ISSN 1178-2293 (Online)

University of Otago
Economics Discussion Papers
No. 1404

March 2014

Artificial Neural Networks and Aggregate Consumption Patterns in New Zealand

Daniel Farhat

University of Otago – Department of Economics

Address for correspondence:

Dan Farhat
Department of Economics
University of Otago
PO Box 56
Dunedin
NEW ZEALAND
Email: dan.farhat@otago.ac.nz
Telephone: 64 3 479 8645

Artificial Neural Networks and Aggregate Consumption Patterns in New Zealand

[DRAFT: October 2013]

Daniel Farhat
University of Otago – Department of Economics
PO Box 54, Dunedin, New Zealand 9054
dan.farhat@otago.ac.nz
(Tel) +64 03 479 8645

Abstract: This study engineers a household sector where individuals process macroeconomic information to reproduce consumption spending patterns in New Zealand. To do this, heterogeneous artificial neural networks (ANNs) are trained to forecast changes in consumption. In contrast to existing literature, results suggest that there exists a trained ANN that significantly outperforms a linear econometric model at out-of-sample forecasting. To improve the accuracy of ANNs using only in-sample information, methods for combining private knowledge into social knowledge are explored. For one type of ANN, relying on an expert is beneficial. For most ANN structures, weighting an individual's forecast according to how frequently that individual's ANN is a top performer during in-sample training produces more accurate social forecasts. By focusing only on recent periods, considering the severity of an individual's errors in weighting their forecast is also beneficial. Possible avenues for incorporating ANN structures into artificial social simulation models of consumption are discussed.

Keywords: Artificial neural networks, forecasting, aggregate consumption, social simulation

JEL codes: C45, E17, E27

1. Introduction

In standard macroeconomic models, the household sector is often comprised of homogeneous, forward-looking, well-informed, perfectly rational agents (*homo economicus*). These agents are embedded into a grad model of the economy and participate in deriving a state of general equilibrium. Frameworks of this sort have been favoured because they produce unique, tractable equilibria which are straightforward to interpret and evaluate empirically. However, the unrealistic assumptions associated with this approach and the explanatory power of "solving for an equilibrium" draw criticism. In reality, it is difficult for ordinary people to think like *homo economicus*. Often, diverse populations of imperfect, ill-informed individuals follow rules-of-thumb and rely on social interactions to make decisions. The state of general equilibrium in these models is artificially-coordinated. We can derive it mathematically, but cannot explain how it is attained or maintained. To address these criticisms, the household sector in our models needs to be re-engineered.

With modern computing, we can deviate from the concept of *homo economicus* and standard notions of equilibrium, yet still craft a model capable of producing output. For example, it is possible to simulate an entire artificial society populated by heterogeneous individuals.[1] In this approach, artificially-intelligent agents are fabricated by the researcher. These individuals interact with each

[1] For examples of this work, see Tesfatsion (2002, 2005), Tesfatsion & Judd (2006), Gatti, et al. (2005), Mirowski (2007), Gaffeo et al. (2008) and Raberto et al. (2008).

other and their environment according to a set of prescribed rules (also designed by the researcher). A repeating algorithm then simulates the evolution of the artificial population. As the simulation runs, aggregate patterns form which can later be interpreted and evaluated.[2] The computer can easily monitor and record each interaction, implement complex procedures, and accommodate both randomness and structural change. Thanks to the flexibility of computer programming, there are more options in choosing design of the agents, the set of rules they follow, and the algorithm guiding the activities of the model. In this area of research, known as *agent-based computational economics* [ACE], agent design becomes a central topic of discussion.

This study is inspired by the ACE approach; a diverse household sector capable of replicating real aggregate consumption for the New Zealand economy is engineered. In the household sector, consumers are designed to be simple information processors. Heterogeneous individuals assess the state of the economy using publically-available macroeconomic data and then follow a prescribed rule to plan consumption expenditures (as measured by changes in the growth rate of consumption). We look for a set of socially-influenced rules that are capable of producing realistic consumption patterns.

To do this, a convenient tool for processing information is used: the *artificial neural network* [ANN]. Like neurons in the brain, artificial neural networks transform input data into output data. These networks are trained to produce accurate output patterns (in this case, consumption patterns). In the beginning, we give each agent a pre-set artificial neural network structure, but provide them each with different data to train it with. The result is a set of heterogeneous consumption rules, some of which are quite accurate in replicating out-of-sample consumption patterns. We can then combine the individual rules in a number of ways to produce "social" rules. We show that these social rules perform quite well. Extra diversity is added by allowing the artificial neural network structures themselves to differ across agents. Again, when combined to form social rules, we can produce quite reasonable consumption patterns.

This study creates a very crude, rudimentary ACE framework, two accomplishments are generated. First, the results contrast existing literature that shows using ANNs to forecast consumption patterns provide only meagre benefits. As such, researchers interested in macroeconomic forecasting may find the approach useful. Second, the study shows that particular combinations of ANN structure and social rule formation work well for reproducing consumption patterns. This can be used to inform more detailed agent-based models in the future. The remainder of this paper is as follows. First, a brief description of the algorithm used to construct and train the ANNs in this study is provided. Next, a series of experiments are conducted in which agents, provided with incomplete and heterogeneous information, are asked to train ANNs to make reliable forecasts about changes in per-capita consumption in New Zealand. The precision of these forecasts are evaluated using out-of-sample data. In a second series of experiments, the individual forecasts are aggregated into a social forecast which is also evaluated. A final group of experiments allows the ANN topology to differ across agents. The article ends by motivating future work from the results.

2. Method

Artificial neural networks (ANNs) are mathematical algorithms that transform input data into output data. They function in a fashion similar to neurons in the brain, which enhance or inhibit incoming impulses before transferring them on to other neurons. The highly flexible, massively parallel and potentially non-linear structure of ANNs, along with their ability to actively learn from their input information, makes them ideal for identifying complex relationships in data. As such, they have been shown to be extremely useful for solving pattern recognition, optimization and forecasting problems. In the model below, each agent will be given an ANN topology and a set of data containing inputs (here, economic variables commonly known by consumers) and outputs (here, a measure of consumption). They will use the data to teach their ANN how to reproduce in-sample output patterns. The trained ANNs can then be used for out-of-sample information processing.

[2] Since the researcher designs of the agents, the structure of their environment, and the algorithm guiding the activities within these models, any emerging aggregate patterns which resemble those in real world data are more appropriately *explained.* See Epstein (1999) for the relationship between explanation and systemic generation.

Beltratti et al. (1996), Jain and Mao (1996), Warner and Misra (1996), Cooper (1999), Gonzalez (2000), and Detienne et al. (2003) provide easy-to-follow introductory guides on ANNs. The best way to understand the basic principle behind ANNs is too look at the simplest type (known as a *perceptron*). In this basic model (Figure 1), K different pieces of input data (X_k, k = 1...K) are fed into the network to produce an output ($\hat{C}$). To generate $\hat{C}$, the input data is passed through "hidden layers" where different mathematical transformations are made. The raw data is weighted (by ω_k) then summed along with a constant term (or *bias* - ω_0). The result is then passed through a function (called an *activation function*, A). The activation function is often either linear or *sigmoid* (s-shaped) in nature. The result from this step is then weighted (γ_1) and summed with a second bias term (γ_0) to generate $\hat{C}$. The perceptron shown in Figure 1 can be thought of as a single neuron which augments input signals so as to produce outputs (just as biological neurons enhance or inhibit incoming impulses before passing them on to other neurons). Appropriate weights (ω's and γ's) must be found for the ANN to generate accurate outputs. The process of *training* an ANN involves finding these weights.

