
340 produces a new set of coefficients (both for the linear and non-linear parameters) and a
341 new predictive model, slightly more accurate than the model in the previous step, $M(i-1)$,
342 as displayed in Figure 4. Once the top $Q = 50$ non-linear terms are selected as potential
343 terms to be added to the model, they are merged with the best linear terms (as illustrated
344 in Figure 3), where the best overall linear and non-linear transforms are selected (i.e. in
345 general less than 50 non-linear terms are chosen) to produce the best possible model, as
346 tested on the out-of-sample data.

347

348 ***2.2.3.1. Input Selection Probabilities***

349

350 The reader may wonder how step 1 (where the input variables are used to build the ANN
351 transforms) is practically implemented since at the i -th iteration we select just a subset $s \leq$
352 K of inputs to produce m ANN candidate transforms. Indeed we could use all K inputs to
353 produce these transforms, simply setting $s=K$. We prefer the subset approach which is
354 motivated by the fact that K may be potentially large. It is more convenient to use few
355 inputs to produce each non-linear transform, since these will be easier to interpret in the
356 analytical terms.

357 In order to produce m ANN candidate transforms we have to select m possible
358 *input sets*. In turn each *input set* may include a different number of inputs. To make clear
359 this point consider a case in which we have only three inputs. Define a vector $I=\{a,b,c\}$ of

360 indexes corresponding to the inputs $\{X_a, X_b, X_c\}$. Then, all possible *input sets* of different
 361 size $s=1, s=2, s=3$ are given by all possible combinations of three elements on $s=1, s=2,$
 362 $s=3$ positions, that is: $S=\{(a),(b),(c),(a,b),(b,c),(a,c),(a,b,c)\}$. Our objective is then to
 363 generate m of such *input sets* with replacement at each iteration by randomly extracting
 364 them from S . Nonetheless we want to assign a higher selection probability on those *inputs*
 365 *sets* which have a small number of elements. In terms of our previous example this means
 366 that we would like:

367

$$368 \quad Pr\{a\}=Pr\{b\}=Pr\{c\} > \quad Pr\{a,b\}=Pr\{b,c\}=Pr\{a,c\} > \quad Pr\{a,b,c\}.$$

369

370 A possible strategy to satisfy the above conditions is to adopt a weighting scheme on
 371 each element of the set S which is proportional to the inverse of the size s of the *input set*.
 372 Following with the above example, we would have the following result as shown in
 373 Table 2.

374

375 Thus, by simply normalizing the selection factors such that their sum is 1, we obtain the
 376 necessary weights in order to favor the selection of small-sized *input sets* over bigger-
 377 sized *input sets*.

378

379 **2.2.3.2. Construction of the γ_m weight vectors**

380

381 The proper choice of the weight vectors is relevant in order to generate a plausibly useful
 382 collection of candidate nonlinear predictors. First we must avoid the generation of

383 candidate ANN transforms that are collinear with previously included predictors. By the
384 same token predictors that are approximately constant or have reduced range of variation
385 should also be avoided.

386 In order to ensure these desirable properties, we first generate the hidden units by
387 scaling adequately and selecting randomly the elements of the weights vector γ_m such that
388 the magnitudes of the coefficients (usually position and the scale parameters) are
389 comparable and independent of each other. As an example take the logistic squashing
390 function $\Psi(\gamma_0 + \gamma_1 X)$ with a single predictor X having mean zero and parameter vector
391 $\gamma = (\gamma_0, \gamma_1)$. If γ_0 (the scale parameter) is chosen too large in absolute value compared to γ_1
392 X , then $\Psi(\gamma_0 + \gamma_1 X)$ behaves approximately as $\Psi(\gamma_0)$, which means that it will be
393 roughly a constant. If γ_1 (the scale parameter) is chosen small relative to the standard
394 deviation of X , then $\Psi(\gamma_0 + \gamma_1 X)$ will vary proportionally as $\gamma_0 + \gamma_1 X$, and therefore be
395 collinear with respect to $\Psi(\gamma_0 + \gamma_1 X)$. To avoid such problems and let $\Psi(\gamma_0 + \gamma_1 X)$
396 behave as a nonlinear function of X , it is thus recommendable to scale γ_0, γ_1 adequately
397 and to choose them independently. Independence will be warranted by choosing γ_0, γ_1
398 randomly. This ensures that correlation among predictors is reduced which is also
399 enforced by further standardization of the generated hidden units. Standardization is
400 beneficial in the fact that it reduces potential numerical problems that may arise during
401 matrix inversions during OLS estimations if the magnitudes of the variances of the
402 predictors vary greatly. These considerations are general and hold in the multivariate
403 case, as well as for different activation functions besides the Logistic Activation function
404 (in our case Ridgelets and Radial Basis Functions).

405

406 **2.2.3.3. Basis functions**

407

408 The logistic function is probably one of the most popular basis functions in the ANN
 409 literature and is given by:

$$410 \quad \Psi_j(x) = \frac{1}{1 + \exp[-(x' \gamma_{1j} + \gamma_{0j})]} \quad (3)$$

411 where $\gamma_{1j} \in \mathbb{R}^k$ and γ_{0j} is scalar. There is a wide range of choices available for the
 412 definition of the basis functions Ψ . Given that our primary objective is to obtain as good
 413 an approximation to Y as possible, we consider two other powerful approximation
 414 methods, Radial Basis Functions [Powell, 1987; Lendasse et al. 2003] and Ridgelets
 415 [Candes 1998]. Ridgelets are defined as:

416

$$417 \quad \Psi_j(x) = \frac{1}{\sqrt{\gamma_{1j}}} \psi\left(\frac{x' \gamma_{2j} - \gamma_{0j}}{\gamma_{1j}}\right) \quad (4)$$

418 where $\gamma_{2j} \in \mathbb{R}^k$, γ_{0j}, γ_{1j} are scalars and ψ is admissible, namely:

$$419 \quad \int z^h \psi(z) dz = 0 \quad (5)$$

420

421 with $h=0, \dots, d/2-1$. We are motivated to use Ridgelets because they turn out to be optimal
 422 for representing otherwise smooth multivariate functions that may exhibit linear
 423 singularities. Finally, Radial Basis Functions (RBF) are linear combinations of
 424 multivariate densities, accommodating a mixture of densities as a special case:

$$425 \quad \Psi_j(x) = \exp\left[-\frac{1}{2}(x - \gamma_{1j})' \gamma_{2j} (x - \gamma_{1j})\right] \quad (6)$$

426

427 where γ_{1j} is a $(n \times k)$ centering matrix and γ_{2j} is a $(k \times k)$ suitable inverse of a given
428 covariance matrix.

