Analyzing MSI Rules for the USA
Extracted from a Feedforward Neural Network

Vincent A. Schmidt
Air Force Research Laboratory
Dayton, Ohio USA

Jane M. Binner
Aston University
Birmingham, UK

Abstract
This paper introduces a mechanism for generating a series of rules that characterize the money-price relationship for the USA, defined as the relationship between the rate of growth of the money supply and inflation. Monetary Services Indicator (MSI) component data is used to train a selection of candidate feedforward neural networks. The selected network is tuned for rules, expressed in human-readable and machine-executable form. The rule and network accuracy are compared, and expert commentary is made on the readability and reliability of the extracted rule set. The ultimate goal of this research is to produce rules that meaningfully and accurately describe inflation in terms of the MSI component dataset.1

Keywords: MSI, Inflation, Neural Network, Data Mining, Rule Generation

1 Introduction
If macroeconomists ever agree on anything, it is that a long-run relationship exists between the rate of growth of the money supply and the rate of growth of prices - inflation. The objective of current monetary policy is to deliver low and stable inflation. In that regard, it is important to identify indicators of macroeconomic conditions that will alert policy makers to impending inflationary pressures sufficiently early to allow the necessary action to be taken to control and remedy the problem. Given the widely held belief in the long-run relationship between money and prices, monetary aggregates would seem to hold much promise as indicator variables. Nevertheless, evidence to date has not tended to provide strong support for this conviction for the United States; see, for example, Stock and Watson [1]. One possible reason is that measuring money is no easy task. In this paper, we investigate a wide range of money measures for the US and evaluate their usefulness as leading indicators for inflation. There are a wide-range of possible monetary assets beyond currency, which may or may not be included in a particular monetary aggregate. At one end of the spectrum lies "narrow" aggregates, which includes currency and various types of checking accounts including non interest bearing demand deposits. Beyond narrow money lies "zero-maturity" money, which includes, in addition to narrow monetary assets, assets that do not have a fixed maturity such as savings deposits and money market mutual fund assets. Beyond zero-maturity money, lies "broad" monetary aggregates, which include assets with fixed terms to maturity such as small-denomination time deposits. Narrow aggregates include only assets that are primarily valued for their ability to facilitate transactions and, consequently, these assets earn relatively low interest rates if they earn interest at all. Broad aggregates, in contrast, include a wide range of assets of which earn competitive rates of return, such as money market mutual funds. These assets are valued both as a form of savings and for their ability to facilitate transactions (to a greater or lesser extent). In practice, monetary aggregates have evolved over time in response to changes in the economic environment.

Given that traditional monetary aggregates are constructed by simple summation, their use as a macroeconomic policy tool is highly questionable. More specifically, this form of aggregation weights equally and linearly assets as different as cash and time and savings deposits. This form of aggregation involves the explicit assumptions that the excluded financial assets provide no monetary services and the included balances all provide equal levels of monetary services, whether they are cash, checkable deposits, or other types of monetary assets. It is clear that all components of the monetary aggregates do not contribute equally to the economy’s monetary services flow. For example, one hundred dollars worth of currency certainly provides greater transactions services than the equivalent value in time deposits. But, how much more and what is the best way to account for this when aggregating them together?

Barnett [2, 3] attributes to a great extent the downgrading of monetary aggregates to the use of Simple Sum aggregates as they have been unable
to cope with either financial innovation or other factors affecting the substitutability between different types of assets. By drawing on statistical index number theory and consumer demand theory, Barnett advocates the use of chain-linked superlative index numbers as a means of constructing a weighted Divisia index number measure of money. The potential advantage of the Divisia monetary aggregate is that the weights can vary over time in response to financial innovation. [3] provides a survey of the relevant literature, whilst [4] reviews the construction of Divisia indices and associated problems. This paper investigates the performance of a wide range of monetary aggregates for the US including aggregates ranging from narrow to broad and including both Divisia and simple sum aggregates in an inflation forecasting experiment, building on several recent papers including [4–6]. The novelty of this paper lies in the use of the most sophisticated artificial intelligence techniques available to examine the USA’s recent experience of inflation. Our previous experience in inflation forecasting using state of the art approaches gives us confidence to believe that significant advances in macroeconomic forecasting and policymaking are possible using techniques such as those employed in this paper. As in our earlier work, [7], results achieved using artificial intelligence techniques are compared with those using a baseline naïve predictor.