[Figure 1 here]

The more complex model shown in Figure 2 is used for this study. It works in the same manner as the simple perceptron except that there are *H*+1 neurons whose combined efforts produce the output, $\hat{C}$. One neuron has a linear activation function which simply passes the input data (weighted by δ_k) directly to the output (this is shown in the "lower level" [dashed lines] of the diagram). The other *H* neurons use non-linear transformations of the input data (denoted as A_h for simplicity – shown in the "upper level" [solid lines] of the diagram). When the inputs to (and the output from) each neuron can be either positive or negative, the hyperbolic tangent sigmoid function (tanh) is useful to use for A_h. As a result, A_h will be bounded by $\pm$ 1. For any non-linear neuron, *h*, we can represent the data processing process algebraically as:

$$n_h = \omega_{0h} + \sum_{k=1}^{K} \omega_{kh} X_k$$

$$A_h = \frac{e^{n_h} - e^{-n_h}}{e^{n_h} + e^{-n_h}}$$

where ω_{kh} is the weight that neuron h places on input k, ω_{0h} is the bias term, and A_h is the result from passing the input data through the neuron's activation function. The result from each non-linear neuron is weighted (γ_h) and, along with an additional bias term (γ_0), is combined with the linear transformation to calculate the final output:

$$\hat{C} = \gamma_0 + \sum_{h=1}^{H} \gamma_h A_h + \sum_{k=1}^{K} \delta_k X_k$$

[Figure 2 here]

The ANN model described in Figure 2 proves to be quite flexible and applicable. Many econometric analyses rely on linear models, but ANNs allow for non-linear relationships between inputs and outputs which can enhance the model's fit. Kuan and White (1994) note that the presence of the linear "lower level" in the ANN described above makes the model directly relatable to many standard linear econometric analyses. The ANN is, in effect, a linear model of the form $\gamma_0 + \sum_{k=1}^{K} \delta_k X_k$ augmented by a non-linear function, $\sum_{h=1}^{H} \gamma_h A(\omega_{0h} + \sum_{k=1}^{K} \omega_{kh} X_k)$. Also, although the input variables in Figure 2 are *fully-connected* to the non-linear neurons (all input variables are used by each non-linear neuron) as shown in the figure, it is possible to easily generate ANNs with *semi-*

connected inputs by forcing select ω-weights to zero. This provides greater scope for incorporating diversity into the analysis. While it is possible to make a more complex (and thus more effective) model by adding additional hidden layers (where the output, $\hat{C}$, is passed on as an input to other neurons, and so on), it has shown that this is unnecessary.[3] By selecting a large number of non-linear neurons (*H*), the model shown in Figure 2 can capture quite complex relationships between inputs and outputs.

Usually, input and output data correspond to a specific time period. Denoting T as the total number of available periods, we feed input data associated with any period $t \in T$, $\{X_{1t}; X_{2t}; ...; X_{Kt}\}$, into the ANN to produce an output for the corresponding time period, $\hat{C}_t$. The input and output data can be organized as time-ordered matrices:

$$X = \begin{bmatrix} X_{11} & \dots & X_{1T} \\ X_{21} & \dots & X_{2T} \\ \dots & \dots & \dots \\ X_{K1} & \dots & X_{KT} \end{bmatrix}$$

$$\hat{C} = [\hat{C}_1 \quad \dots \quad \hat{C}_T]$$

and the time-indexed representation of the generalised ANN can be written as:

$$n_{ht} = \omega_{0h} + \sum_{k=1}^{K} \omega_{kh} X_{kt}$$

$$A_{ht} = \frac{e^{n_{ht}} - e^{-n_{ht}}}{e^{n_{ht}} + e^{-n_{ht}}}$$

$$\hat{C}_t = \gamma_0 + \sum_{h=1}^{H} \gamma_h A_{ht} + \sum_{k=1}^{K} \delta_k X_{kt}$$

The only task left to complete is to train the ANN to produce acceptable outputs from input data (in other words, to determine appropriate values for the ω's, γ's, and δ's in the model). To do this, a training algorithm similar to that described by Aminian et al. (2006) is employed. First, all available data is divided into three sets: a *training set*, a *validation set*, and a *forecasting set*. The forecast set is used to test the performance of the ANN. For this study, the forecast set is pre-determined (for convenience, it is set to be the last 16 periods of the available data). The remaining periods are randomly divided between the training and validation sets with approximately 70% of the in-sample data allocated to the training set and the remaining 30% to the validation set. It should be noted that we feed the input data into the ANN one period at a time, but we need not do this sequentially. The random allocation of data will generate non-sequential training and validation sets which will be different each time an ANN is trained.[4] This adds informational heterogeneity to the study.

Denoting G as the number of periods in the training set and V as the number of periods in the validation set, the neural network is trained using *back-propagation* (BP), a common procedure to

[3] As noted by Hornik et al. (1989) and Hornik (1991), ANNs can approximate any functional relationship between inputs and outputs to an arbitrary precision provided there are a sufficient number of hidden layers in the model (hence, they are known as 'universal approximators'). Cybenko (1989), Hornik et al. (1990), and Barron (1993) note that ANNs with a single layer may also be universal approximators provided that the activation functions used in the model satisfy certain properties (namely, smoothness, which is common in sigmoid functions) and the number of neurons (H) is large enough. Note that models with a large number of non-linear neurons may take excessive amounts of computation time to fit to data. As a result, some researchers expand the number of hidden layers. (Zhang et al., 1998). This is not the case for this study.

[4] In this study, either a prescribed ANN structure is simulated multiple times, or multiple ANN structures are simulated simultaneously. In any single ANN simulation, the training set and validation set always differ from those in any other simulation due to the random allocation of the data. The forecast set, however, remains the same across simulations so that their performance can be compared.

determining the weights. In BP, initial values for the weights are first guessed.[5] The following iterative algorithm is then followed:

1. Using provided weights and the input data from the training set, the ANN produces output estimates.

2. The estimation error is calculated for each period in the training set ($E_g = C_g - \hat{C}_g$), and a measure of fitness proportional to the mean squared error (MSE) is computed:

$$\frac{1}{2}\sum_{g=1}^{G} E_g^2 \propto \frac{1}{G}\sum_{g=1}^{G} E_g^2 = \text{training set MSE}$$

3. The weights are then updated with the goal of reducing estimation errors (thus improving the model's fit to the training set data).[6] Using the results from the training set, a set of adjustments are computed:[7]:

$$\Delta\omega_{kh} = -\mu \frac{\partial\left(\frac{1}{2}\sum_{g=1}^{G} E_g^2\right)}{\partial(\omega_{kh})}$$

$$\Delta\gamma_h = -\mu \frac{\partial\left(\frac{1}{2}\sum_{g=1}^{G} E_g^2\right)}{\partial(\gamma_h)}$$

$$\Delta\delta_k = -\mu \frac{\partial\left(\frac{1}{2}\sum_{g=1}^{G} E_g^2\right)}{\partial(\delta_k)}$$

   The parameter μ is scale parameter that controls the rate at which the ANN learns. This parameter must be chosen with care as a μ too large can result in excessive imprecision while a μ too small can take a great deal of computing time or result in convergence to a poor-fitting local optimum. In this study, μ = 0.01. Once the adjustments are found, a new value for each weight is derived by adding the corresponding adjustment to the weight's current value ($\omega_{kh}' = \omega_{kh} + \Delta\omega_{kh}$; $\gamma_h' = \gamma_h + \Delta\gamma_h$; $\delta_k' = \delta_k + \Delta\delta_k$).

