A Connectionist Approach to Producing Rules Describing Monthly UK Divisia Data

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

Jane M. Binner
Aston University
Birmingham, UK

Abstract

This paper demonstrates a mechanism whereby rules can be extracted from a feedforward neural network trained to characterize the money-price relationship, defined as the relationship between the rate of growth of the money supply and inflation. Monthly Divisia component data is encoded and used to train a group of candidate connectionist architectures. One candidate is selected for rule extraction, using a custom decompositional extraction algorithm that generates rules in human-readable and machine-executable form. Rule and network accuracy are compared, and comments are made on the relationships expressed within the discovered rules. The types of discovered relationships could be used to guide monetary policy decisions.

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

1 Introduction

In recent years the relationship between "money" and the macroeconomy has assumed prominence in academic literature and in Central Banks' circles. Although some Central Bankers have stated they have formally abandoned the notion of using monetary aggregates as indicators of the impact of their policies on the economy, research into the link between some kind of monetary aggregate and the price level is still prevalent. Attention is increasingly turning to the method of aggregation employed in the construction of monetary indices. The most sophisticated index number used thus far relies upon the formulation devised by Divisia [1], with roots firmly based in microeconomic aggregation theory and statistical index number theory.

Our hypothesis is that measures of money constructed using the Divisia index number formulation are superior indicators of monetary conditions when compared to their simple sum counterparts. Our hypothesis is reinforced by a growing body of evidence from empirical studies around the world which demonstrate that weighted index number measures may be able to overcome the drawbacks of the simple sum, provided the underlying economic weak separability and linear homogeneity assumptions are satisfied. Ultimately, such evidence could reinstate monetary targeting as an acceptable method of macroeconomic control, including price regulation.

The theoretical case for weighted monetary aggregates never has been challenged seriously. Their potential for use in practice, however, has been questioned on three fronts. First, criticisms about the choice of a benchmark rate of return and the treatment of risk when measuring monetary user costs (both of which affect index weights) suggest that such an index is subject to unknown, and presumably large, measurement error. Second, if the money stock were measured as the sum of its components, with each weighted by its share of total expenditures on monetary services, it has been alleged (without evidence) that central banks would be unable to influence the behaviour of such an index in the pursuit of a monetary policy objective. Most commonly, however, the case against the construction, publication, and use of any superlative index of money has been grounded in empirical evidence showing that an official simple sum measure, in the context of a particular model, time period, or set of tests, performs as well as or better than a weighted index of the same asset collection. In sum, these perceived shortcomings have led most monetary economists and policymakers to conclude that the practical difficulties associated with finding empirical proxies for a weighted index's theoretical components and explaining the behaviour of such an index to authorities who monitor central bank actions more than offset the small marginal gains (if any) from use of the index itself.

This paper addresses the problem of how best to construct monetary aggregates, given the extraordinary debate on this topic in the macroeconomics literature. A useful summary for 11 countries is provided in Belongia and Binner [2]. Even the superlative Divisia monetary aggregates have been found to perform less than optimally in the recent past using monthly US data over the period 1960–2004 (see [3]), therefore guidance on improved construction of the monetary aggregates is a vital area for further research. Our policy goal in this paper is inflation, the current focus of monetary policy targets in the UK and most major macroeconomies in the world today.

We have jointly examined various aspects of finding relationships in quarterly UK Divisia data for several years (see [4] for the most recent UK Divisia report), and recently applied the same models (using an identical construction approach) successfully to the US's MSI data [5]. Our work together began in 2002 with the use of a specialized feedforward neural model tightly coupled with a custom decompositional rule extraction algorithm. These initial efforts yielded exciting results as a proof of concept, but the rules were both numerous and complex. As our research continued, we were able to demonstrate the discovery of interesting relationships using simpler and more standard feedforward connectionist models, and using a newly decoupling and revised rule extraction algorithm further simplified and reduced the number of rules. (These rules are still automatically produced as a collection of MATLAB-based human-readable and machine-executable *if-then* rules, expressing the discovered relationships in terms of the original data.)

This year we are able to use monthly (vs. quarterly) Divisia data due to its availability, and the complexity and quantity of the generated rules is reduced even further. This paper describes our experimentation with the latest set of monthly UK Divisia data, and compares these models and results briefly with those of the quarterly Divisia work and our recent departure into the US MSI.

2 Dataset Preparation

Historical UK Divisia M4 and corresponding inflation data was obtained[1] in order to investigate the relationship between money supply and inflation. The training data used for connectionist model selection included monthly seasonally adjusted values from January 1988 through September 2007, a total of 117 exemplars. Inflation was constructed for each month as year-on-year growth rates of prices. Our preferred price series, the Consumer Price Index (CPI), was obtained from DataStream. The CPI data originated from the Office for National Statistics (ONS) and all data was seasonally adjusted.

