Why Hammerstein-Type Block Models Are So Efficient: Case Study of Financial Econometrics

Thongchai Dumrongpokaphan[1], Afshin Gholamy[2]
Vladik Kreinovich[2], and Hoang Phuong Nguyen[3]

[1]Department of Mathematics, Chiang Mai University, Thailand, tcd43@hotmail.com
[2]University of Texas at El Paso, El Paso, Texas 79968, USA
afshingholamy@gmail.com, vladik@utep.edu
[3]Division Informatics, Math-Informatics Faculty, Thang Long University, Hanoi, Vietnam, nhphuong2008@gmail.com

1. Linear Models And Need to Go Beyond Them

- In the 1st approximation, the dynamics of an economic system can be often described by a linear model.
- In a linear model, the values $y_1(t), \ldots, y_n(t)$ of the desired quantities at moment t linearly depend:
  - on the values of these quantities at the previous moments of time, and
  - on the values of related quantities $x_1(t), \ldots, x_m(t)$ at the current and previous moments of time:

$$y_i(t) = \sum_{j=1}^{n} \sum_{s=1}^{S} C_{ijs} \cdot y_j(t-s) + \sum_{p=1}^{m} \sum_{s=0}^{S} D_{ips} \cdot x_p(t-s) + y_{i0}.$$

- In practice, however, many real-life processes are non-linear.
- It is desirable to take this non-linearity into account.

2. Hammerstein-Type Block Models for Nonlinear Dynamics Are Efficient in Econometrics

- In many econometric applications:
  - the most accurate and the most efficient models turned out to be models
  - which in control theory are known as *Hammerstein-type block models*.
- These models combine linear dynamic equations with non-linear static transformations.
- In such models, the transition from moment t to the next one consists of several sequential transformations.
- Some are linear dynamical transformations.
- Others are *static* non-linear transformations, that take into account only the current values of the quantities.

3. A Toy Example of a Block Model

- To illustrate the idea of a Hammerstein-type block model, let us consider the simplest case, when:
  - the state of the system is described by a single quantity y_1,
  - the state $y_1(t)$ at moment t is uniquely determined only by its previous state $y_1(t-1)$, and
  - no other quantities affect the dynamics.
- In the linear approximation, the dynamics of such a system is described by a linear dynamic equation

$$y_1(t) = C_{111} \cdot y_1(t-1) + y_{10}.$$

- The simplest possible non-linearity here will be an additional term which is quadratic in $y_1(t)$:

$$y_1(t) = C_{111} \cdot y_1(t-1) + c \cdot (y_1(t-1))^2 + y_{10}.$$

4. A Toy Example (cont-d)

- In terms of an auxiliary variable $s(t) \stackrel{\text{def}}{=} (y_1(t))^2$, the above system can be described in terms of two blocks:
  - a linear dynamical block described by a linear dynamic equation
  $$y_1(t) = C_{111} \cdot y_1(t-1) + c \cdot s(t-1) + y_{10}, \text{ and}$$
  - a nonlinear block described by a non-linear static transformation $s(t) = (y(t))^2$.
- In econometrics, non-quadratic transformations are often used: e.g., logarithms and exponential functions.
- They transform a multiplicative relation $z = x \cdot y$ between quantities into a linear relation between logs:
$$\ln(z) = \ln(x) + \ln(y).$$

5. Formulation of the Problem

- In many cases, a non-linear dynamical system can be represented in the Hammerstein-type block form.
- However, the question remains why necessarily such models often work the best in econometrics.
- Indeed, there are many other techniques for describing non-linear dynamical systems, such as:
  – Wiener models, in which $y_i(t)$ are described as Taylor series in terms of $y_j(t-s)$ and $x_p(t-s)$,
  – models that describe the dynamics of wavelet coefficients,
  – models that formulate the non-linear dynamics in terms of fuzzy rules, etc.
- In this talk, we explain why such models are efficient in econometrics, especially in financial econometrics.

6. Specifics of Computations Related to Econometrics, Especially to Financial Econometrics

- In many economics-related problems, it is important:
  - not only to predict future values of the corresponding quantities,
  - but also to predict them as fast as possible.
- This need for speed is easy to explain; for example:
  - an investor who is the first to finish computation of the future stock price
  - will have an advantage of knowing in what direction this price will go.
- If his/her computations show that the price will go up, the investor will buy the stock at the current price.
- Thus, the investor will gain a lot.