4. After the weights are adjusted, the algorithm repeats at step 1.

The training set MSE will fall with each iteration. If we allow the BP algorithm to operate until the MSE of the training set is minimized, the ANN will 'over-fit' this data. In other words, the ANN will memorize the training patterns and fail to learn the underlying relationship between inputs and outputs, resulting in poor performance. To avoid this, we use the validation set data to refine training process. We compute how well the ANN fits the validation set during step 2 of each iteration:

$$\frac{1}{2}\sum_{v=1}^{V} E_v^2 \propto \frac{1}{V}\sum_{v=1}^{V} E_v^2 = \text{validation set MSE where } E_v = C_v - \hat{C}_v$$

We then allow the weights to be updated until the MSE of the validation set is minimized, at which time the training algorithm is stopped.[8] While we may not get an ANN that can fit all the in-sample data perfectly, out-of-sample forecasting is improved.

---

[5] Random starting weights are chosen. Choosing appropriate initial weights is important as it is possible for the BP algorithm to converge to local optima which produce poorly-performing ANNs. To reduce these occurrences, 500 starting points are randomly selected for each ANN and the BP algorithm is performed. The starting point that produces the best fit to the validation set is used. Note that this procedure does not identify a starting point associated with a global optimum.

[6] See Beltratti et al. (1996) or Warner and Misra (1996) for a more detailed description of the weight updating process for the basic BP algorithm.

[7] With the tanh activation function, it can be shown: $\Delta\omega_{kh} = \mu\gamma_h \sum_{g=1}^{G}\left(E_g\left(1-A_{hg}^2\right)X_{kg}\right)$, $\Delta\gamma_h = \mu \sum_{g=1}^{G}\left(E_g A_{hg}\right)$, $\Delta\delta_k = \mu \sum_{g=1}^{G}\left(E_g X_{kg}\right)$.

Once an ANN is trained, it is used to produce output estimates for the forecast set. Denoting F as the number of periods in the forecast set, the associated errors and MSE are then computed to evaluate the model's out-of-sample fit:

$$\frac{1}{F}\sum_{f=1}^{F} E_f^2 = \text{forecast set MSE where } E_f = C_f - \hat{C}_f$$

Because the forecast set is the same across all ANNs, we can compare the performance of ANNs with different topologies or ANNs with training and validation sets.

In the 1990s and 2000s, research demonstrated that ANNs can perform well in reproducing patterns seen in several types of financial and economic data, and as such are a valuable tool to use for forecasting.[9] It was also shown, however, that they do not outperform traditional methods consistently. As a result, researchers tend to evaluate the usefulness of ANNs on a case-by-case basis. For aggregate consumption patterns, one particular ANN study, done by Church and Curram (1996) for the United Kingdom, finds that although ANNs are more flexible in the types of variables which can be included and do not require a large number of data points to produce useful forecasts, they do not outperform standard econometric approaches significantly. The study presented below seeks to find the value of ANNs in generating aggregate consumption patterns for New Zealand (with the intention of using them to form behavioural rules for social simulation models), but also attempts to enhance their potential by adding diversity and social influence.

Several studies in the forecasting literature note that imperfect, heterogeneous forecasts can be combined to produce a 'social forecast' which is more accurate than any of its components. In nature, as well, evidence suggests that ecosystem diversity aids in system stability[10], which implies that the aggregated predictions of more heterogeneous communities may be more resilient to environmental shocks. Provided the method for combining forecasts is wisely chosen, the estimated patterns from a large variety of ANNs may be able to capture aggregate consumption patterns in New Zealand with improved precision. Evaluating different combination methods, and the social mechanism that may produce them, is done via simulation experiments.

To combine the forecasts of individual ANNs (each trained using the method above) into an aggregate forecast, we assign each ANN a weight and compute the weighted sum of their outputs. Denoting J as the total number of ANNs that we wish to combine and θ_j as a weight assigned to an individual simulation, this is:

$$\bar{C}_f = \sum_{j=1}^{J} \theta_j \hat{C}_{jf}$$

$$\frac{1}{F}\sum_{f=1}^{F} \bar{E}_f^2 = \text{aggregate forecast set MSE where } \bar{E}_f = C_f - \bar{C}_f$$

where $\bar{C}_f$ denotes the social forecast for period, f, in the forecast set. There is a large literature on the optimal combination of forecasts to draw upon when choosing θ_j (see Clemen (1989) and de Menezes et al. (2000) for a review). Four approaches in particular will be used in the analysis below:

---

[8] To induce efficiency in running the program, the BP algorithm operates until reductions in the MSE of the validation are sufficiently small (less than 1E-6 in this study).

[9] Zhang et al. (1998), Vellido et al. (1999), and Kourentzes and Crone (2010) review a variety of neural network applications and describe several of the main advantages and shortcomings of this approach.

[10] See McCann (2000) or Ives & Carpenter (2007) for an overview of the ecological relationship between diversity and stability.

1. *The simple average*. This approach weights all ANN forecasts equally: $\theta_j = 1/J$. We can think of this as the set-up in an egalitarian social environment where each individuals perspective is given equal value.

2. *Best overall performers*. This scheme assumes that ANNs that are good at overall in-sample forecasting will also be good at out-of-sample forecasting. The MSE for the entire training phase (training set + validation set) is constructed for each ANN:

$$E_{j\tau} = C_\tau - \hat{C}_{j\tau};\ \tau = 1...(G+V)$$

$$MSE_j^{TP} = \frac{1}{G+V}\sum_{\tau=G+V}^{G+V} E_{j\tau}^2$$

   Two weighting regimes are considered: one in which the ANN with the lowest MSE_j^{TP} receives all of the weight and one in which the top 25% of ANNs (i.e. the $0.25\times J$ ANNs with the lowest MSE_j^{TP}s) are equally weighted ($\theta_j = 1/(0.25\times J)$ if in the top 25%, else $\theta_j = 0$). This can be thought of as a social environment where only the predictions of "experts" matter (i.e. the existence of an intelligentsia). Note this scheme can result in ANNs with poor in-sample performance having no input into the combined forecast, even though these ANNs may have valuable information to contribute.