The data was prepared by calculating the percentage of increase in value for corresponding months in consecutive years. This reduced the dataset to 105 exemplars. The automated clustering algorithm we've used in previous studies (to examine quarterly Divisia data) was applied again this time to discretize the monthly component data. Components were represented using thermometer encoding. After inspection, inflation was manually discretized into 3 distinct ranges:

- inflation % changed < 0.010
- inflation % changed 0.010 — 0.020
- inflation % changed > 0.020

Inflation was encoded using mutex (1-of-N) encoding. These two encoding schemes were selected based on the successful training of quarterly Divisia data in our past work. (Mutex and thermometer encoding schemes are commonly used to prepare discretized data for neural network consumption.)

Table 1 summarizes the components and encoding levels generated for each component, as well as for inflation. The table identifies the type of asset (component name), the component ID # and symbol used to represent the component for this study, the type of encoding used, and the number discretized component levels. When all components are used as inputs to the neural

[1]Component data is available on the Internet at http://www.bankofengland.co.uk/statistics/index.htm (Bank of England Statistical Interactive Database).

Table 1: Divisia M4 Encoding

Component (Attribute)	ID #	Symbol	Encoding	Levels
Notes and Coins	1	NC	Thermometer	2
Non-Interest Bearing Bank Deposits	2	NIBD	Thermometer	5
Interest Bearing Bank Sight Deposits	3	IBSD	Thermometer	14
Interest Bearing Bank Time Deposits	4	IBTD	Thermometer	9
Building Society Deposits	5	BSD	Thermometer	2
ISA and TESSA (tax-free savings)	6	ISA	Thermometer	6
Inflation	N/A	INFL	Mutex	3

network, there are 38 binary-valued inputs to the network, and 3 binary-valued outputs (representing inflation).

3 Neural Network Selection

A series of carefully controlled tests were performed to determine the best type of simple feedforward connectionist models to use for rule generation. The 105 data cases were divided at random into a training set (80%, 84 cases) and a validation set (20%, 21 cases). The breakout was examined to ensure that the components and outputs were reasonably represented in both the training and validation data. This same randomly generated selection was used for each test.

The results of these tests are summarized in Tables 2, 3, and 4. The tables include columns for training and validation "success" expressed as a number and percentage of correct outputs. All network architectures were trained to find the INFL target, a set of 3 mutex-encoded binary values corresponding to each data case. Training (and validation) success was measured by determining the number of total binary matches for all training (and validation) cases:

- Training: (84 cases)x(3 outputs) = 252
- Validation: (21 cases)x(3 outputs) = 63

For each candidate model architecture (table row), 500 models with randomly generated initial conditions were trained for 2500 epochs each, with the "best" model instance selected to represent the specified model architecture. It is important to note that "best" is a somewhat arbitrary term due to the way the "best" network is selected in our study. When numerous networks with discrete outputs are trained, they tend to fall into classes, where multiple networks yielding identical results all belong to the same class. We simply choose a network from the class of all networks yielding the most accurate training and validation results. For comparison, the tables below indicate how many clusters the 500 trained networks fall into for each architecture ("Net Clusters"), and how many of these 500 are members of the "class of best networks" from which our selection is made ("Qty"). This data is intended to ease the minds of those concerned with our selection of the "best network" for each architecture as we proceed with our analysis.

No instance of any network trained for longer than 6 seconds on the experimental machine, 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). The neural models executed in this study contained (except where indicated otherwise) a single hidden layer using MATLAB's *logsig* function, a traditional sigmoid activation function. The nodes in the output layer use the unconstrained linear function (MATLAB's *purelin*).

Table 2 is for models where only a single component is used as an input to a network model with 5 nodes in the hidden layer. Components 4 (IBTD) and 6 (ISA) seem to be the most reliable individual indicators of inflation, based on their training accuracy of over 82%, but neither component has an overwhelmingly good validation accuracy (77-80%).

We also trained a series of models where the inputs lead directly to the outputs (no hidden layer). Surprisingly, these training and validation results (not shown here) were almost identical to the models trained with a single hidden layer of 5 nodes as shown in Table 2.