Clearly, the foundations of the construction of monetary aggregates are well rooted in monetary aggregation theory and require extremely strong assumptions. (Barnett and Serletis give a detailed treatment of the theory of monetary aggregation [8].) However, the underlying philosophy of the current research is that all assumptions can be weakened and the Divisia formulation can still be improved.

Many tools have been applied to search for relationships between monetary aggregates and inflation. One such tool is the neural network, a trainable mathematical model that tends to be robust with respect to noise and can generalize well under many circumstances. Gazely and Binner successfully used feedforward neural networks to investigate the relationships between UK Divisia M4 assets and inflation, demonstrating that reasonably simple connectionist architectures are expressive enough to prove the existence of such relationships [9]. Their model was designed to examine sensitivity of the relationships, which were specified as weights within the trained network. Additional effort would be needed to extract and express these relationships.

Our own collaborative research started in 2002 and has progressed annually. We initially trained an Aggregate Feedforward Neural Network (AFFNN) using Divisia components and corresponding inflation values in order to evaluate the feasibility of analyzing Divisia data with this model [10]. At that time, a decompositional algorithm custom-designed for the AFFNN produced a collection of MATLAB-based human-readable and machine-executable if-then rules expressing the discovered relationships in terms of the original data [11]. This algorithm was completely redesigned last year to run on Divisia dataset [12], and is used without modification on the US MSI dataset in our current research. One goal of the current research effort remains to concentrate on the quantity and quality of the rules describing the relationships between money supply and inflation.

The following sections of this paper describe the selection of an appropriate feedforward model. This includes a description of the encoding used for the monetary component asset data of the USA MSI M2 series and corresponding inflation values. Most importantly, this paper includes a discussion of the rules extracted from the selected connectionist model using our most recent decompositional rule extraction algorithm.

2 Dataset Preparation

Historical MSI M2 component and corresponding inflation data was obtained from the United States Federal Reserve Bank of St. Louis in order to determine the relationship between money supply and inflation. Data used for the connectionist model consisted of seasonally adjusted monthly values from January 1961 through January 2006, 541 exemplars in all. For this initial study involving US data, model inputs were constructed as aggregates of the 18 components of MSI M2, summarized in Table 1. From the table, note that commercial bank and thrift institution values were always combined for this model.

The data was prepared using a series of steps. First, for each category of data, the dataset was recalcualted to compute the percentage of increase in value for corresponding months in consecutive years. This reduced the dataset from 541 to 530 exemplars. Then, an automated clustering algorithm was employed to bin similar (recomputed) values within each category of data together. The number of bins was also determined automatically by the algorithm developed in Schmidt [13]. Finally, the bins were used to recode the dataset using a thermometer encoding scheme, a common approach for discretizing continuous data for neural network consumption. Similar data preparation was performed for the inflation values, except the final recoding used a 1-of-N scheme instead of thermometer encoding.

The automated clustering algorithm demonstrated an interesting intermediate result: the val-