3. *Error-based weighting*. This approach was first developed for combining two forecasting methods by Bates and Granger (1969), and later described for multiple methods by Newbold and Granger (1974). ANNs with lower in-sample MSEs are given a relatively higher weight in constructing the aggregate:

$$E_{j\tau} = C_\tau - \hat{C}_{j\tau}$$

$$\theta_j = \left[\left(\sum_{\tau=G+V-w}^{G+V} E_{j\tau}^2\right)^{-1}\right] \Bigg/ \left[\sum_{i=1}^{J}\left(\left(\sum_{\tau=G+V-w}^{G+V} E_{i\tau}^2\right)^{-1}\right)\right]$$

   Note that the w most 'recent' periods in the non-forecasting set (the training and validation sets combined) are used to create the weights. If we wish to consider all of the available non-forecasting set data, we can set $w = G + V - 1$. A low w indicates that success in matching the most current data is important to producing an accurate aggregate forecast. Unlike scheme 2 above, this scheme allows all methods to receive some weight. However, the weights are biased towards good in-sample performers. Everyone's opinion matters, but those considered to be 'experts' receive more credence.

4. *Period-by-period outperformance*. De Menezes et al. (2000) note that assigning weights to each model based on how *frequently* it outperforms other models can produce combined forecasts with increased predictive accuracy. Unlike the combination methods above, how well a model performs during individual periods is scrutinized as opposed to overall performance. Often, a Bayesian approach is adopted to derive these weights from prior distributions (see, for example, Bunn (1975)). In a less rigorous approach with a similar flavour, the method used here establishes weights by scoring each ANN a point for each period in the non-forecast set that it is a "top performer". Formally:

   a. Select a parameter σ, $0 < \sigma < 1$, which represents the cut-off value for top performer status (e.g. if $\sigma = 10\%$, a top performer is among the 10% of the population with the lowest absolute error).

   b. For each ANN, compute how often the ANN was a top performer in the non-forecast set as defined by σ:

$$P_j = \frac{1}{G+V}\sum_{\tau=1}^{G+V} s_{j\tau}$$

where:

$$s_{j\tau} = \begin{cases} 1 \text{ if j is one of the } (\sigma \times J) \text{ ANNs with the lowest absolute error in period } \tau \\ 0 \text{ otherwise} \end{cases}$$

and P_i is the ANN's total average point accumulation.

c. Normalise the weights to sum to 1:

$$\theta_j = \frac{P_j}{\sum_{i=1}^{J} P_i}$$

As a starting case, σ = 25%. Note that ANNs that are most *often* successful relative to the other available ANNs will receive higher weight regardless of their accuracy; the *size* of the estimation error is not taken into account in this combination scheme. This is a social system favouring tradition. Those who have performed well most of the time will hold a great deal of influence, despite a few severe mistakes.

While there are several other[11] methods for forming combined forecasts described in the forecast literature, the four methods mentioned here are common and simple to implement for a large number of forecasts. The model described above is programmed into MATLAB. All simulations produce J = 15,000 ANNs. While a large J is desirable, increasing J beyond 15,000 lengthens computation time considerably. In addition to combining the individual forecasts using the methods described in the previous section, the program searches for the ANN which best matches out-of-sample data (an 'optimal' ANN).

3. Simulations and Results

A. Data

Data used in this study includes the growth rate of final private consumption expenditures per worker (c_t, the main variable of interest), the growth rate of GDP per worker (y_t, which serves as a proxy for income and overall economic performance), the point change in the unemployment rate (u_t, which captures labour market dynamics), the point change in the money market interest rate (r_t, which captures changes in the incentive to save), the CPI inflation rate (p_t, which captures the cost of living) and the point change in the nominal effective exchange rate (q_t, which captures incentives to purchase goods from abroad). These variables, commonly seen in both theoretical and empirical business cycle studies, are specific to the household sector (the largest sector in the economy) and represent the set of information that an average member of the population can readily access. A subset of these variables also appears in the study by Church and Curram (1996). All data pertains to the New Zealand economy, 1992q1 – 2011q3 (see Figure 3). Data for interest rates and exchange rates are sourced from the International Monetary Fund (2011) while all other data are sourced from the OECD (2011).

[Figure 3 here]

For the ANN models in the following experiments, the input variables in any period, t, include c_{t-1}, y_t, u_t, r_t, p_t and q_t. The output variable is c_t. As noted above, the final 16 periods of the sample

[11] One particular alternative approach involves using OLS to derive the optimal weights. This method, initially suggested by Granger and Ramanathan (1984), has been shown by Hashem (1997) to improve the accuracy of combined ANN forecasts specifically. In the algorithm described above, however, it is difficult to produce reasonable OLS estimates since many of the ANNs generate correlated outputs and there is no guarantee that two ANNs in the sample will not produce identical patterns (i.e. be linearly dependent). Although there are procedures for correcting this potential multicolinearlity, as shown by Hashem (1997), they can be complicated to implement when the number of models is large (as it is in this study).

(2007q4 – 2011q3) are reserved to be the forecast set[12] and the remaining 62 periods are randomly sampled into either the training set or the validation set for each individual ANN (G + V = 62).

B. Benchmark

Two key properties of ANNs are that they are flexible in structure and non-linear in nature. It will be useful to evaluate the performance of ANNs by comparing their results to that of a rigid, linear econometric model. To do this, the following linear benchmark model will be employed:

$$c_t=\alpha_0+ \alpha_1 c_{t-1}+ \alpha_2 y_t+ \alpha_3 u_t+ \alpha_4 r_t+ \alpha_5 p_t+ \alpha_6 q_t+ \varepsilon_t$$

where α_k, k = 0…6, are model coefficients (estimated via OLS using all data not in the forecast set) and ε_t is an error term (assumed to be i.i.d.). Over the forecast set, this model produces a mean squared error of 0.456. Results from the experiments in this study will be presented as percentage deviation from this value.

Note that we can think of the linear model as a restricted version of the ANN model described above (with all ω's and γ's forced to zero) which over-fits the in-sample data. As such, comparing the mean squared errors of the simulated ANNs to that of the linear benchmark describes how incorporating non-linearity, learning, and heterogeneity (in both data and structure) improves the production of out-of-sample patterns. There is no reason, however, to expect this particular linear benchmark model to provide the best possible econometric forecasts given its naivety. In practice, we can choose any linear model that we wish; provided we structure the inputs to the neural network accordingly, the linear framework will be relatable to the ANN framework. Exploring how ANNs perform compared to more complex linear models is left for future work.

C. Homogeneous linear network structure

We now use the heterogeneous ANN model with social aggregation to produce predictions for the forecast set. Let us first consider an ANN topology with no non-linear neurons activated (i.e. H = 0, only the 'bottom level' in Figure 2 is active). Individual ANNs do not differ structurally; all heterogeneity in the model occurs from the random allocation of training and validation sets. Results measuring the out-of-sample fit of each aggregation method relative to the benchmark model are reported in Table 1.

[Table 1 here]

The model produced an ANN that can generate an MSE over the forecast set data which is 69% *lower* than that produced by the linear benchmark model (the 'best' ANN at out-of-sample prediction of all 15,000 simulated). This outcome suggests ANNs can generate substantial gains; a result contrasting that of Church and Curram (1996). However, this best-performer is found after the fact. In practice, out-of-sample data is often not available at the time we need to produce forecasts; as such, we may not be able to find the ANN producing this good result.