As a quick test of sensitivity analysis, we also trained a series of models where all components *except* for one specific component were included

Table 2: Single Components as Inputs to n-5-3 FF Networks

ID	Inputs	Train (of 252)	Valid. (of 63)	Net Clusters	Qty in Best Cluster (of 500)
1	2	198 / 78.57 %	45 / 71.43 %	1	500
2	5	196 / 77.78 %	45 / 71.43 %	4	317 (148, 34)*
3	14	203 / 80.56 %	49 / 77.78 %	5	1 (276, 183)*
4	9	216 / 85.71 %	51 / 80.95 %	2	277 (223)*
5	2	196 / 77.78 %	45 / 71.43 %	1	500
6	6	208 / 82.54 %	49 / 77.78 %	1	500

* Quantity in next best cluster(s) shown in parentheses for reference

as inputs. The rows of Table 3 identify the component not included as inputs to the network. All network models have 5 nodes in their hidden layer for this series of tests.

A more traditional approach was also taken; a collection of models was trained with various numbers of nodes in the hidden layer. Table 4 reflects the results of these tests, all of which use all 6 components as inputs to the network (38 encoded values in each input vector).

The goal of training various architectures is to find an appropriate model from which a collection of human-readable rules can be generated to accurately describe the dataset. Experimental models using only a single component as input, either with or without a hidden layer, was not convincingly accurate enough to justify continuing with these simpler models. Using all but one component showed promise, but didn't really suggest any specific component could be easily eliminated.

The use of all components as inputs in the model consistently yielded the best results. Note, however, the excellent results the neural model containing no elements in the hidden layer (the row with H as "0" in the H column of Table 4), hence no hidden layer. These results are nearly as good as models containing 10 nodes in the hidden layer!

4 Rule Generation

Since most of the models in Table 4 had a high degree of accuracy in training and validation, we chose to do rule extraction on the simplest model, the network containing a hidden later with the fewest (non-zero) number of nodes: the 38-2-3 model (2 nodes in the hidden layer). This network is depicted in Figure 1.

Metrics were collected during rule extraction in order to verify the rules would be a faithful

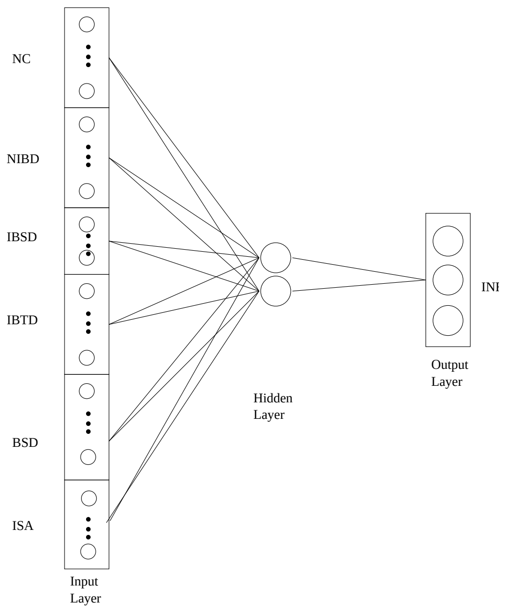

Figure 1: Architecture Selection: 38-2-3

reproduction of the relationships learned by the network. One intermediate test ran an exhaustive combination of all possible discrete inputs through the neural network, examining the differences in output produced by the network and the rule generation process. Table 1 indicates the number of discrete bins for each input. The total number of all combinations of possible inputs is a simple product of these values: 2 * 5 * 14 * 9 * 2 * 6 = 15120. Looking at each of the three output nodes individually, the accuracy (compared to providing the same inputs to the neural network) of the intermediate rule extraction is (in number of mismatches per 15120):

- Node 1: 6 mismatches, 99.96% match
- Node 2: 21 mismatches, 99.86% match
- Node 3: 5 mismatches, 99.97% match

Although this check is merely a quick test taken during the extraction exercise, it is encouraging

Table 3: Single Components Excluded as Inputs to n-5-3 FF Networks

ID	Inputs	Train (of 252)	Valid. (of 63)	Net Clusters	Qty in Best Cluster (of 500)
1	36	244 / 96.83 %	61 / 96.83 %	16	1 (4, 15)*
2	33	244 / 96.83 %	62 / 98.41 %	16	2 (7, 12)*
3	24	234 / 92.86 %	61 / 96.83 %	8	8 (110, 117)*
4	29	228 / 90.48 %	63 / 100.0 %	15	1 (2, 6)*
5	36	244 / 96.83 %	61 / 96.83 %	16	2 (5, 25)*
6	32	236 / 93.65 %	58 / 92.06 %	17	2 (3, 6)*