---

2 We acknowledge the help of Vice President Richard Anderson at the US Federal Reserve Bank of St. Louis and Professor Barry Jones of the State University of New York at Binghamton for the production of the MSI dataset used in this paper.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Components Included</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Currency</td>
<td>- CUR (Currency),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TRAV (Travelers checks)</td>
</tr>
<tr>
<td>DD</td>
<td>Demand Deposits</td>
<td>- DD (Demand Deposits)</td>
</tr>
<tr>
<td>OCD</td>
<td>Other Checkable Deposits</td>
<td>- OCD (Other checkable deposits, excluding SNOW at commercial banks)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OCDT (Other checkable deposits, excluding SNOW at thrift institutions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SNOWC (Super NOWS at commercial banks)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SNOWT (Super NOWS at thrift institutions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OCDCTOT (Total other checkable deposits at commercial banks)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OCDTTOT (Total other checkable deposits at thrift institutions)</td>
</tr>
<tr>
<td>SAV</td>
<td>Savings and Money</td>
<td>- MMDAC (Money market deposit accounts at commercial banks)</td>
</tr>
<tr>
<td></td>
<td>Market Deposits</td>
<td>- MMDAT (Money market deposit accounts at thrift institutions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SAVC (Savings deposits, excluding MMDA at commercial banks)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SAVT (Savings deposits, excluding MMDA at thrift institutions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SAVMMDAC (Total savings deposits at commercial banks)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SAVMMDAT (Total savings deposits at thrift institutions)</td>
</tr>
<tr>
<td>BDMMF</td>
<td>Non-institutional</td>
<td>- BDMMF (money market mutual funds, non-institutional)</td>
</tr>
<tr>
<td></td>
<td>Money Market Deposits</td>
<td></td>
</tr>
<tr>
<td>STD</td>
<td>Small Denomination</td>
<td>- STDC (small denomination time deposits at commercial banks),</td>
</tr>
<tr>
<td></td>
<td>Time Deposits</td>
<td>- STDT (small denomination time deposits at thrift institutions)</td>
</tr>
</tbody>
</table>
ues of BDMMF (non-institutional money market deposits) seemed to all belong to a single cluster. Since the neural network model would treat this input essentially as a constant, BDMMF was simply eliminated from the training and testing dataset used in this experiment.

Table 2 summarizes the encoding results for the variables used in the final network. (CC and BDMMF are not included in the table because they are not used in the selected model, as described in the following section.) The name of each category is identified, followed by the symbol used to represent the attribute in the generated rule. The final two columns are the number of clusters and the breakpoints dividing the clusters. Four aggregated and encoded attributes comprise the 31-element binary-valued input vectors, and inflation becomes the three-element binary-valued output. Note that these binary-valued vectors are only used for training and testing the neural models. The final rules extracted from the network are expressed in terms of the continuous attribute values (percentage increase). This is the same data preparation and encoding style we’ve used for Divisia experiments performed in previous years [12].

3 Model Selection & Analysis

Our ongoing success with analyzing UK Divisia data encouraged us to try similar methods with the latest M2M2 dataset available from the US Federal Reserve. We are especially interested in determining if the same types of analyses apply to multiple financial datasets, so the marriage of our existing model and the new dataset seemed ideal: perhaps the network-learned rules would provide additional valuable insight that could be applied not only to the US and UK data, but to similar other international material.

Preliminary input encoding showed that BDMMF (non-institutional money market deposits) belonged to a single cluster, so it would not be a factor in training this model. This variable was eliminated to simplify the model.

The automated clustering algorithm suggested the output variable (INFL) be divided into six (6) distinct clusters: [−∞ − 0.3070], [−0.3070 − 0.5675], and four clusters above the 0.5675 breakpoint. Since the top three of the four clusters only contained one or two values each, these were all aggregated into a single cluster [0.5675 − ∞]. This data preprocessing simplification relaxes the training constraints on the neural model.

Note that the central bin [−0.3070 − 0.5675] crosses zero. Since this is an output describing the percentage of increase in inflation, it seemed interesting to split this cluster, explicitly representing inflation below and above zero: [−0.3070 − 0], [0 − 0.5675]. This would allow the neural model to learn the relationship of input producing small negative or small positive inflation, versus simply “inflation near zero.” We refer to this modification as a dataset that has been “zero-crossing-corrected.”

With these simplifications and questions in mind, a series of feedforward neural models were trained and evaluated for potential rule extraction. The training included models with various (but not all) combinations of the following characteristics:

- All aggregated inputs from table 1, all except BDMMF, or all except BDMMF and CC
- 4, 5, or 6 nodes in the hidden layer
- 3 (not zero-crossing-corrected) or 4 (zero-crossing-corrected) INFL output classes

Each candidate architecture used the same encoding for selected inputs, the same encoding for outputs, and contained a single hidden layer, with training and testing data randomized separately for each architecture. All nodes in the hidden layer used a traditional sigmoid activation function (MATLAB’s logsig function) and the unconstrained linear function (MATLAB’s purelin) for the nodes at the output layer.