Turning to the aggregation methods described above (which rely solely on in-sample data), the results in Table 1 suggest that the simple average of all ANNs, the average of the top 25% in-sample performers, and the error-based weighting method perform as well as, if not mildly better than, the linear benchmark; a result supporting the weak benefits of ANNs found in Church and Curram (1996). The outperformance weighting method, however, produces markedly more accurate predictions. Looking at the best in-sample performer (the *expert*) and using their ANN alone for out-of-sample forecasting produces the lowest MSE compared to the other aggregation methods. The performance of

[12] Because this period includes the most recent severe economic downturn, it offers a stringent testing ground for any forecasting model.

the expert exceeds that of the outperformance weighting method by a small amount, however (by 5.26% of the linear benchmark MSE).

Three central results emerge from this experiment. First, a diverse set of ANNs can be used to reproduce consumption patterns in New Zealand with improved accuracy. Second, ample improvements in pattern reproduction are achievable when the *frequency* of success is accounted for in combining forecasts. A social system that favours historical success, and disregards the severity of past errors, produces this. Finally, looking for an 'expert' and giving full weight to their advice produces the best forecasts (when ANNs are linear). The improvements made by the expert, however, are marginal. It should also be noted that this expert is not the best *out-of-sample* forecaster; the best out-of-sample performer trained an ANN that was less successful at matching in-sample data. This implies that there are optimal and second-best ways to the divide the in-sample data between the training and validation sets (which is what produces the heterogeneous ANNs).

D. Homogeneous non-linear network structure

Next, non-linearity is added. Table 2 reports the performance of ANNs with fully-connected inputs and $H > 0$ non-linear nodes. It is assumed that every ANN is homogeneous in structure. As in the previous experiment, individual ANNs differ only by the randomly selected training and validation sets used for training.

[Table 2 here]

The results in Table 2 show that adding non-linearity can increase the accuracy of *individual* ANNs at generating out-of-sample patterns. For example, with $H = 1$ non-linear neurons, the best out-of-sample performer produces an MSE that is 81% lower than the linear benchmark; an impressive improvement upon the best linear ANN in the previous section (which was 69% more accurate than the benchmark). This outcome further substantiates the benefits of ANN algorithms over traditional methods: non-linearity can produce more precise consumption patterns. Again, however, the best out-of-sample performer cannot be identified with in-sample data only.

As above, the simple average of all ANNs, the average of the top 25% in-sample performers, and the error-based method are mildly more accurate than the linear benchmark. Further, the outperformance aggregation method generates substantial improvements for a low number of non-linear neurons. While this aggregation method performed better when there were no non-linear neurons, the reduction in accuracy is rather small (7.02% of the linear model's MSE).

Unlike the previous experiment, however, the best in-sample performer no longer reasonably produces out-of-sample patterns. It is likely this occurs because the non-linear ANN structure of the 'expert' allowed them to excel at learning patterns in their own training set which differ substantially from those in the forecast set. These results suggest it is better to rely on the combined information of many than the expertise of a single forecasting method when non-linearity is included. As before, that information should be combined according to the frequency of success and not the severity of errors.

We can induce additional diversity by making inputs to non-linear neurons semi-connected (done by randomly forcing selected ω-weights to zero[13]). In these simulations, the number of non-linear neurons is identical for each ANN and ANNs continue to differ by their randomly selected training and validation sets. Simulation results are reported in Table 3.

[Table 3 here]

[13] Note that it is possible for some non-linear neurons to be completely severed from input data, thus rendering the neuron inactive.

The results from this experiment make the same implications as those in the experiment above when inputs were fully connected: (1) the simulation produces best out-of-sample forecasters that significantly improve upon the linear model; (2) the simple average of all ANNs, the average of the top 25% in-sample performers, and the error-based method outperform the linear benchmark mildly; (3) the best in-sample performer no longer produces accurate forecasts consistently; and (4) the outperformance aggregation method continues to perform well. These outcomes indicate that the ANN framework is robust to diversity of input information.

E. Heterogeneous non-linear network structure

To further evaluate the power of heterogeneity in producing reliable forecasts, we can add extra diversity by giving each agent a different ANN topology. For this exercise, the number of non-linear neurons for each ANN is assigned randomly (done by forcing randomly selected γ-weights to zero). Both fully-connected inputs and semi-connected inputs are considered. To allow for differing degrees of non-linearity, both a low number and a high number of potential non-linear neurons is simulated (maximum H = 5 and maximum H = 10 respectively). Results are reported in Table 4.

[Table 4 here]

As in previous experiments, we can find a best out-of-sample forecaster in the ANN framework with structural heterogeneity who produces patterns much more accurately than the linear benchmark. The simple average of all ANNs, the average of the top 25% in-sample performers, and the error-based method continue to perform as well as if not mildly better than the linear benchmark. However, the 'expert' at in-sample forecasting predicts out-of-sample patterns quite poorly. Also, the outperformance aggregation method only produces gains when inputs are semi-connected and the number of potential non-linear neurons is low. Although these combination methods had successes in earlier experiments, they should be used with caution in more complex settings.

F. Aside – Using out-of-sample results to select aggregation method parameters

The experiments in the previous sections used arbitrarily chosen parameter values for w (the number of past periods considered in the error-based aggregation method, = 62 above) and σ (the cut-off value to be a top performer in the outperformance aggregation method, = 25% above). With computational simulation, we can use out-of-sample data to evaluate our choices for these parameters. After ANNs have been trained, we attempt to determine values for w and σ that produce a social forecast with a minimal MSE over the forecast set. For the error-based weighting method, we simply calculate the weights associated with various values of w (w = 5, 10, 15, 20, 30, 40 and 50) and evaluate the accuracy of the resulting social forecasts. For the outperformance method, an optimising algorithm is used to determine the optimal σ (= σ^*). Table 5 reports simulation results for each of the ANN structures described in parts (C) and (D) above.

For the error-based weighting method, w = 15 produces aggregated forecasts with MSEs that are 37% to 40% lower than the linear benchmark; a striking improvement upon the mild success of the error-based weighting method in the previous experiments (w = 62). This outcome does not depend on the linearity of the ANNs nor on input connectedness. Note that w = 10 and w = 20 produce MSEs of a similar accuracy (less than 1% less accurate than w = 15 in many cases). Non-linear ANNs with varying degrees of complexity (H = 1:5) also produce similar MSEs. These results imply the value of w ought to be fairly low (i.e. forgetfulness has benefits – the social structure aggregating the forecasts should focus on performance during the most recent periods), but there is a range of choices for w and H which can perform equally well.

In the outperformance weighting method, a fairly large σ^* can produce substantially more accurate predictions than the linear benchmark (with MSEs that are 32% to 40% lower). It should be

noted that, in some cases, the improvements that a large σ achieves over a smaller σ are mild.[14] In other cases, they are not. Specifically, the simulation results show that a larger σ produces better results when the number of non-linear neurons (H) is large. In other words, expanding the class of top performers is important if information is processed with a complex ANN. Note, however, that the most successful models this study involved a low *H*, hence our uninformed choice of σ yielded fairly reasonable estimates.