429

430 ***2.2.4 The MineTool Method Input Parameters***

431 One of the main advantages of MineTool over standard data mining methods is its ability
432 to automatically analyze data and build predictive models. However, there are a few
433 input parameters, normally set to default values, which could be changed (in case the user
434 chooses to experiment with different values). The method input parameters are:

- 435 • redundancy threshold $\lambda = 0.1, 0.2, \dots, 1.0$
- 436 • number of candidate non-linear transforms (per basis function) $m = 100$
- 437 • total number of added non-linear transforms (single hidden-layer
438 feedforward units) $Q = 50$ (in increments of $c = 5$)
- 439 • non-linear transformations: Logistic, Ridgelets and Radial Basis Function

440 The redundancy (or colinearity) threshold λ controls which available transformations are
441 included into the candidate model. Higher values of λ mean more relaxed redundancy
442 checks for variable inclusion. One might choose the λ increments to be higher than 0.1 in
443 order to speed up the model building process (at the cost of potentially missing a few
444 possibly excellent candidate models in the process). The other input parameters are used
445 only for nonlinear modeling. The integer m defines the number of non-linear
446 transformations that are created per each basis function, to be further evaluated for the
447 inclusion into the model. We find that $m = 100$ works reasonably well and offers a nice

448 tradeoff between finding good non-linear transforms in a reasonable amount of
449 processing time. Moreover, we add them in increments of $c = 5$ at the time, which seems
450 to work really well for the datasets we have analyzed so far. We also find that using the
451 logistic function, ridgelets and the radial basis function helps solve a variety of problems,
452 when used as the basis functions for the non-linear variable transforms creation. Other
453 functions could be used as well.

454

455 **3. Description of the Test Problem**

456 We are currently applying MineTool to a number of projects using spacecraft data.
457 These include development of a 3D model of magnetopause, identification of flux
458 transfer events and traveling compression regions, among others. We will report on these
459 studies elsewhere. Here our goal is to benchmark the algorithm under controlled settings.
460 Given that the models derived from MineTool are in analytical form, we consider two
461 test problems where the test data is generated from analytical equations. We then apply
462 MineTool to the test data and determine how well we can recover the original equation.

463

464 **3.1 Test Problem 1**

465

466 Here, we demonstrate Mintool's performance using a data set generated with an
467 exponential function, and show that (i) Minetool can recover the original function when
468 exponential functions are included in the library of transformations, and (ii) that if we

469 exclude exponentials from its list of transformations, MineTool recovers the first few
470 terms of a Taylor expansion of an exponential.

471

472 We generated a data set consisting of 10,000 observations using random input variables
473 on the following intervals:

474

475 $X_1 = [-2,2]$

476 $X_2 = [-2,2]$

477 $X_3 = [-2,2]$

478

479 and the goal variable was generated as:

480

481
$$Y = 3 + 5X_1 + \exp(X_3) + noise \quad (7)$$

482

483 where the noise is Gaussian distributed with sigma of 1. Note that Y has no dependence
484 on X_2 and we were interested in checking whether the variable selection procedure of
485 MineTool would discard X_2 from the model. Each of the 10,000 lines contained a set of
486 X_1, X_2, X_3 , and a goal variable Y generated from those numbers.

487

488 **Without an exponential fit function.** The Taylor expansion of an exponential function
489 is particularly simple: $\exp(x) \sim 1 + x + 1/2x^2 + \dots$

490 In the default mode, Minetool does not generate exponential transforms. In this case,

491 MineTool's best fit model is:

492

493
$$Y = 3.95 + 4.98X_1 + 1.44X_3 + 0.658X_3^2 \quad (8)$$

494

495 This is very close to the original equation (7), with the exponential replaced by the first
496 three terms of the Taylor expansion. Notice that the coefficients are not exactly what
497 would be calculated from the expansion; this is due the fact that Minetool calculates the
498 best multivariate fit using all the available input data, whereas a Taylor expansion is
499 constructed so that it is most accurate around $X=0$. Also note that the spurious variable
500 X_2 is absent from the model, providing confidence in the variable selection algorithm of
501 MineTool.

502

503 **With exponential transforms of input data.** Next, we added exponential transforms to
504 the list of transformations that MineTool would consider in the model building process.
505 The best fit model created by Minetool (as measure by mean average error) is:

506

507
$$Y = 3.03 + 4.99X_1 + 0.99\exp(X_3) \quad (9)$$

508

509 Comparing equations (7) and (9), we observe that MineTool was able to derive the
510 original equation to a very high accuracy from the data. In this example, the λ feature of
511 the modeling process which eliminates redundancy in the model was critical to obtaining
512 the correct model. The transformed variable $\exp(X_3)$ is highly correlated with Y (since it
513 looks somewhat like a straight line, or the variable X_1), but is excluded from the fit model

514 because $\exp(X_1)$ is highly correlated with the variable X_1 , which is already included in
515 the model.