* Quantity in next best clusters shown in parentheses for reference

Table 4: All Components, Variable Hidden Layer Nodes in 38-n-3 FF Networks

H	Train (of 252)	Valid. (of 63)	Net Clusters	Qty in Best Cluster (of 500)
0	244 / 96.83 %	60 / 95.24 %	6	16 (88, 203)*
2	242 / 96.03 %	61 / 96.83 %	15	6 (4, 43)*
3	242 / 96.03 %	62 / 98.41 %	17	2 (6, 3)*
4	244 / 96.83 %	62 / 98.41 %	16	2 (4, 7)*
5	244 / 96.83 %	62 / 98.41 %	17	1 (3, 9)*
6	244 / 96.83 %	63 / 100.0 %	18	1 (2, 6)*
7	244 / 96.83 %	62 / 98.41 %	16	1 (5, 10)*
8	244 / 96.83 %	63 / 100.0 %	16	1 (1, 8)*
9	244 / 96.83 %	62 / 98.41 %	17	1 (4, 10)*
10	244 / 96.83 %	61 / 96.83 %	14	3 (11, 29)*

* Quantity in next best clusters shown in parentheses for reference

to see the high correlation between the trained network and the "intermediate" extracted rules.

The rule extraction technique applied for this research is a traditional decompositional approach, peering back through the trained network with an emphasis on the values dynamically generated by the hidden nodes. For each hidden node, all values are automatically clustered, and a representative (mean) value is assigned to each cluster. All combinations of these mean values are evaluated against the output node weights to determine combinations ("candidate expressions") producing the desired outputs. These candidate expressions are simplified and re-expressed as simple rules in terms of the original network inputs. This is the same rule extraction algorithm we devised for our previous Divisia and US MSI research efforts, based on the algorithm originally described in Schmidt and Chen [6].

The automated binning algorithm originally separated INFL outputs into fifteen bins, but we artificially re-binned the outputs into three groups. This is more consistent with the automated results from our previous research, and still yields an excellent mix of potential outputs among the bins. The bins for these outputs are:

- Node 1: $(-\infty \ldots 0.01)$
- Node 2: $(0.01 \ldots 0.02)$
- Node 3: $(0.02 \ldots \infty)$

The rule generator produces rules describing each range separately, so each rule file corresponds to a specific output node, representing a specific range of output values. (The current generation algorithm generously allows boundary conditions between two nodes to be represented in both rulesets.) Note that these rules are expressed in terms of the original input values for readability, verses the encoded forms.

Each file contains a list of rules, numbered for reference by human readers for convenience. If some combinations of attribute values (nc, nibd, etc.) is described by any rule in a specific file, those values would be expected to result in the "inflation increase %" represented by that node. I.e., all rules in the "node 2" output file describe conditions producing inflation increases in the range (0.01 ... 0.02) %.

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

```
    if (
      (-Inf <= nc & nc <= 0.091411) ...
    & (-Inf <= nibd & nibd <= 0.447313) ...
    & ( (0.076953 <= ibsd & ibsd <= 0.106028) ...
      | (0.116952 <= ibsd & ibsd <= 0.122060)) ...
    & (0.122171 <= ibtd & ibtd <= 0.190650) ...
    & (-Inf <= bsd & bsd <= 6.845814) ...
    & (-Inf <= isa & isa <= 0.186993) ...
    ) return true;
```

Figure 2: Sample Generated Rule

Table 5: Generated Rule Accuracy

Output Node	Rule Qty	Training set (correct, of 84)	Validation set (correct, of 21)
1	57	82 (97.62%)	20 (95.24%)
2	64	79 (94.05%)	19 (90.48%)
3	10	81 (96.43%)	21 (100%)

"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, the attribute is not required for the given rule.

Figure 2 shows an example of a rule extracted from our trained network. The example clearly demonstrates the human-readable format and nature of extracted rules. This makes them ideal for validation by subject-matter experts. These rules can also be executed as code and applied to new data.

Table 5 shows the number of rules generated for each output node. The original unencoded training and validation data were processed by the rule files, with the outputs tested against the known targets for each dataset. The table clearly indicates a good match between the learned relationships (rules) and the actual data.

Once again, the chief value provided by the rules is that they are human-readable and can be vetted by a subject-matter expert (econometrician, in this case), while also being machine-executable.

5 Interpretation

The generated rules in all three output files were examined by one of the authors, a subject-matter expert in econometrics, for specific applications in economic theory. Although the rules are expressed as executable code, they were found to be descriptive and easy to read.

Inspection of the rules indicates exactly the same trend as we saw in our analysis of US MSI data: higher yielding assets have higher impact on inflation than the lower yielding assets, which conforms with the construction of Divisia / MSI aggregates. This adds credence to the argument that we should construct statistically weighted aggregates as our money supply measure.

