Learning Ordinary Differential Equations for Macroeconomic Modelling

Zhivko Georgiev
Department of Computer Science
University of York
York YO10 5GH, United Kingdom
Email: zgeor@alumni.york.ac.uk

Dimitar Kazakov
Department of Computer Science
University of York
York YO10 5GH, United Kingdom
Email: dimitar.kazakov@york.ac.uk

***Abstract*—This article describes an empirical approach to the macroeconomic modelling of the Euro zone. Data for the period 1971–2007 has been used to learn systems of ordinary differential equations (ODE) linking inflation, real interest and output growth. The *equation discovery* algorithm LAGRAMGE was used in conjunction with a grammar defining a potentially large range of possible parametric equations. The coefficients of each equation are automatically fitted on the training data and the ones with the lowest error rates returned as a result. We have added a tool for out-of-sample error evaluation to the in-sample evaluation built in LAGRAMGE. The paper compares the performance of ODE models to previous work on the learning of ordinary equations for the same purpose.**

I. Introduction

When building mathematical models of economic data, the approaches often centre on choosing the 'right' parametric equation or system of equations, based on a mixture of theoretical reasoning and empirical observations. The task is then reduced to fitting the parameters of the selected equations in order to minimise an error criterion, such as the mean squared error (MSE). An example of this is when each equation is produced as the result of a linear or a (specific) non-linear regression on the training data.

Equation discovery [9] is a machine learning approach in which the learning bias is much weaker in comparison, and a potentially wide range of possible equations are considered to find the best fit of the data provided. The LAGRAMGE system used here is an equation discovery tool in which a context-free grammar combining variables with mathematical operators and functions is used to produce parametric equations whose coefficients are subsequently fitted. The grammar may define a potentially infinite number of equations, which in practice requires the use of a threshold limiting the complexity of each equation considered by the search.

The LAGRAMGE system has already been successfully applied in the field of ecology [1], [10]. More recently, it was also used for macroeconomic modelling on the case of the Euro zone, resulting in complex non-linear models with several equations showing high fidelity [5]. This article uses the same data set as a starting point, but tests another instrument in the LAGRAMGE toolkit, namely the ability to learn ordinary differential equations (ODE). Quarterly data about inflation, interest and output is used in a number of experiments to evaluate this approach. The resulting models provide a link between monetary policy (in the form of interest) and the state of the economy as measured by its output and rate of inflation.

II. Brief history of the Euro Area

In order to put into context the data used for learning macroeconomic models of inflation, real interest and output growth in the Euro Area a summary of the union's history is required. Since the 1970s there have been three main initiatives towards European monetary union and four oil crises. This section aims at providing a very brief overview of the events which occurred in the studied period from 1971 to 2007.[1] It is mainly focused on the events with a direct impact on the monetary policy in the Euro Area.

The first attempt was the *"Snake in the Tunnel"* plan launched in the first quarter of 1971. Its founding countries were Belgium, Luxembourg, France, Italy, the Netherlands, and West Germany. It involved setting the maximum difference between two currencies from the union to be $\pm 2.25\%$, while at the same time their maximum margin against the USD was $\pm 4.5\%$. No single currency was implemented as a result of the plan. While the plan was in motion the number of members of the union increased and decreased on a regular basis. For example, the United Kingdom joined the union and then was forced to leave only six weeks later. It was followed by Denmark and Italy—both having joined after initial agreement was signed—due to certain economic factors as summarised by Pinsky and Kvasnicka [8]. 1973 marked the beginning of the *first oil crisis*. In just six months, until March 1974, the prices of oil almost quadrupled from \$3 to nearly \$12 per barrel. In 1979 a second oil crisis shook up the prices of crude oil, with prices more than doubling in the span of a year. Such major events have a great impact on the world's economy, and are deemed to create recessions [3], [4]. By 1979 the first attempt at increased European market cooperation had failed. This marked an age of great instability in the Euro area which is reflected in the data for that period.