516

517 **3.2 Test Problem 2**

518 Next, we consider a case where the original equation is much more complex. Our
519 starting point is an empirically derived model of magnetopause [Shue et al., 1997] which
520 was obtained by using least square fit to a pre-defined functional form using spacecraft
521 data:

$$522 \quad R = R_o \left(\frac{2}{1 + \cos \theta} \right)^\alpha \quad \text{where } \alpha = (0.58 - 0.01B_z)(1 + 0.01D_p), \quad (10a)$$

523

$$524 \quad R_o = (11.4 + 0.013B_z)(D_p)^{-\frac{1}{6.6}} \quad \text{for } B_z \geq 0 \quad (10b)$$

525

$$526 \quad R_o = (11.4 + 0.14B_z)(D_p)^{-\frac{1}{6.6}} \quad \text{for } B_z < 0 \quad (10c)$$

527

528 Here R and θ are polar coordinates representing the position of the magnetopause, and B_z
529 and D_p are the z-component of the interplanetary magnetic field (IMF) and solar wind
530 dynamic pressure, respectively. This model has a complex dependence on B_z and D_p
531 including a change in the functional form as a function of sign of B_z , and thus makes it an
532 ideal choice for our test.

533

534 We find it convenient to work with the following variables: $\log(R)$, $\cos(\theta)$, $\ln(D_p)$, and B_z .
535 We then generate a data set of $\log(R)$ for a range of values in $\cos(\theta)$, $\ln(D_p)$, and B_z . The
536 idea is to use the resulting data set and derive a model for $\log(R)=f(\cos(\theta), \ln(D_p), B_z)$
537 which can then be compared against the original equation (10).

538

539 We consider two cases: (i) noiseless case and (ii) noisy case where:

540

$$541 \quad \log(R')=\log(R) + \sigma * \text{random} \quad (11)$$

542

543 with R' representing the magnetopause distance with noise. Here R is from Eq. (10a),
544 σ is the level of noise, and random is a random value from $-\infty$ to $+\infty$, which has a
545 normal statistical distribution with a mean of zero and a standard deviation of one. The
546 level of noise is regulated by the dispersion σ . Because random can reach very large
547 positive and negative values, we introduce upper and lower limits for R' . The upper limit
548 R'_{max} is equal to $20 R_E$ and the lower $R'_{min}=4 R_E$. Here R_E is the earth's radius. The
549 limitation is also useful for realistic representation of the orbital bias in experimental
550 measurements, which are usually restricted by a satellite perigee and apogee.

551

552 ***Binning Scheme***

553

554 We used the above scheme to generate a data set of R consisting of 10800 data points for
555 various values of θ , B_z and D_p . We binned the data set to reflect the proper statistics of
556 solar wind conditions. The solar wind pressure D_p has a log-normal occurrence

557 probability distribution with maximum ~ 2 nPa. Hence we chose 30 bins of D_p in a range
 558 from at least 1 nPa to 50 nPa according to: $\exp\{\ln(1)+(i-1)*[\ln(50)-\ln(1)]/29\}$, where
 559 $i=1\dots 30$. Because the dependence of R on B_z is approximately linear, we used nine bins
 560 for B_z with 5 nT step in the range of -20 to 20. It can be shown that a uniform coverage
 561 of magnetopause surface requires a θ binning of the form:

562

563
$$\theta_{i+1}=\theta_i+\Delta_{i+1}(\theta), \text{ and } \Delta_i(\theta) = const * \left(\frac{2}{1+\cos\theta}\right)^{1.2} \sin\theta \left\{ \cos\theta + \frac{0.6 * \sin^2\theta}{1+\cos\theta} \right\}. \quad (12)$$

564 **3.2.1 Results**

565 In this section we compare and contrast the results, including the resulting analytical
 566 equations, from various algorithms. We use three different error measures, root mean
 567 squared error (RMSE), mean absolute error (MAE), and mean relative error (MRE), to
 568 compare performance across models:

569
$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (predicted - actual)^2}$$

570

571
$$MAE = \frac{1}{n} \sum_{i=1}^n | predicted_i - actual_i | \quad (13)$$

572

573
$$MRE = \frac{1}{n} \sum_{i=1}^n \frac{predicted_i - actual_i}{actual_i}$$

574 Here n is the number of data samples.

575

576 3.2.2 Noiseless Case

577 To enhance numerical accuracy, MineTool works with standardized variables denoted by
578 x_s which are related to the usual variables x by: $x_s = (x - \mu) / \sigma$, where μ is the mean and
579 σ is the standard deviation. Note that $\cos_s(\theta)$ refers to the normalized version of
580 $\cos(\theta)$ rather than *cosine* of normalized θ . Here we chose the benchmark model based
581 on linear regression of the three variables $\cos_s(\theta)$, $\ln_s(D_p)$, and B_{zs} (but in general any
582 model can serve as a benchmark):

$$583 \quad \log(R) = -0.0632268 \cos_s(\theta) - 0.0841309 \ln_s(D_p) + 0.024491 B_{zs} + 0.9859123 \quad (14)$$

584

585 or expressed in terms of non-normalized variables:

586

$$587 \quad \log(R) = -0.2127 \cos(\theta) - 0.0577 \ln(D_p) + 0.001946 B_z + 1.204 \quad (15)$$

588

589 Note that this equation is very similar to that obtained from Pace Regression Eq. (A.3).

590

591 The best candidate model in MineTool based on linear transformations consists of 43
592 explanatory variables including one constant. The 10 leading terms are listed in Table 3.

593

594

595 Next we examined the possibility of obtaining more accurate results by including
 596 nonlinear functions (neural nets) in the model as in Eq.(2). Note that MineTool still
 597 produces an analytical model even in case of neural nets. The neural net model we tried
 598 had 50 candidate hidden units. The resulting analytical model has the form:

$$599 \quad \log(R) = \sum_{i=1}^{33} G_i + \sum_{j=1}^{45} \Psi(\text{hiddenlayer}_j) + \text{constant} \quad (16)$$

600 where G_i consists of the nonlinear combination of variables, and the $\Psi(z)$ includes
 601 ridgelets and radial basis functions. Note that the nonlinear transformation stage results
 602 in 50 terms in $\Psi(z)$ but the final modeling process (Fig. 3) often results in reduction in the
 603 number of needed terms. In this case, the number of hidden layers kept has been reduced
 604 to 45. The summation involving $\Psi(z)$ is the ANN contribution to the model.