These conclusions continue to be consistent with our own previous analysis of UK Divisia quarterly data, our recent work with US MSI data, and contemporary published results from other sources. See Barnett [7] and Elger and Binner [8] for a more detailed description of user costs of monetary assets.

There are also some interesting relationship patterns that can be seen from simple inspection of the resultant rules. Table 6 shows the frequency of occurrences of components within the generated rules. From the Table, all of the components are generally important inputs for describing the learned relationships. IBSD and IBTD are frequently included multiple times to describe cases when "INFL % increase < 0.02" (the first two columns of the Table). (See Figure 2 for an example where IBSD is referenced twice in the same relationship.) In addition, the last column of the Table shows that BSD and ISA are only important about half of the time for the relationships describing "INFL increase > 0.02." The implications of these results will merit closer evaluation. In most cases the rule complexity is fairly low, with each component being mentioned only once per relationship.

It is worth noting that, for the monthly Divisia data, there are 131 total rules across three output nodes (57 + 64 + 10), with a minimum accuracy of 90%. Our most recent quarterly Divisia experiments yielded 714 rules (96 + 256 + 282 + 80) for four output nodes. The best results of our US MSI study yielded 116 rules across three outputs, but accuracy varied between 75% and 92% for each output.

6 Conclusion

The goal of this research effort was to generate rules describing the relationship of monthly Divisia component data as it applies to prediction of inflation. A collection of connectionist models were trained to learn these relationships, then a representative model was chosen for rule extraction. The successfully generated rules were shown to be reasonable in number, accurate with respect to both training and valida-

Table 6: Component Frequency of Occurrence

Component Name	Output 1 (of 57) (INFL < 0.01)	Output 2 (of 64) ($0.01 <$ INFL < 0.02)	Output 3 (of 10) ($0.02 <$ INFL)
NC	57 / 100%	58 / 91%	10 / 100%
NIBD	57 / 100%	64 / 100%	9 / 90%
IBSD	87 / 153%	92 / 144%	10 / 100%
IBTD	64 / 112%	79 / 123%	10 / 100%
BSD	57 / 100%	61 / 95%	4 / 40%
ISA/TESSA	62 / 109%	67 / 105%	6 / 60%

tion data, a faithful representation of the trained neural network, and (most importantly) easy for econometric experts to visually examine. These rules, expressed in terms of the original unencoded data, are also machine-executable MATLAB code, and can be used independently of the original neural network.

The data used in this series of experiments included the ISA/TESSA term. The inclusion of this term proved to be a valuable addition to the five terms already used in previous models. The seasonally adjusted monthly data also yielded superior modeling results and rule quality when compared to the seasonally adjusted quarterly Divisia data we've used in the past. The results we report in this series of experiments is also consistent with other contemporary published model results.

It is our hope that techniques such as the one represented here can be commonly employed to provide useful inputs for prediction and control of inflation. Calibration of these results in a large scale macro model would still be an interesting route to pursue to determine the full extent of the impact and implications of these rules for the U.K. economy.

References

[1] Francois Divisia. L'indice monétaire et al théorie de la monnaie. *Rev. d'Econ. Polit.*, 39:980–1008, 1925.

[2] Michael T. Belongia and Jane M. Binner. Divisia monetary aggregates: Theory and practice. *Palgrave Publishers Ltd UK.*, 2000.

[3] Jane M. Binner, Barry E. Jones, Peter Tino, Jonathan Tepper, and Graham Kendall. Does money matter in inflation forecasting. *Physica A (under review)*, 2008.

[4] Vincent A. Schmidt and Jane M. Binner. Analyzing Divisia rules extracted from a feedforward neural network. In *Proceedings of the 2006 International Conference on Artificial Intelligence*, Las Vegas, Nevada, 2006.

[5] Vincent A. Schmidt and Jane M. Binner. Analyzing MSI rules for the USA extracted from a feedforward neural network. In *Proceedings of the 2007 International Conference on Artificial Intelligence*, Las Vegas, Nevada, 2007.

[6] Vincent A. Schmidt and C. L. Philip Chen. Using the aggregate feedforward neural network for rule extraction. *International Journal on Fuzzy Systems*, 4(3), 2002.

[7] William A. Barnett. The user cost of money. *Economic Letters*, 1(2):145–149, 1978. Reprinted in W.A. Barnett and A. Serletis (Eds.) The Theory of Monetary Aggregation, North-Holland, Amsterdam, Chapter 1, 2000, 6–10.

[8] Jane M. Binner and Thomas Elger. The UK household sector demand for risky money. *Berkeley Electronic Press, Topics in Macroeconomics Series*, 2004. http://www.bepress.com/bejm/topics/vol4/iss1/art3.