The training data consisted of a randomly selected set of 345 exemplars (65% of 530), and the testing set contained the remaining 185 exemplars (35% of 530). For each tested network architecture, the single best of 1000 instances was selected for evaluation. Each instance was generated with a random set of initial weights, and trained for 2500 epochs. (Earlier experiments showed no substantial accuracy improvement when trained past 2500 epochs.) The best network instance was defined as the instance with the highest training and testing accuracy. All model execution was performed on a Slackware 10.1 Linux-based (custom SMP 2.6.13 kernel) dual AMD Opteron 244 system with 2GB RAM running MATLAB 5.3 (R11). No instance of model training, in any configuration, exceeded 7.5 seconds. (In general, 1000 instances trained at 2500 epochs each could be accomplished in roughly 2.5 hours.)

The most interesting proof-of-concept experimental results are highlighted in Table 3. The table includes summaries of three feedforward neural network experimental architectures (poc3, poc4, and poc7), all with four nodes in the hidden layer. (There were poc1 and poc2 experiments executed to determine initial variable requirements and gauge network size. Separate poc5 and poc6 tests increased the number of hidden layer nodes with no appreciable accuracy improvements, but produced an exponential number of rules. These results are not considered practical due to combinatorial issues.) The table also reports the number of nodes in the output layer, and whether or not zero-crossing correction was applied (the column labeled “Z?”) in
the table). When training was completed, rule extraction was performed for each output node. The number of rules, as well as the accuracy of the extracted rules on the training and testing data, is shown for reference. (Intermediate checks were made to verify the accuracy of the rules vs. their source neural network. For every network, the rules were in agreement with the corresponding network to an accuracy of 99%). Finally, the table lists the aggregated input variables used in each experiment.

The key difference between the experiments reported in Table 3 lies in the use of zero-crossing correction (the column labeled “Z?” in the table) and the number of input variables used. Recall, without zero-crossing correction, output nodes categorize “INFL % Increase” into three bins:

- Node 1: \([-\infty, -0.3707]\),
- Node 2: \([-0.3707, 0.5675]\]
- Node 3: \([0.5675, \infty]\)

and when zero-crossing correction is applied, the center range is divided into the portion below zero and the portion above zero, resulting in four output node categories:

- Node 1: \([-\infty, -0.3707]\),
- Node 2: \([-0.3707, 0.0000]\]
- Node 3: \([0.0000, 0.5675]\]
- Node 4: \([0.5675, \infty]\)

From the table, note that the training and testing accuracy of rules extracted from all three networks were similar, especially for the non-center (edge) outputs. Experiment poc3 used five aggregated input variables and achieved high training and testing accuracy, but produced 957 rules describing the learned relationships. In comparison, poc7 used only four of the variables (CC was not used), with similar testing and training accuracy, and required only 116 rules to describe the dataset. Even without the inclusion of CC, poc4 (zero-crossing-corrected) produced almost twice as many rules as poc7, without a substantial improvement in accuracy. The (relatively) small number of reasonably accurate

<table>
<thead>
<tr>
<th>Table 2: Divisia M4 Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category (Attribute)</td>
</tr>
<tr>
<td>Demand Deposits</td>
</tr>
<tr>
<td>Other Checkable Deposits</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Savings and Monet Market Deposits</td>
</tr>
<tr>
<td>Small Denomination Time Deposits</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Inflation</td>
</tr>
</tbody>
</table>

![Figure 1: Architecture Selection](image_url)

4 Rule Analysis

Rules are generated using a method closely resembling traditional decompositional approaches, examining the trained network weights and static data. A complete description of the basis of this approach is found in Schmidt, and Schmidt and Chen [13,14].

There are three output nodes for the poc7 experiment, corresponding to the ranges shown in the previous section where zero-crossing correction was not applied. The rule generator produces a separate file representing rules describing each range of values. All rules within a file are tested against the input variables (here, % increase for each variable). If any rule in the file “matches” an input data exemplar, that data is said to be represented by the output range defined by that file. For example, all rules in the “node 3” output file describe conditions producing inflation increases in the range (0.5675 - \(\infty\))%. (That is, inflation % increases > 0.5675.)