605

606 Table 4 compares the performance of the three models in MineTool on the hold-out data.
 607 A visual method of gauging the performance of the results is to plot the actual versus the
 608 predicted values as shown in Fig. 5. In the zero error limit, all data will be lined up along
 609 the 45° line. Visual inspection of this figure along with the error measures in table 4
 610 reveal a number of interesting points. First, the simple regression model (benchmark)
 611 does a reasonable job and with RMSE of ~0.026 would be considered adequate for most
 612 space physics applications. Secondly, the ANN model of MineTool achieves an
 613 amazingly high accuracy.

614

615 Figure 6 shows the distribution of the MRE as a function of $\log(R)$, B_z , $\cos(\theta)$, and $\ln(D_p)$.
 616 Such a figure can be used to help identify any non-uniformity in the model performance

617 as may occur if the quality of data (e.g., coverage, noise, etc.) varies significantly. In the
618 present case the data are more sparse at large values of B_z and dynamic pressure, but the
619 error shows a fairly uniform distribution.

620

621 **3.2.3 Effect of Noise in the Data**

622 We now generate a data set based on Eq. (11) with sigma of 0.1. It is easy to show that
623 this puts a theoretical limit of ~ 0.1 in the accuracy in $\log(R)$ that can be obtained. Using
624 MineTool, the modeling steps are the same as for the noiseless case.

625 The benchmark model is similar in form to that for the noiseless case Eq. (15) but with
626 different coefficients:

$$627 \quad \log(R) = -0.0611228 \cos(\theta_s) - 0.0804243 \ln(D_{ps}) + 0.024 B_{zs} + 0.9884392 \quad (17)$$

628 The best candidate model now consists of only 12 (including the constant) terms as
629 compared to 43 terms for the noiseless case. This is because the presence of noise puts a
630 theoretical limit on the level of accuracy that can be achieved, thereby limiting the
631 number of terms required. This testifies to the power of our algorithm which only keeps
632 the minimum number of terms required to achieve the desired accuracy and hence
633 avoiding overfit.

634 The neural net model (not shown) consists of 8 terms plus 6 hidden layer terms that
635 involve ridgelets, radial basis functions and logistic functions. Table 5 compares the

636 relative performance of the three models in MineTool. As is clear from this table, the
637 accuracy of predictions is very close to the theoretical value in all models.

638 **4.0 Summary and Conclusion**

639 Data mining techniques provide powerful analysis and modeling tools and have been
640 used in a wide variety of fields. However, their adoption in physical sciences in general
641 and in space physics in particular, has been slow. A major obstacle is the steep learning
642 curve of some of the techniques and/or the requirement to have a working knowledge of
643 statistics. Another factor is the existence of a plethora of data mining approaches and it is
644 often a daunting task for a scientist to determine the appropriate technique. Here we
645 presented a technique called MineTool which automates the model building and model
646 evaluation and aids some aspects of the data preparation phase. As we have seen, the
647 underlying technique is quite complex. However, the user is shielded from this
648 complexity through high-level interfaces which hide the data mining concepts away from
649 the users thus helping to bridge the conceptual gap usually associated with data mining.
650 MineTool incorporates the various stages of modeling into a four-step methodology
651 consisting of i) data segmentation and sampling, ii) variable pre-selection and transform
652 generation, iii) predictive model estimation and validation, and iv) final model testing. In
653 standard approach to data mining, the user has the task of deciding on proper
654 strategies/techniques for each of these steps and for meshing them together. This is a
655 daunting task even for an experienced user. Another advantage of MineTool is that the
656 final model is always in a closed analytical form. This facilitates easier dissemination of
657 the model as well as exploration of the effects of various terms. We emphasize that the

658 automation of algorithms does not remove the need for human direction of data mining.
659 Rather data mining techniques should be considered as a powerful tool incorporated into
660 human process of problem solving.

661 *What do these techniques offer?*

662 The computer aided algorithmic approach to data analysis as facilitated through data
663 mining techniques are essential for analysis of large data sets and enable discovery of
664 hidden information and patterns in the data. The test problem considered here illustrates
665 only one type of application of data mining techniques, namely a point forecast or
666 prediction of a *target* variable Y given a vector of *predictors* X , where X is known or
667 observed prior to the realization of Y . Examples of this type of application include
668 models for various boundaries (magnetopause, bow shock, plasma sheet, etc.). Other
669 obvious application of data mining methods discussed here is automated search and
670 identification of events such as shocks, flux ropes, magnetopause crossings, sunspots,
671 coronal mass ejections, among others. Yet another application is discovery of cause and
672 effects and assessment of relative importance of variables in affecting an outcome.

673 *What next?*

674 The goal of this paper was to lay the foundation of the new technique MineTool.
675 However, convincing proof of the viability of such techniques can only be achieved
676 through their application to various problems using spacecraft data. This is beyond the
677 scope of this paper. We have, however, started on several such applications including
678 modeling of magnetopause, automated detection of flux ropes in the magnetotail and at

679 the magnetopause and understanding their trigger mechanisms. We will report on the
680 results elsewhere. Through our work, we hope to make the application of machine
681 learning techniques to space physics a commonplace in the very near future.

682

683

684

685

686 **Acknowledgments.** The research of H. Karimabadi, T. Sipes, and J. Driscoll were
687 supported by a NASA LWS grant NNH06CD22C and a NASA SBIR Program contract.

688 The authors acknowledge useful discussions with Aaron Roberts as well as comments by
689 the two reviewers that led to significant improvement in the presentation of the material.