[1] This period has been selected to allow for direct comparison of results with existing work, as the main thrust of this study is on methodology.

978-1-4799-7560-0/15 \$31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.133

IEEE computer society

The next attempt was the *European Monetary System* (EMS), which became established in 1979. The founding members of the union were Belgium, Denmark, France, Germany, Ireland, Luxembourg, and the Netherlands followed by a restricted membership of the UK. A major difference when comparing it to the "Snake in the Tunnel" plan was the introduction of the European Central Unit (ECU) and European Monetary Cooperation Fund (EMCF), the former being a virtual currency managed by the European Central Bank (ECB), while the latter is an administrative structure for increased monetary policy coordination. This second attempt at cooperation is marked with considerably more success than the "Snake in the tunnel" plan. As McNamara [6] points out, EMS has proven to be quite stable and has paved the way for the European Monetary Union.

What followed was the *European Monetary Union (EMU)*, which was planned out in the 1989 *Delors* report as a three-stage process. Shortly after the 1990 wave of *oil price shocks* [11], the 1992 Maastricht Treaty was signed, marking the beginning of the first stage of the Delors plan. In the context of this research, it is important to mention that two of the main Maastricht treaty goals are the convergence of economic criteria imposed on inflation and interest rates. 1992 is also associated with a period of spikes in interest rates and short term instability in the area, which became known as the European Exchange Rate Mechanism (ERM) crisis and is best remembered for Britain's dramatic exit from the mechanism. Stage 2 started in 1994, further increasing monetary cooperation through the "Stability and Growth Pact". The stage was completed in 1999 with the creation of the *Euro zone*. The members of the union adopted a single monetary policy, along with the *Euro*, which was scheduled to substitute the member currencies in a three year period.

In conclusion, it was important to show the major events in the Euro area so the data could be better understood. The period described above contains a wide range of events from energy crises to increased cooperation and change in monetary policies. These variations in the macroeconomics of the Euro area create a great challenge for any algorithm trying to model the complex relationships underlying the data.

III. The Data

The dataset used consists of quarterly data of the annual rates of inflation π, output growth y, and the real interest r for the Euro area in the period 1971Q1–2007Q1. Additional columns containing the same variables, but with a number of time lags $(t-1, t-2, \dots)$ were also added to allow the discovery of equations making reference to such lagged variables. Time was also included as an explicit variable (using a step of 1 to represent a move from one quarterly entry to the next). The dataset was divided into data used for model estimation (i.e., training sample/dataset, comprising all readings in the period 1971Q1–2005Q1), and a test dataset 2005Q2–2007Q1, which was used to evaluate the models on previously unseen data ('out-of-sample evaluation'). In line with other studies [5], the real interest rates have been assumed to be equal to the nominal interest rates minus the realised (i.e., actual) inflation one period ahead, the latter being used as a proxy for the expected inflation rate.

IV. Learning ODE with LAGRAMGE

When using equation discovery systems for modelling, it is essential to provide background knowledge [10], which in the case of LAGRAMGE takes the form of a context free grammar. The grammar consisting of Rules 1, 3 and 4 below which was first proposed by Kazakov and Tsenova [5] is used here as a starting point of our study. It has been shown to successfully generate accurate systems of non-linear ordinary equations (which are acquired one at a time). Here these previous results are replicated, then the same grammar is used for learning ODEs.