[Table 5 here]

4. Discussion and Concluding Remarks

The experiments above highlight the benefits of using artificial neural networks to predict aggregate consumption patterns in New Zealand. If we have access to out-of-sample data, we can find a trained ANN with considerably lower forecast errors than a simple linear econometric model. This is possible for varying degrees of non-linearity in the ANN and connectedness of inputs to the output. Although the linear benchmark is naively chosen[15], this result supplements those of Church and Curram (1996). However, out-of-sample data is typically not available and this 'best out-of-sample performer' cannot be identified.

With access to in-sample data only, combining heterogeneous out-of-sample estimates to produce a 'social forecast' improves accuracy. Finding the best in-sample performer (or 'expert') and relying on their predictions is best when ANNs are linear, but not for other ANN topologies. With non-linearity, weighting forecasts according to period-by-period outperformance (frequency of success determines a forecaster's weight) produces substantial accuracy improvements. Other aggregation methods (simple average, top 25% in-sample performers, error-based weighting) produce milder improvements, but improvements nonetheless. The error-based weighting method can be improved if only recent periods are considered in the weighting process (i.e. the distant past is forgotten).

Connecting these results to the ACE literature, we can think of the above model as a rather crude agent-based model in which a community of individuals come together and pool their private understanding of their environment to form social knowledge. This occurs via a social mechanism (venerating experts, building intelligentsia, punishing (or not punishing) severe errors by shunning, etc.). However, the exercise above focuses only on information processing and not on actual choice; the forecasts can be thought of as a rule that households should follow when determining spending, but they are never obligated to follow the rule in the model. In a more complete agent-based model, agents would have to choose how much to consume (subject to a budget constraint) based on the information they acquire. This is left for future work. Further, information above is aggregated globally.[16] Local interaction is preferred in agent-based models. Allowing agents to create socially-influenced forecasts using only a few of their neighbours' forecasts would incorporate this, and is also left for future work.

By engineering a household sector where information processing is used by consumers to guide their spending patterns, we can perhaps better understand consumers that think like real people (and less like *homo economicus*). We add realism to this sector by allowing for heterogeneity and social interaction. The New Zealand case above suggest that using artificial neural networks to make forecasts, and combining these forecast in a certain way, is one possible approach to use when building such models.

[14] For example, for linear ANNs, σ = 25% produced a forecast set MSE that is 39% lower than the linear benchmark model. For this ANN structure, σ* = 36% produces a forecast set MSE that is 40% lower than the linear benchmark (a 1% improvement). For ANNs with 1 non-linear neuron and fully connected inputs, σ = 25% produced a forecast set MSE that is 32% lower than the linear benchmark model. For this ANN structure, σ* = 52% produces a forecast set MSE that is 39% lower than the linear benchmark (a 7% improvement).

[15] The linear benchmark model specified in this study is a useful starting point. Further comparisons to more complicated benchmark models can be performed and is left for future work.

[16] It is as if a single agent collects the forecasts from all 15,000 others to produce a single social forecast.

References

Arifovic, J. (1994). Genetic algorithm learning and the cobweb model. *Journal of Economic Dynamics and Control*, 18(1), 3-28.

Aminian, F., Suarez, E. D., Aminian, M. & Walz, D. T. (2006). Forecasting economic data with neural networks. *Computational Economics*, 28(1), 71-88.

Barron, A. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3), 930-945.

Bates, J. M. & Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Quarterly*, 20(4), 451-468.

Beltratti, A., Margarita, S., & Terna, P. (1996). *Neural Networks for Economics and Financial Modelling*. London: International Thomson Computer Press.

Bunn, D. W. (1975). A Bayesian approach to the linear combination of forecasts. *Operational Research Quarterly*, 26(2), 325-329.

Church, K. B. & Curram, S. P. (1996). Forecasting consumers' expenditure: A comparison between econometric and neural network models. *International Journal of Forecasting*, 12(2), 255-267.

Clemen, R. T. (1989). Combining forecasts: A review and notated bibliography. *International Journal of Forecasting*, 5(4), 559- 583.

Cooper, J. C. B. (1999). Artificial neural networks versus multivariate statistics: An application from economics. *Journal of Applied Statistics*, 26(8), 909-921.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4), 303-314.

De Menezes, L. M., Bunn, D. W. & Taylor, J. W. (2000). Review of guidelines for the use of combined forecasts. *European Journal of Operational Research*, 120(1), 190-204.

Detienne, K. B., Detienne, D. H. & Joshi, S. A. (2003), Neural networks as statistical tools for business researchers. *Organizational Research Methods*, 6(2), 236-265.

Epstein, J. M. (1999), Agent-based computational models and generative social science. *Complexity*, 4(5), 41-60.

Gaffeo, E., Delli Gatti, D., Desiderio, S., & Gallegati, M. (2008). Adaptive microfoundations for emergent macroeconomics. *Universita Degli Studi Di Trento Working Paper No. 2/2008.*

Gatti, D., Di Guilmi, C., Gaffeo, E., Giulioni, G., Gallegati, M., & Palestrini, A. (2005). A new approach to business fluctuations: heterogeneous interacting agents, scaling laws and financial fragility. *Journal of Economic Behavior and Organization*, 56(4), 489-512.

Gonzalez, S. (2000). Neural networks for macroeconomic forecasting: A complementary approach to linear regression models. *Department of Finance Canada Working Paper 2000-07*.

Granger, C. W. J. & Ramanathan, R. (1984). Improved methods of combining forecasts. *Journal of Forecasting*, 3(2), 197-204.

Hashem, S. (1997). Optimal linear combinations of neural networks. *Neural Networks*, 10(4), 599-614.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks,* 4(2), 251-257.

Hornik, K., Stinchcombe, M. & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366.

Hornik, K., Stinchcombe, M. & White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5), 551-560.

International Monetary Fund [IMF] (2011). e-Library Data. Available at http://elibrary-data.imf.org/.

Ives, A. R. & Carpenter, S. R. (2007). Stability and diversity of ecosystems. *Science*, 317(58), 58-62.

Jain, A. K. & Mao, J. (1996). Artificial neural networks: A tutorial. *Computer*, 29(3), 31-44.

Kourentzes, N. & Crone, S. (2010). Advances in forecasting with artificial neural networks. *Lancaster University Management School Working Paper 2010/023*.

Kuan, C.-M. & White, H. (1994). Artificial neural networks: An econometric perspective. *Econometric Reviews*, 13(1), 1-91.

McCann, K. S. (2000). The diversity-stability debate. *Nature*, 405, 228-233.