690

691

692 **References**

693

694

695 Candes, E.. *Ridgelets: Theory and Applications*. PhD thesis, Stanford University,
696 Department of Statistics, 1998.

697

698

699 Lendasse, A., Lee, J. , de Bodt, E., Wertz, V. and Verleysen, M.. Approximation by
700 Radial Basis Function Networks Application to Option Pricing. In *Connectionist*
701 *Approaches in Economics and Management Sciences*, C. Lesage, M. Cottrell eds.,
702 Kluwer academic publishers, 2003, pp. 203-214.

703

704 Looney, C. G., *Pattern recognition using neural networks, Theory and algorithms for*
705 *engineers and scientists*, Oxford University Press, 1997.

706

707 Marinucci, M. (2006) Ph.D. Dissertation, Departamento Departamento de
708 *Economia Cuantitativa*, Universidad Complutense de Madrid.

709

710 Pérez-Amaral, T., Gallo, G. M. and White, H., A Flexible Tool for Model Building: the
711 *Relevant Transformation of the Inputs Network Approach (RETINA)*, *Oxford*
712 *Bulletin of Economics and Statistics*, 65 (s1), 821-838, 2003.

713

714 Pérez-Amaral, T., Gallo, G. M. and White, H., A Comparison of Complementary
715 Automatic Modeling Methods: RETINA and PcGets,” *Econometric Theory*, 2005.
716

717 Powell, M. J. D. *Radial basis functions for multivariate interpolation: A review*. In
718 Algorithms for Approximation, J. C. Mason and M. G. Cox, Eds. Clarendon Press,
719 Oxford, 1987.
720

721 Ripley, B.D., *Pattern Recognition and Neural Networks*, Cambridge University Press;
722 1996.
723

724 Shue, J-H., J. K. Chao, H. C. Fu, C. T. Russell, P. Song, K. K. Khurana, and H. J. Singer,
725 A new functional form to study the solar wind control of the magnetopause size and
726 shape, *J. Geophys. Res.*, 102, 9497, 1997.
727

728 Stinchcombe, M. and White, H., “Consistent Specification Testing with Nuisance
729 Parameters Present Only Under the Alternative,” *Econometric Theory*, 14, 295-325,
730 1998.
731

732 White, H., Approximate nonlinear forecasting methods, in *Handbook of Economic*
733 *Forecasting*, Volume 1, Edited by Elliott, Granger and Timmermann, Elsevier,
734 Amsterdam, 2006.
735
736

737 White, H., Personnel Readiness: Neural Network Modeling of Performance-Based
738 Estimates, *Final Report to the Office of Naval Research, Contract #: N00014-95-C-*
739 *1078, 1999.*
740

741 **Figure Captions**

742

743 **Figure 1.** The High-Level MineTool Illustration

744

745 **Figure 2.** The Architecture of the MineTool Model including both linear and nonlinear
746 (ANN-like) transformations.

747

748 **Figure 3.** The MineTool Algorithm.

749

750 **Figure 4.** Non-Linear Input Variable Transformations Creation.

751

752 **Figure 5.** Plot of actual versus predicted values for a benchmark model based on linear
753 regression and two MineTool's models.

754

755 **Figure 6:** Plots of relative error as a function of $\log(R)$ and three input variables for each
756 of the three MineTool models.

757

758 **Tables**

759

Table 1: The training dataset consists of input variables X_j and the output variable Y . Each row represents a sample of the input and output values.

X_1	X_2	...	X_j	...	X_K	Y
x_{11}	x_{12}		x_{1j}		x_{1K}	y_1
x_{21}	x_{22}		x_{2j}		x_{2K}	y_2
...	...					
x_{i1}	x_{i2}		x_{ij}		x_{iK}	y_i
...	...					
x_{N1}	x_{N2}		x_{Nj}		x_{NK}	y_N

760

761

762

Table 2:			
Input Set	Elements of the Input Set: s=	Selection Factor: 1/s=	Assigned Weight
{a},{b},{c}	1	1	0.5455
{a,b},{b,c},{a,c}	2	1/2	0.2727
{a,b,c}	3	1/3	0.1818
	Sum	11/6	1.0000

763

764

Table 3: MineTool's 10 leading terms of the Best Linear Model

Variable	Correlation with log(R)	Coefficient
$\ln_s(D_p)$	0.757	-9.15e-2
$\cos_s(\theta)$	0.68	7.59e-3
$\cos_s(\theta) \ln_s^2(D_p)$	0.578	-5.67e-2
$[\ln_s(D_p)]^3$	0.564	1.2e-3
$\cos_s(\theta) B_{zs}$	0.531	-1.73e-3
$\cos_s(\theta) [\ln_s(D_p)]^3$	0.463	-7.58e-3
B_{zs}	0.218	3.75e-2
$\cos_s(\theta) \ln_s(D_p) B_{zs}$	0.166	-2.32e-4
$\cos_s^2(\theta)$	0.155	-2.3e-3
$\cos_s(\theta) [\ln_s(D_p)]^2 B_{zs}$	0.155	3.77e-4

766

767

768

769

770

Table 4: Performance comparison of the three MineTool models on the hold-out data

Performance Measure	Benchmark Model	Linear Model	ANN Model
RMSE	2.663E-02	8.56E-04	3.42E-4
MAE	2.064E-02	5.89E-04	2.52E-4
MRE	-9.3E-04	1.58E-05	-1.17E-5

771

772

773

774

Table 5: Performance comparison of the three models in MineTool.