7. Computations in Econometrics (cont-d)

- If the computations show that the price will go down, the investor will sell his/her stock at the current price.
- Thus, the investor will avoid losing money.
- Similarly, an investor who is the first to predict the change in the ratio of two currencies will gain a lot.
- In all these cases, fast computations are extremely important.
- Thus, the nonlinear models that we use in these predictions must be the fastest to compute.

8. How Can We Speed Up Computations: Need for Parallel Computations

- If a task takes a lot of time for a single person, a natural way to speed it up is:
  - to have someone else help,
  - so that several people can perform this task in parallel.
- Similarly,
  - if a task takes too much time on a single computer processor,
  - a natural way to speed it up is to have several processors work in parallel on different subtasks.

9. Need to Consider the Simplest Possible Computational Tasks for Each Processor

- The overall computation time is determined by the time during which each processor finishes its task; so:
  - to make the overall computations as fast as possible,
  - it is necessary to make the elementary tasks assigned to each processor as fast as possible,
  - thus, as simple as possible.
- Each computational task involves processing numbers.
- We are talking about the transition from linear to nonlinear models.
- So, it makes sense to consider linear versus nonlinear transformations.
- Clearly, linear transformations are much faster than nonlinear ones.

10. Need for the Simplest Tasks (cont-d)

- However, if we only use linear transformations, then we only get linear models.
- To take nonlinearity into account, we need to have some nonlinear transformations as well.
- A nonlinear transformation can mean:
  - having one single input number and transforming it into another,
  - having two input numbers and applying a nonlinear transformation to these two numbers,
  - it can mean having three input numbers, etc.
- Clearly, in general, the fewer numbers we process, the faster the data processing.

11. Need for the Simplest Tasks (cont-d)

- Thus, to make computations as fast as possible:
  - it is desirable to restrict ourselves to the fastest possible nonlinear transformations,
  - namely, the transformations of one number into one number.
- Thus, to make computations as fast as possible, it is desirable to make sure that:
  - on each computation stage, each processor performs one of the fastest possible transformations,
  - either a linear transformation or the simplest possible nonlinear transformation $y = f(x)$.

12. Need to Minimize the Number of Computational Stages

- We agreed how to minimize the computation time needed to perform each computation stage.
- Now, the overall computation time is determined by the number of computational stages.
- To minimize the overall computation time, we thus need to minimize the overall number of such stages.
- In principle, we can have all kinds of nonlinearities in economic systems; thus:
  - we need to select the smallest number of computational stages
  - that would still allow us to consider all possible nonlinearities.
- How many stages do we need?

13. One Stage Is Not Sufficient

- One stage is clearly not enough.
- Indeed, during one single stage, we can compute:
  - either a linear function $Y = c_0 + \sum_{i=1}^{N} c_i \cdot X_i$ of the inputs $X_1, \ldots, X_N$,
  - or a nonlinear function of one of these inputs

$$Y = f(X_i),$$

  - but not, e.g., a simple nonlinear function of two inputs, such as $Y = X_1 \cdot X_2$.

14. What About Two Stages?

- Can we use two stages?
- If both stages are linear, all we get is a composition of two linear functions which is also linear.
- Similarly, if both stages are nonlinear, all we get is compositions of functions of one variable.
- This will also be a function of one variable.
- Thus, we need to consider two different stages.
- Let us first consider the case when:
  - on the first stage we use nonlinear transformations
  $$Y_i = f_i(X_i),$$
  - and on the second stage, we use a linear transformation $Y = \sum_{i=1}^{N} c_i \cdot Y_i + c_0$.

15. Two Stages (cont-d)

- Then, we get the expression $Y = \sum\limits_{i=1}^{N} c_i \cdot f_i(X_i) + c_0$.
- For this expression, the partial derivative $\dfrac{\partial Y}{\partial X_1} = c_1 \cdot f_1'(X_1)$ does not depend on X_2, so thus,

$$\frac{\partial^2 Y}{\partial X_1 \partial X_2} = 0.$$

- This means that we cannot describe the product $Y = X_1 \cdot X_2$ for which $\dfrac{\partial^2 Y}{\partial X_1 \partial X_2} = 1$.
- But what if:
  - we use linear transformation on the first stage, getting $Z = \sum\limits_{i=1}^{N} c_i \cdot X_i + c_0$, and then
  - we apply a nonlinear transformation $Y = f(Z)$.