As with previous experiments performed on UK Divisia data, each line in a rule is formatted: (low_value <= attr & attr <= high_value), the
mathematical equivalent to: low_value <= attr <= high_value. The symbols "&" and "|" are logical "AND" and "OR" operations, respectively, and Inf represents infinity. The logic of the rule must evaluate to "TRUE" for the rule to be true. If a rule does not include an attribute, then that attribute is not required for the given rule. The rules are also numbered for human reference.

Figure 2 shows an example of a rule (rule #19, in this example) extracted from our trained network. Note the human-readable format and nature of extracted rules, which makes them ideal for validation by subject-matter experts. Since the rules are also represented as Matlab code, they can also be executed by computer and applied to new data.

The rules from all three generated files were examined by both authors, one a subject-matter expert in econometrics. Despite being executable as code, the rules were found to be descriptive and easy to read. Interesting patterns were found merely by examining the rule content. An initial examination of the rules indicates:

1. Rules generated for Node 1 and Node 2 frequently contained complex relationships for STD, but simple relationships for all other vars (as seen in Figure 2). That is, STD often had multiple ranges within each rule, where other variables can be described within a single range. This suggests that the relationships for STD are more complex when inflation changes more than 0.5675% from the previous year.

2. For all generated rules, it is rare to have fewer than all 4 components (DD, OCD, SAV, and STD) participating. This suggests that all 4 vars are important factors in nearly every case; at least all of these components are necessary to describe their complex relationships to inflation.

3. Not many rules are needed to describe outside ranges (Node 1, Node 3) of INFL accurately, but accuracy was not as good for the center range (Node 2). This is demonstrated in the poc7 row of Table 3. Perhaps any data falling into the center ranges should be examined by an additional neural network or other method.

Additional rule inspection indicates (in every case examined except for four) that OCD and SAV (Savings & MMD) have the highest impact on inflation (highest weights). This finding is perfectly valid because OCD and savings are more "moneylike" in the sense of being more liquid than STD and MMMF. They have lower and smoother interest rates than STD and MMMF, hence higher user costs. That is how we are able to identify substitution in response to Fed Funds rate moves. Our results are consistent with Drake and Mills [4] who find that the optimal weights for (OCD,CC, DD) and (SAV, MMD) are consistently higher than the less liquid assets over the period 1960 onwards.

Finally, the actual generated rules for the poc7 model look appealing from an econometrician's point of view because of their small number, readability, and reasonable accuracy. Additional ongoing study will determine if data encoding and other factors can be used to improve the rule accuracy and descriptiveness. If promising results continue to be produced as this research is pursued, they would be of tremendous interest to proponents of Divisia & MSI money.

5 Conclusion

The goal in this paper was to experiment with the methods we have already used with UK Divisia data on a current US MSI dataset, and examine the impact of ongoing model improvements with respect to rule quantity and accuracy. Although some of the interesting improvements (such as our new zero-crossing corrections) were expected to improve results, we found these actually complicated the model results and added combinatorial complications we had worked to eliminate last time. This may be a result of the differences in relationship between component impacts on UK inflation versus US inflation, or it may merely be the result of suboptimal data encoding.

Despite the disappointing outcome of the zero-crossing additional to the model (less accuracy than we had hoped), we believe the overall experiment was a resounding success. US MSI data can be represented by the same model, with no modification of the model required. This data was used to successfully train the feedforward network, and then
was mined for representative rules. In fact, the rule quantity generated for the US dataset is the smallest to date, with excellent overall accuracy.

The rules are still machine-executable and human-readable, but the real objective is to use them to gain insight into the MSI-inflation relationship. It will take time for a trained eye to carefully comb through these generated rules and determine if they offer any specific insight into the issues guiding inflation; we hope to publish the results of our careful examination as the research continues. Preliminary conclusions are encouraging.

Now that we have used this method on both US and UK data, we hope to compare our method to other contemporary techniques as the opportunity allows. Since many other techniques produce equation coefficients instead of specific rules, we are challenged to design an appropriate experiment to ensure the comparison is legitimate, fair, and conclusive.

References