Performance Measure	Benchmark Model	Linear Model	ANN Model
RMSE	9.59E-02	9.66E-02	9.64E-2
MAE	7.68E-02	7.73E-02	7.7E-2
MRE	1.12E-02	-1.1E-02	-1.0E-2

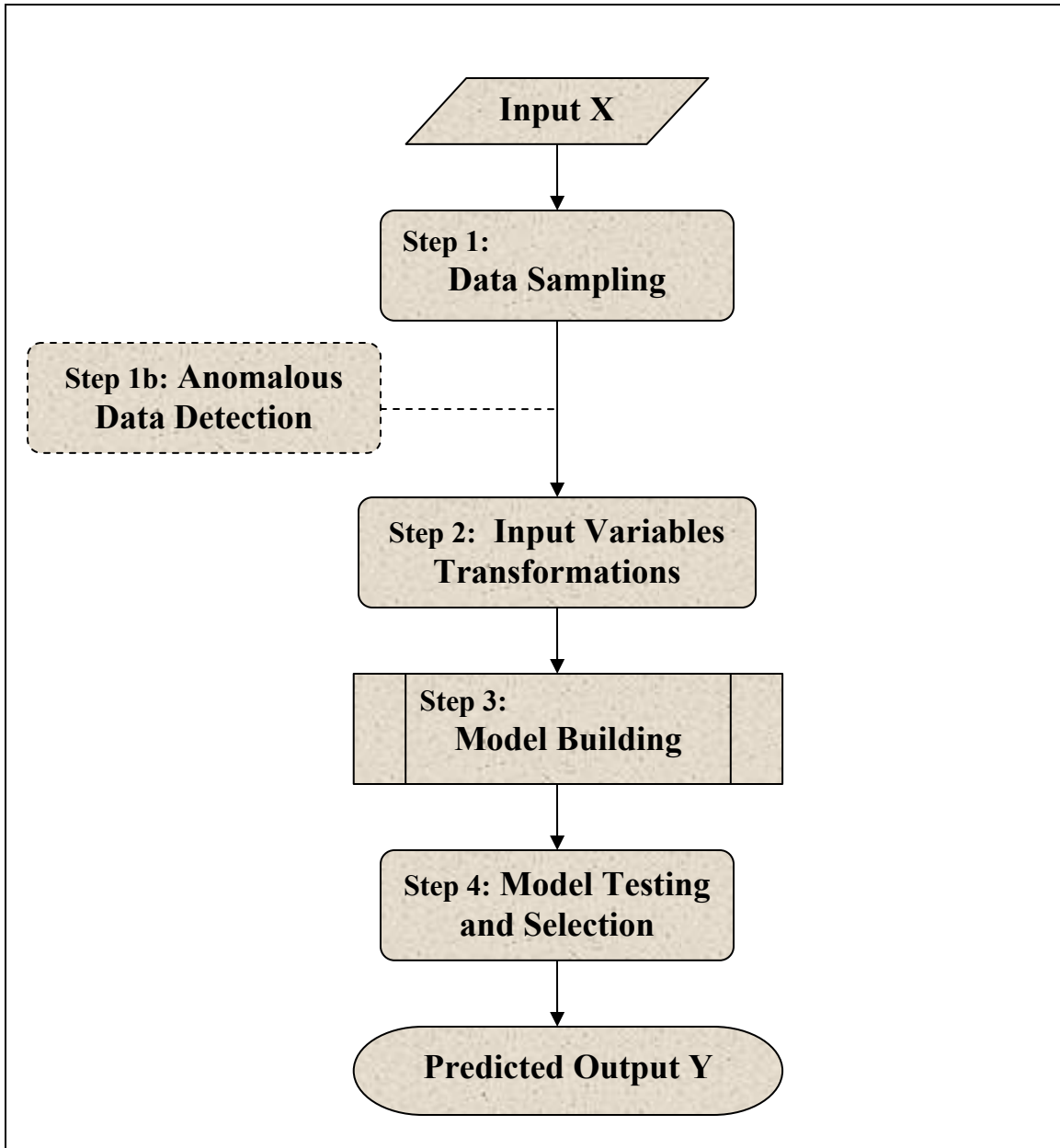
775

776

777 **Figures**

778