A number of not fully differentiable functions have been considered for inclusion in the grammar. One function was eventually added on the basis of its performance. The grammar was extended with the addition of the $step(c,\ t)$ function as defined in Table 1 and incorporated in the grammar through Rules 2 and 5. This discontinuous function is equal to its

```
double step(double c, double t) {
          return c>t ? t : 0 ;
}
```

Fig. 1. Definition of `step` function

second argument t when its value is below the threshold c; otherwise it is equal to zero. For any equation generated by the grammar, the constants are varied using the provided range and step (e.g. $[-10, 10]$ and 0.01) until the best fit is found. The search algorithm explores all successive derivations starting with the grammar axiom E up to a set depth, which here was limited to 7.

$$
\begin{aligned}
E &\rightarrow E + Term | Term && (1)\\
E &\rightarrow ST + E | ST * E && (2)\\
Term &\rightarrow const[-10:0.01:10] \\
&\rightarrow const[-10:0.01:10] * V \\
&\rightarrow const[-10:0.01:10] * sin(LT) * sin(LT) \\
&\rightarrow const[-10:0.01:10] * V * sin(LT) \\
&\rightarrow const[-10:0.01:10] * sin(LT) && (3)\\
LT &\rightarrow const[-10:0.01:10] * V + \\
&\quad + const[-10:0.01:10] && (4)\\
ST &\rightarrow step(const[0:0.001:10], Term) \\
&\rightarrow step(const[0:0.001:10], V) && (5)
\end{aligned}
$$

For example, one possible derivation of depth 3 is: $E \rightarrow E + Term \rightarrow Term + Term \rightarrow const * V + const$. It is obtained by replacing every nonterminal symbol with the right hand side of a matching grammar rule until the result does not contain nonterminals. Every occurrence of the symbol V in the resulting equation is then replaced by an input variable (corresponding to a column in the table containing the input data) and the coefficients $const$ fitted to obtain an optimal fit.

LAGRAMGE returns the top 15 equations minimising the means squared error (MSE) on the training data. These were then tested on out of sample data and the best were selected.

As pointed out by Brown [2] time-step choice is a common problem when modelling dynamic systems with components unstable in a limited time range. The publication discusses the range of appropriate values for multistep numerical integration methods. Based on the study, and some prior experimentation, the time step was chosen to be 0.05 for all equations. As a result of running all experiments with the same time-step value, it is possible to create a system of ODEs as a combination of equations for π, y and r where each was obtained with a different grammar.

V. Forecasting with Differential Equations

A gap in the literature was identified, namely the absence of convenient forecasting tools based on ordinary differential equations. Although LAGRAMGE can find the differential equations for the dynamic processes underlying the data, no clear method of forecasting using the ODE models is provided. The problem is interesting because unlike forecasting with ordinary equations, differential equations require integration to create projections about the future. Numerous integration methods have been developed throughout the years. Some of the classes that they can fall into are: implicit or explicit, single or multistep, numerical or analytical. Numerical integration methods are of interest for this project since they are easily implementable in any general purpose programming language. In contrast, not all functions have analytic solutions and even though advances have been made in the field of analytical integration a general solver does not exist. The downside of numerical integration methods, as compared to analytical method, is that they are all associated with an error. Higher order integration methods have better accuracy, however they are more complex for implementation and computation. Two methods of numerical integration were reviewed, namely Runge-Kutta and explicit Adams-Bashforth.

The main difference between implicit and explicit integration methods is the values they required for calculating the value of y_{n+1} coupled with their stability. Explicit integration methods require the current state of the system in order to find a solution as shown in Equation 6.

$$Y(t + \delta t) = F(Y(t)) \tag{6}$$

On the other hand, implicit methods require the current system state and the state at a further point, as shown in Equation 7. In turn, this requires an extra computation and also makes these methods harder to implement. The advantage of implicit methods over explicit is the fact that they are more stable when working with stiff systems. In fact, Nevanlinna and Sipilä [7] proved that there are no explicit methods which are A-stable. This property of integration methods shows whether a method converges to the true solution when small changes in the initial values occur.

$$G(Y(t), Y(t + \delta t)) = 0 \tag{7}$$

Runge-Kutta methods devised in the 1900s by Carl Runge and Martin Kutta is a family if explicit and implicit methods for numerical integration. The integration formula for the implicit method is shown on equation 8. As it could be seen the term y_{n+1} is present on the right hand side, thus it is implicitly defined. As a result, another method, such as Newton-Raphson, is required for solving the implicit equation. However, implicit methods allow for the specification of a reasonably sized time step in order to increase the accuracy.