Mirowski, P. (2007). Markets come to bits: Evolution, computation and markomata in economic

science. *Journal of Economic Behavior and Organization*, 63(2), 209-242.
Newbold, P. & Granger, C. W. J. (1974). Experience with forecasting univariate time series and the combination of forecasts. *Journal of the Royal Statistical Society, Series A (General)*, 137(2), 131-165.
Organisation for Economic Co-operation and Development [OECD] (2011). OECD.StatExtracts. Available at http://stats.oecd.org.
Raberto, M., Teglio, A. & Cincotti, S. (2008). Integrating real and financial markets in an agent-based economic model: An application to monetary policy design. *Computational Economics*, 32(1-2), 147-162.
Tesfatsion, L. (2002). Agent-based computational economics: Growing economies from the bottom up. *Artificial Life*, 8(1), 55-82.
Tesfatsion, L. (2005). Agent-based computational modeling and macroeconomics. *ISU Economic Report 05023: July 2005.*
Tesfatsion, L. & Judd, K. (eds) (2006). *Handbook of Computational Economics, V.2: Agent-based Computational Economics*. Amsterdam: Elsevier.
Vellido, A., Lisboa, P. J. G. & Vaughan, J. (1999). Neural networks in business: A survey of applications (1992 – 1998). *Expert Systems with Applications*, 17(1), 51-70.
Warner, B. & Misra, M. (1996). Understanding neural networks as statistical tools. *The American Statistician*, 50(4), 284-293.
Zhang, G., Patuwo, B. E. & Hu, M. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14(1), 35-62.

Tables and Figures

Figure 1– A perceptron.

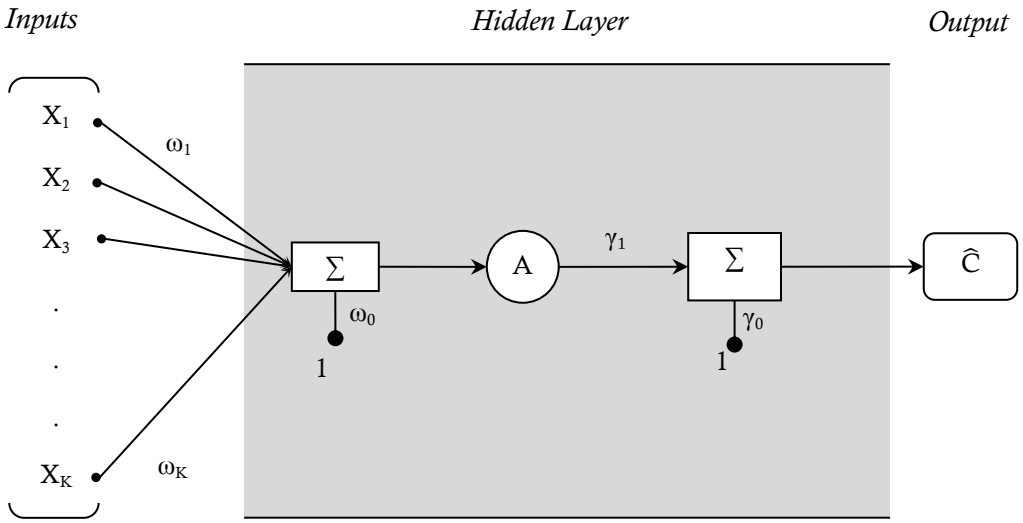


Figure 2 – A simple ANN with 1 linear and H non-linear neurons.

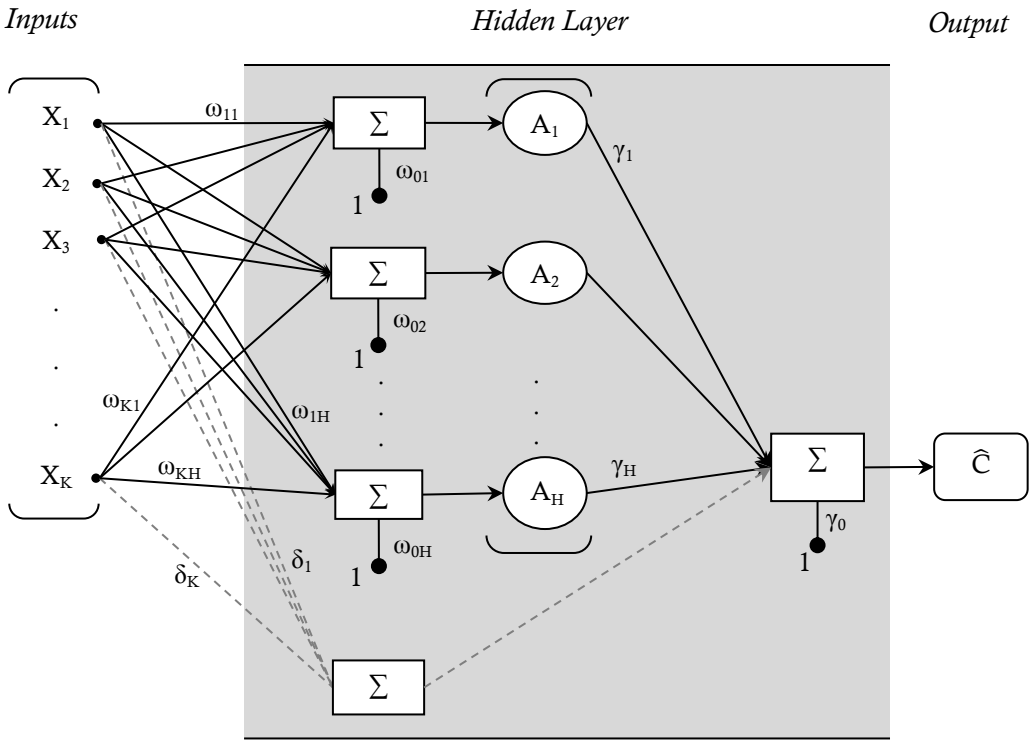
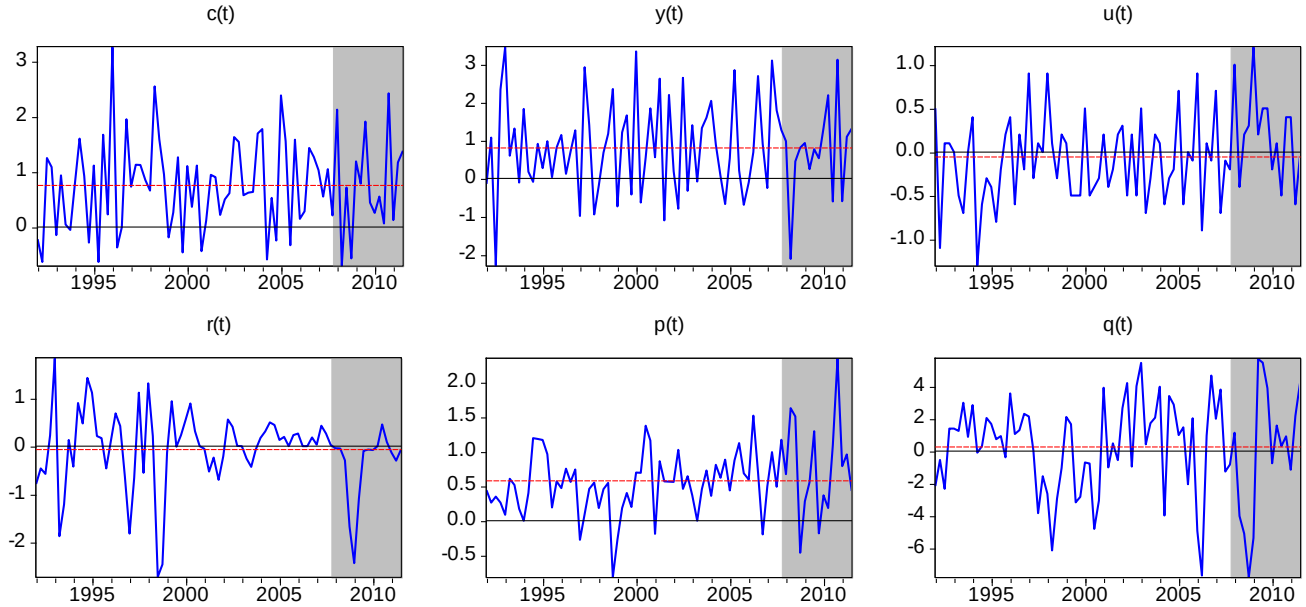


Figure 3 – Empirical regularities of the New Zealand economy (1992q1-2011q3).