779 **Figure 1:**



780

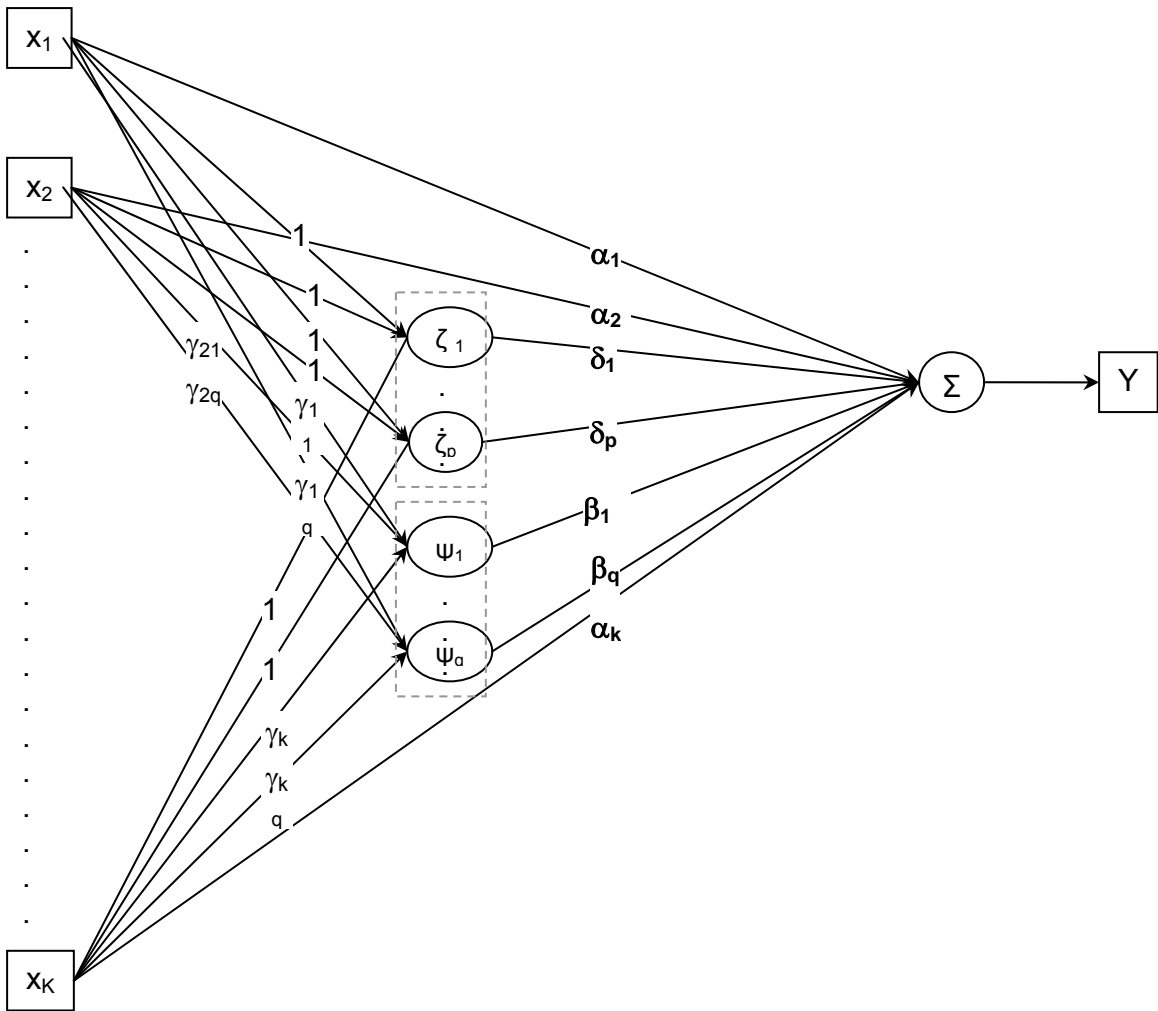
781

782 **Figure 2:**

783

784

785

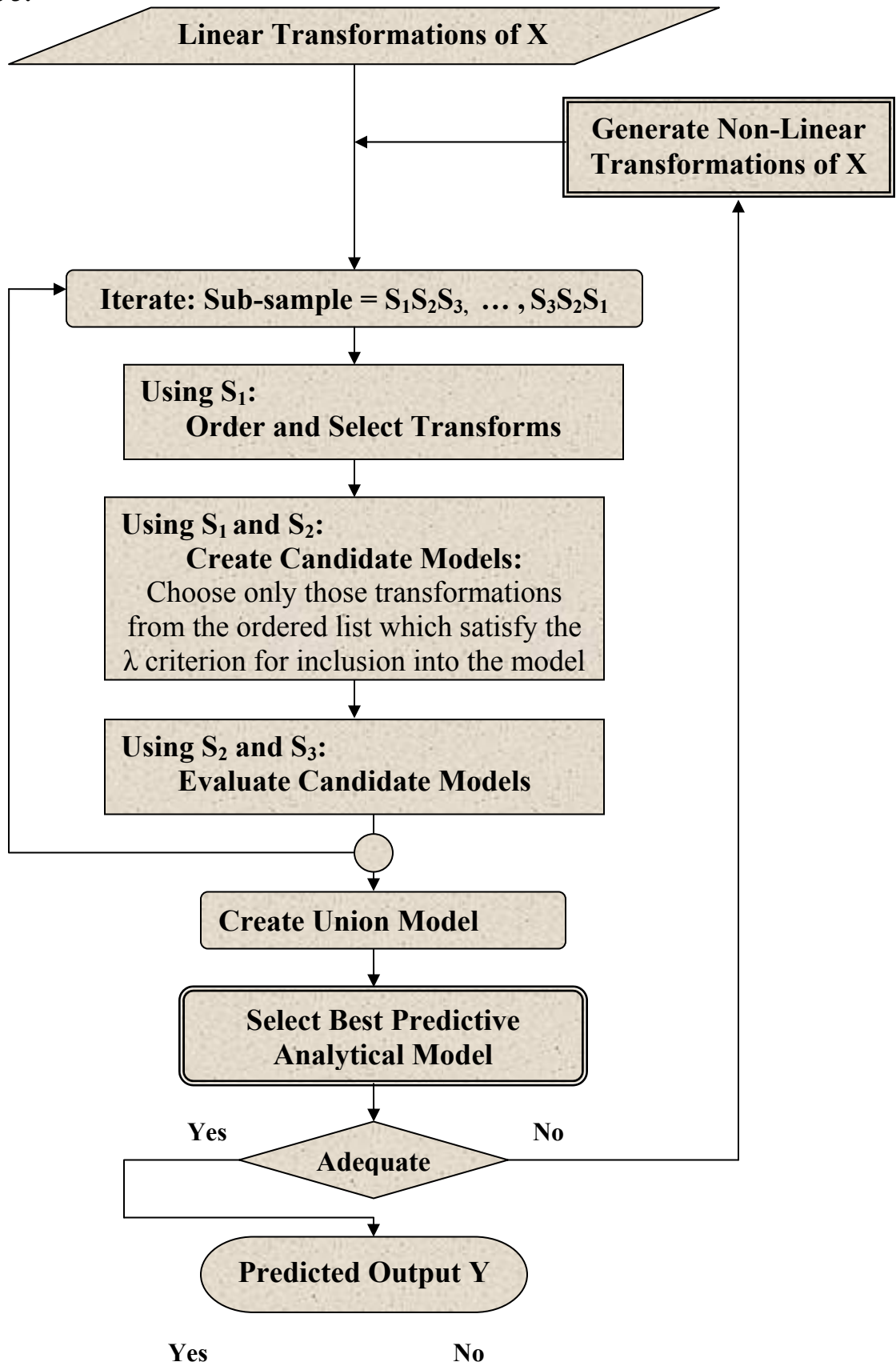


786

787

788 **Figure 3:**

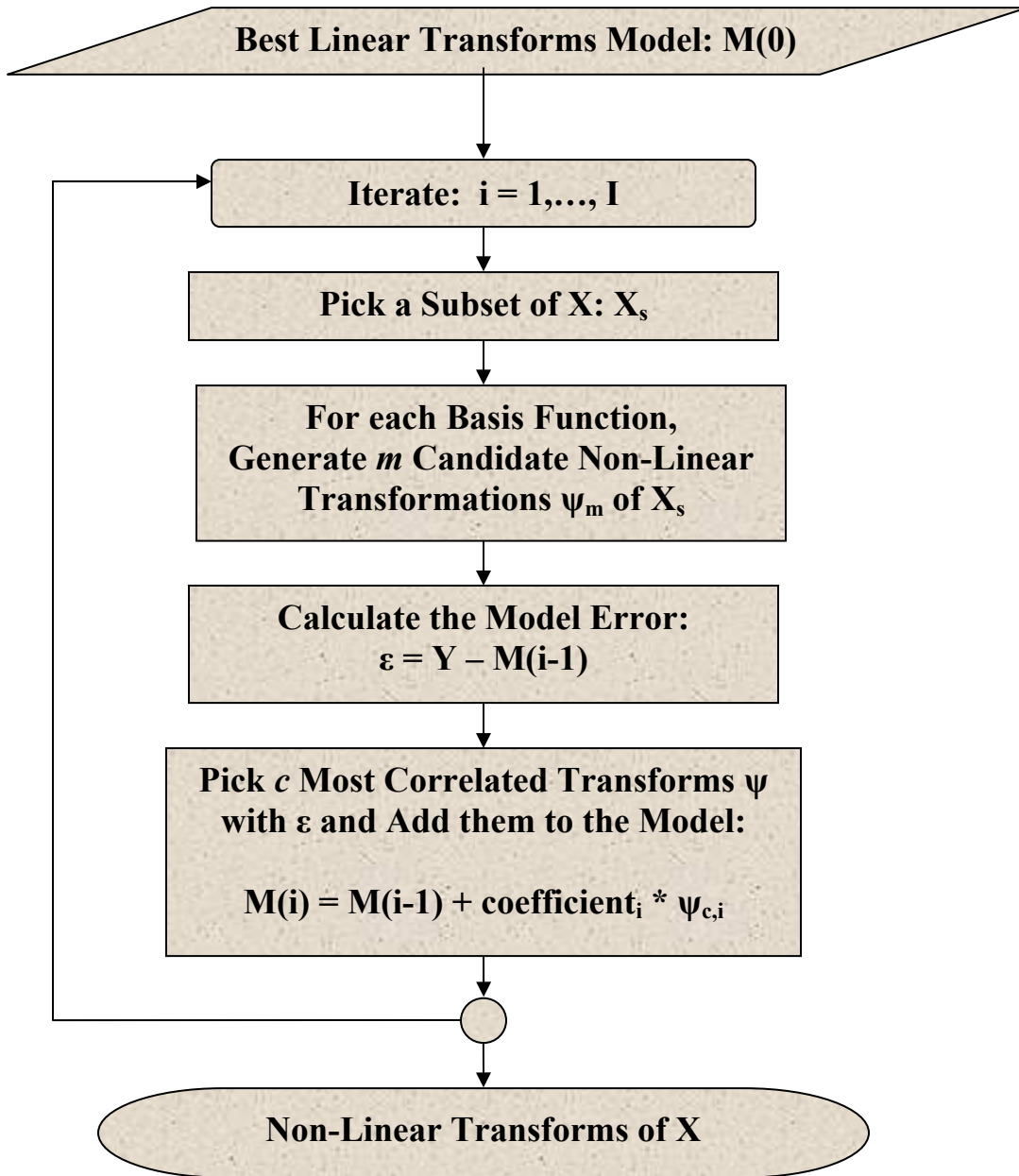