$$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, \frac{1}{2}(y_n + y_{n+1})\right) \tag{8}$$

Equation 9 below shows an explicit Runge-Kutta 2 method. The method uses half steps towards the solution. This requires the derivatives to be calculable at half steps. This is possible with a system of equations, however it is not the output of lagramge is just a single equation. As a result the calculation of derivatives at midpoint is not possible unless all of the variables in the system have been modelled.

$$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}f(t_n, y_n)\right) \tag{9}$$

Adams-Bashfort is a family of explicit integration methods. Its second order method is a two-step linear method. Eq. 10 shows the mathematical representation of the method. This makes it a good candidate for integration method of choice because only full step derivatives are calculated. One will only require the differential equation discovered by LAGRAMGE along with data for two time lags back. Furthermore, the simplicity of the method provides an elegant and straightforward solution.

$$y_{n+2} = y_{n+1} + h(\frac{3}{2}(f(t_{n+1}, y_{n+1}) - \frac{1}{2}f(t_n, y_n))) \tag{10}$$

VI. Results

First we start by discovering ordinary non-linear equations using the initial grammar. The input variables explicitly present in the data (in the form of separate columns) include lagged values for π, y and r for several steps back in time. Time t itself is also included explicitly in the data, hence it could appear as one of the variables in the equations found. The results below reproduce the equations in Kazakov and Tsenova's article [5]. Equations 11, 12 and 13 model the inflation, output growth and real interest. Note how in the second term of Equation 11 defining the output growth, the combination of a sin function and explicit time results in a term with a periodic behaviour (with period $T = 2\pi/0.21 \approx 30$, which for this quarterly data corresponds to a cycle of 7.5 years). Similarly, the equation for the interest rate contains a term with a period of $2\pi/0.045$ quarters or 35 years approx.

$$\pi_t = 6.65\, sin(0.60y_{t-1} - 1.02)\, sin(0.02\pi_{t-1} - 0.04) \\ -0.62\, sin(-0.28\pi_{t-1} + 0.71)\, sin(0.21t - 1.09) \\ +0.11 + 0.96\pi_{t-1} \tag{11}$$

$$y_t = 10.66\, sin(0.56y_{t-1} - 2.02)\, sin(0.11y_{t-2} + 0.88)) \\ +8.84\, sin(0.11r_{t-2} + 1.20)\, sin(0.58y_{t-1} + 0.84) \\ +1.60 + 0.07y_{t-2} \tag{12}$$

$$r_t = 8.88\, sin(0.09r_{t-1} - 32.21)\, sin(0.05\pi_{t-1} + 14.18) \\ +23.14\, sin(0.01\pi_{t-2} + 0.01)\, sin(0.045t - 1.78) \\ +6.98 + 0.05y_{t-2} \tag{13}$$

Then we use the same grammar to discover ODE and the reported equations below are the ones that performed best on the out of sample dataset.

$$\frac{d\pi_t}{dt} = 0.78y_t - 10\pi_{t-2}sin(-0.23\pi_t + 1.90) \\ -0.84 - 9.24\pi_{t-1}\, sin(-0.25\pi_{t-1} + 5.16) \tag{14}$$

$$\frac{dy_t}{dt} = 10 + 8.15y_t\, sin(0.187413y_{t-2} + 0.92) \\ +0.22t\, sin(-0.07\pi_{t-2} - 0.05) - 10y_{t-2} \tag{15}$$

$$\frac{dr_t}{dt} = 1.08t\, sin(0.02\pi_t - 0.01) - 3.16 \\ +10r_t\, sin(0.14y_t + 1.11) - 10r_{t-1} \tag{16}$$

As we can see the models found are structurally simpler, however they still contain lagged variables. For the inflation π we see a little worsening in the root mean square error (RMSE) value (see Table 1), however output growth and real interest models yield better results.