16. Two Stages (cont-d)

- This results in $Y(X_1, X_2, \ldots) = f\left(\sum\limits_{i=1}^{N} c_i \cdot X_i + c_0\right)$.
- Y's level set $\{(X_1, X_2, \ldots) : Y(X_1, X_2, \ldots) = \text{const}\}$ is $\sum\limits_{i=1}^{N} c_i \cdot X_i = \text{const}$, i.e., a plane.
- In the 2-D case $N = 2$, it is a straight line.
- For multiplication, the level sets are hyperbolas $X_1 \cdot X_2 = \text{const}$ – and not straight lines.
- Thus, a 2-stage function cannot describe or approximate multiplication $Y = X_1 \cdot X_2$.
- So, two computational stages are not sufficient, we need at least three.

17. Are Three Computational Stages Sufficient?

- An arbitrary function can be represented as a Fourier transform:

$$Y(X_1,\ldots,X_N) \approx \sum_k c_k \cdot \sin\left(\omega_{k1}\cdot X_1 + \ldots + \omega_{kN}\cdot X_N + \omega_{k0}\right).$$

- The right-hand side expression can be easily computed in three simple computational stages:
  - first, we have a linear stage

$$Z_k = \omega_{k1}\cdot X_1 + \ldots + \omega_{kN}\cdot X_N + \omega_{k0},$$

  - then, we have a nonlinear stage at which we compute the values $Y_k = \sin(Z_k)$, and
  - finally, we have another linear stage $Y = \sum\limits_k c_k \cdot Y_k$.
- Thus, three stages are indeed sufficient.
- So, in our computations, we should use three stages, e.g., linear-nonlinear-linear as above.

18. Relation to Traditional 3-Layer Neural Networks

- The same three computational stages form the basis of the traditional 3-layer neural networks.
- On the first stage, we compute a linear combination of the inputs $Z_k = \sum\limits_{i=1}^{N} w_{ki} \cdot X_i - w_{k0}$.
- Then, we apply a nonlinear transformation $Y_k = s_0(Z_k)$.
- The corresponding *activation function* $s_0(z)$ usually has:
  - either the form $s_0(z) = \dfrac{1}{1 + \exp(-z)}$
  - or the rectified linear form $s_0(z) = \max(z, 0)$.
- Finally, a linear combination of the values Y_k is computed: $Y = \sum\limits_{k=1}^{K} W_k \cdot Y_k - W_0$.

19. Relation to Neural Networks (cont-d)

- In neural networks, the first two stages are usually merged into a single stage:

$$Y_k = s_0\left(\sum_{i=1}^{N} w_{ki}\cdot X_i - w_{k0}\right).$$

- The reason is that in the biological neurons, these 2 stages are performed within the same neuron:
  - first, the signals X_i from different neurons come together, forming a linear combination

$$Z_k = \sum_{i=1}^{N} w_{ki}\cdot X_i - w_{k0},$$

  - and then, within the same neuron, the nonlinear transformation $Y_k = s_0(Z_k)$ is applied.
- Note: it is sometimes beneficial to use different functions $Y_k = s_k(Z_k)$ for different k.

20. How This Applies to Non-Linear Dynamics

- In non-linear dynamics, to predict each of the desired quantities $y_i(t)$, we need to take into account:
  - the previous values $y_j(t-s)$ of the quantities $y_1, \ldots, y_n$, and
  - the current and previous values $x_p(t-s)$ of the related quantities $x_1, \ldots, x_m$.
- In the 3-stage computation scheme, prediction of $y_i(t)$ consists of the following three stages.
- First, there is a linear stage:

$$\ell_{ik}(t) \stackrel{\text{def}}{=} \sum_{j=1}^{n} \sum_{s=1}^{S} w_{ikjs} \cdot y_j(t-s) + \sum_{p=1}^{m} \sum_{s=0}^{S} v_{ikps} \cdot x_p(t-s) - w_{ik0}.$$

- Then, there is a non-linear stage $a_{ik}(t) \stackrel{\text{def}}{=} s_{ik}(\ell_{ik}(t))$.

21. Non-Linear Dynamics (cont-d)

- Finally, a linear stage $y_i(t) = \sum_{k=1}^{K} W_{ik} \cdot a_{ik}(t) - W_{i0}$.
- We thus have the Hammerstein-type block structure:
  - a linear dynamical part is combined with
  - static transformations, in which we only process values corresponding to the same moment t.
- Thus, *the desire to perform computations as fast as possible leads to the Hammerstein-type block models.*
- We have therefore explained the efficiency of such models in econometrics.
- As we have mentioned, 3-layer models of the above type are universal approximators.
- So, Hammesterin-type models can approximate any nonlinear dynamics with any given accuracy.

22. Acknowledgments

- This work was supported by Chiang Mai University.
- It was also partially supported by the US National Science Foundation via grant HRD-1242122.
- The authors are greatly thankful to Hung T. Nguyen for valuable discussions.