Source: IMF (2011), OECD (2011).
Notes: Forecast set (shaded). Sample average (---).

Table 1 – Mean squared forecast error of ANNs with no non-linear neurons (measured as % deviation from linear benchmark).

Aggregation Method	Forecast Set MSE
Simple Average	-7.98
Best In-sample Performer	**-44.30**
Top 25% In-sample Performers	-12.94
Error-based Weighting (w=62)	-11.62
Period-by-period Outperformance (σ = 25%)	-39.04
Best Out-of-sample Performer	*-68.86*

Notes: Best performing aggregation method is bolded.

Table 2 – Mean squared forecast error of homogeneous non-linear ANNs with fully-connected inputs (measured as % deviation from linear benchmark).

	Forecast Set MSE					
Aggregation Method	H = 1	H = 2	H = 3	H = 4	H = 5	H = 10
Simple Average	-10.53	-10.53	1.54	-6.80	-2.63	-3.95
Best In-sample Performer	-0.66	-3.29	140.13	45.39	56.80	70.61
Top 25% In-sample Performers	-10.53	-9.21	-8.99	-8.99	-7.24	1.75
Error-based Weighting (w=62)	-10.96	-11.84	-7.24	-8.77	-8.33	-3.07
Period-by-period Outperformance (σ = 25%)	**-32.02**	-23.90	-20.39	1.97	-21.27	203.95
Best Out-of-sample Performer	*-80.70*	*-68.20*	*-72.37*	*-67.98*	*-68.20*	*-46.49*

Notes: Best performing aggregation method is bolded.

Table 3 – Mean squared forecast error of homogeneous non-linear ANNs with semi-connected inputs (measured as % deviation from linear benchmark).

	Forecast Set MSE					
Aggregation Method	H = 1	H = 2	H = 3	H = 4	H = 5	H = 10
Simple Average	-0.66	-5.48	-5.48	-11.62	-6.80	0.44
Best In-sample Performer	-24.34	8.99	275.44	142.76	19.52	298.25
Top 25% In-sample Performers	-7.24	-9.87	-9.87	-14.69	-12.50	-6.36
Error-based Weighting (w=62)	-6.80	-9.65	-9.43	-12.72	-11.62	-3.73
Period-by-period Outperformance (σ = 25%)	**-35.53**	-32.68	-29.82	-21.93	-19.96	18.64
Best Out-of-sample Performer	*-67.32*	*-69.52*	*-71.71*	*-67.54*	*-63.82*	*-66.45*

Notes: Best performing aggregation method is bolded.

Table 4 – Mean squared forecast error of heterogeneous non-linear ANNs (measured as % deviation from linear benchmark).

	Forecast Set MSE			
	Fully-connected		Semi-connected	
Aggregation Method	Max H = 5	Max H = 10	Max H = 5	Max H = 10
Simple Average	-8.55	-8.99	-2.63	0.88
Best In-sample Performer	160.53	92.32	56.80	223.90
Top 25% In-sample Performers	-4.39	-4.17	-7.24	-2.41
Error-based Weighting (w=62)	-6.36	-5.26	-8.33	-3.29
Period-by-period Outperformance (σ = 25%)	27.63	211.40	**-21.27**	18.64
Best Out-of-sample Performer	*-75.23*	*-66.67*	*-68.20*	*-69.08*

Notes: Best-performing aggregation method is bolded.

Table 5 – Optimised mean squared forecast error of fully- and semi-connected ANNs (measured as % deviation from linear benchmark) for error-based and period-performance aggregation regimes.

(a) Linear network structure

	Forecast Set MSE
Error-based Weighting	
w = 5	-28.51
w = 10	-38.82
w = 15	-39.25
w = 20	-38.16
w = 30	-32.02
w = 40	-13.16
w = 50	-18.42
Period-by-period Outperformance ($\sigma = \sigma*$)	**-40.13**
$\sigma*$	36%

Notes: Best performing aggregation method is bolded. Neural networks that improve upon the benchmark model by more than 35% are highlighted.

(b) Non-linear network structure with fully-connected inputs

	Forecast Set MSE					
	H = 1	H = 2	H = 3	H = 4	H = 5	H = 10
Error-based Weighting						
w = 5	-28.29	-28.73	-27.19	-26.97	-26.32	-20.61
w = 10	-39.04	-39.04	-37.50	-37.28	-36.84	-33.77
w = 15	-39.69	**-39.91**	-38.60	-38.38	-37.50	-34.65
w = 20	-38.60	-39.04	-37.72	-37.72	-36.62	-34.43
w = 30	-32.68	-33.77	-32.24	-32.89	-30.70	-29.82
w = 40	-12.94	-14.04	-10.09	-11.40	-10.75	-7.02
w = 50	-18.86	-20.18	-16.89	-18.64	-16.67	-14.47
Period-by-period Outperformance ($\sigma = \sigma*$)	-39.25	-38.60	-37.28	-36.84	-35.53	-31.36
$\sigma*$	52%	58%	56%	65%	56%	80%

Notes: Best performing aggregation method is bolded. Neural networks that improve upon the benchmark model by more than 35% are highlighted.

(c) Non-linear network structure with semi-connected inputs

	Forecast Set MSE					
	H = 1	H = 2	H = 3	H = 4	H = 5	H = 10
Error-based Weighting						
w = 5	-27.19	-28.95	-28.07	-28.95	-28.95	-25.00
w = 10	-37.72	-38.60	-38.82	-39.04	-38.16	-36.40
w = 15	-38.38	-39.25	-39.47	**-39.91**	-39.04	-37.28
w = 20	-37.06	-38.16	-38.16	-39.04	-38.16	-36.40
w = 30	-30.26	-32.02	-32.02	-33.99	-33.33	-30.70
w = 40	-8.99	-11.62	-11.62	-14.47	-13.82	-7.02
w = 50	-14.91	-17.76	-17.76	-20.61	-20.18	-14.69
Period-by-period Outperformance ($\sigma = \sigma*$)	-38.38	-38.82	-38.38	-38.60	-37.72	-35.96
$\sigma*$	42%	48%	52%	59%	60%	65%

Notes: Best performing aggregation method is bolded. Neural networks that improve upon the benchmark model by more than 35% are highlighted.