Finally, the equations discovered after the initial grammar has been modified with the $step(c,\ t)$ function and all lagged variables removed from the dataset are reported below. In the resulting equations, all variables have a time index t, which has been omitted.

$$\frac{d\pi_t}{dt} = 0.57y + 0.49\pi.sin(0.27r + 3.35) \\ +0.84\pi.sin(0.77y + 0.70). \\ step(0.001,\ \pi.sin(0.31\pi - 3.21)) \tag{17}$$

$$\frac{dy_t}{dt} = 0.74y + 0.94y.sin(0.34\pi + 2.30) \\ +step(0.001, r.sin(0.43\pi - 1.70)). \\ step(0.001, \pi.sin(0.44y - 2.55)) \tag{18}$$

$$\frac{dr_t}{dt} = y + step(8.41, 0.60\pi.sin(0.50\pi + 3.15)) \\ +0.87r.sin(0.48y + 2.68). \\ step(0.33, \pi.sin(0.60\pi - 3.88)) \tag{19}$$

The complexity of these models has increased compared to previous models reported. However, the absence of lagged values in the data allows for a simpler data input. Furthermore, these models produced the best results in terms of RMSE.

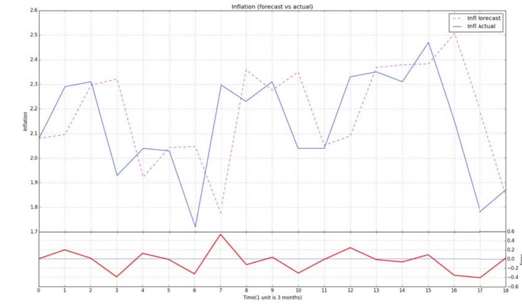

Fig. 2. One step ahead prediction of inflation with $step()$ function grammar model.

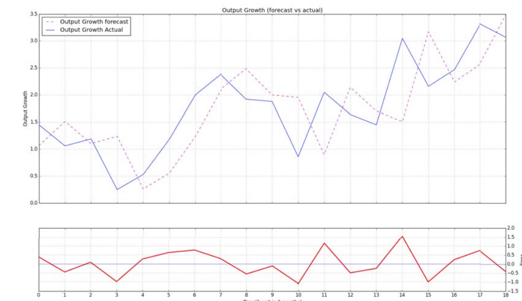

Fig. 3. One step ahead prediction of output growth with $step()$ function grammar model.

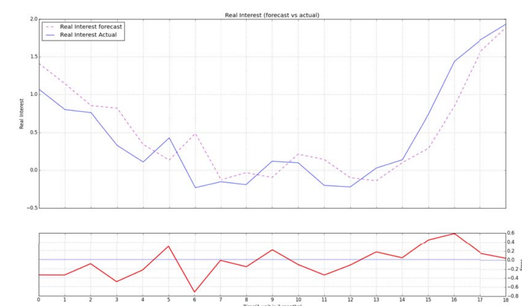

Fig. 4. One step ahead prediction of real interest with $step()$ function grammar model.

As we can see from graphs 2, 3 and 4, the forecast values (dashed red line) follow closely the actual values recorded for the period (continuous blue line). As with most of the regression models a lag is observed. The error graphs (subplot of every graph) show increase in the error of forecasting when sudden spikes appear.