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833



834 **Figure 4:**

835

836



849

850

851

852

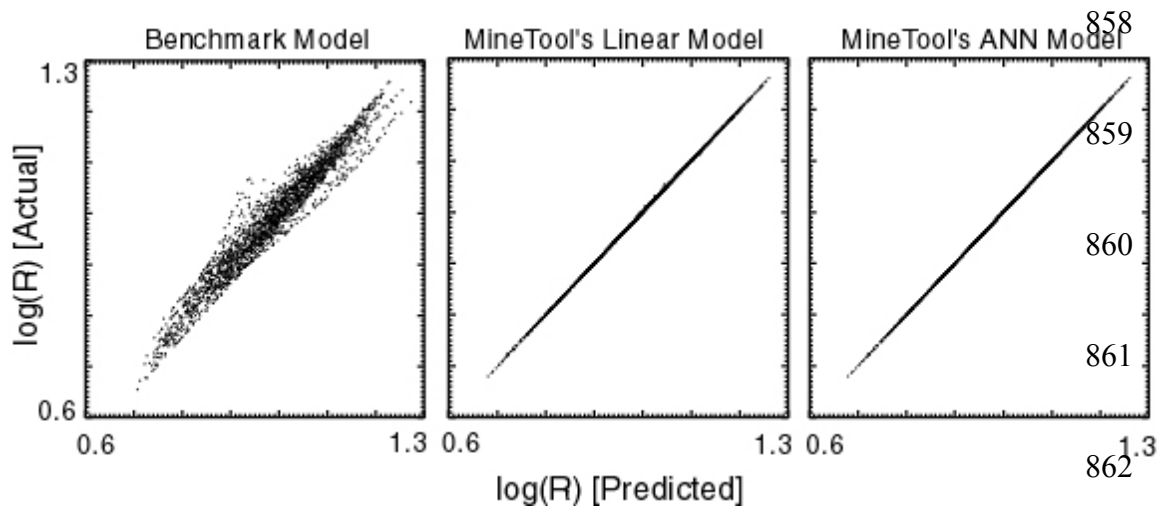
853

854

855

856 **Figure 5:**

857



858

859

860

861

862

863 **Figure 6:**

864