VII. Further Use of Discontinuous Functions

There are several reasons for including discontinuous functions in the ODE grammar. For instance, the variables that are being modelled may be limited in their range, e.g. the economy

TABLE I
SUMMARY OF THE RESULTS

Name	Eqn	Var	In-Sample RMSE	Out-of-Sample RMSE
Ord.	11	π	0.38	0.23
ODE sin	14	π	0.359	0.247
ODE step	17	π	0.723	**0.215**
Ord.	12	y	0.83	1.13
ODE sin	15	y	0.701	0.851
ODE step	18	y	0.359	**0.588**
Ord.	13	r	0.56	0.64
ODE sin	16	r	0.701	0.311
ODE step	19	r	0.741	**0.299**

output cannot be negative. We have therefore experimented with several other such functions, as defined below:

```
double step2(double t0, double t) {
return t0>t ? 0 : 1;
}

double upperlimit(double c, double t) {
return c>t ? t : c ;
}

double interval(double c, double delta,
                                double t) {
% delta > 0
return t>c && t<(c+delta) ? t : 0 ;
}
```

Fig. 5. More discontinuous functions

Thus, `step2` is zero for $t < t0$ and 1 otherwise. The function `upperlimit` caps the value of its second argument. Finally, the third argument of `interval` is returned as its value when within a given interval, or zero is returned otherwise. LAGRAMGE was then used with each of these functions in turn (in place of the `step` function). While some of the equations approached the ones reported in the previous section in terms of out of sample accuracy, none of them was better. Nevertheless, the discontinuous functions described in this section may be easier to interpret, and one ultimately needs to make a choice between a focus on accuracy alone, and the desire to offer a plausible interpretation of the model described by the equations.

VIII. CONCLUSION

The ODE models provided a good match to ordinary equations. The results from the table below show that ODEs performed better for all three of the dependent variables when the `step` function was embedded in the domain knowledge. In case of the initial grammar, ODEs provided better results for output growth and real interest, with the inflation result being just slightly worse.

The benefit of using ODEs is that their forecasts range from comparable to considerably better than ordinary equations even when the learning dataset is simplified. On the down side, implementing the forecasting procedure is not as trivial as with ordinary equations. This study reviewed a good few integration methods and concluded that linear multistep methods like Adams-Bashforth 2 are of good fit to compute the forecast values using LAGRAMGE ODE models. As the tool produces a single model, rather than a complete system, Adams-Bashforth was a good starting point. The lower order method also reduced the number of initial values needed. Finally, the difference between in-sample and out-of-sample accuracy, both in absolute terms and in terms of the ranking of the best equations serve as a reminder that in any practical application an out-of-sample validation set should be used to select the best model.

REFERENCES

[1] Nataša Atanasova, Sašo Džeroski, Boris Kompare, Ljupčo Todorovski, and Gideon Gal. Automated discovery of a model for dinoflagellate dynamics. *Environmental Modelling & Software*, 26(5):658–668, 2011.

[2] RL Brown. Error behaviour of multistep methods applied to unstable differential systems. *Applied Mathematical Modelling*, 2(2):115–118, 1978.

[3] James D Hamilton. This is what happened to the oil price-macroeconomy relationship. *Journal of Monetary Economics*, 38(2):215–220, 1996.

[4] G John Ikenberry. The irony of state strength: comparative responses to the oil shocks in the 1970s. *International Organization*, 40(01):105–137, 1986.

[5] Dimitar Kazakov and Tsvetomira Tsenova. Equation discovery for macroeconomic modelling. In *Proc. of ICAART*, pages 318–323. INSTICC, 2009.

[6] Kathleen R McNamara. *The currency of ideas: monetary politics in the European Union*. Cambridge Univ Press, 1998.

[7] Olavi Nevanlinna and Aarne H Sipilä. A nonexistence theorem for explicit A-stable methods. *Mathematics of computation*, 28(128):1053–1056, 1974.

[8] Neil J Pinsky and Joseph G Kvasnicka. The European monetary system. *Economic Perspectives*, (Nov):3–10, 1979.

[9] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[10] Ljupčo Todorovski, Sašo Džeroski, Pat Langley, and Christopher Potter. Using equation discovery to revise an Earth ecosystem model of the carbon net production. *Ecological Modelling*, 170(2):141–154, 2003.

[11] Philip K Verleger. Understanding the 1990 oil crisis. *The Energy Journal*, pages 15–33, 1990.