A Practical, Accurate, Information Criterion for Nth Order Markov Processes

Sylvain Barde

Comput Econ (2017) 50:281–324
DOI 10.1007/s10614-016-9617-9

A Practical, Accurate, Information Criterion for Nth Order Markov Processes

Sylvain Barde[1,2]

Accepted: 5 August 2016 / Published online: 3 September 2016

Abstract The recent increase in the breath of computational methodologies has been matched with a corresponding increase in the difficulty of comparing the relative explanatory power of models from different methodological lineages. In order to help address this problem a Markovian information criterion (MIC) is developed that is analogous to the Akaike information criterion (AIC) in its theoretical derivation and yet can be applied to any model able to generate simulated or predicted data, regardless of its methodology. Both the AIC and proposed MIC rely on the Kullback–Leibler (KL) distance between model predictions and real data as a measure of prediction accuracy. Instead of using the maximum likelihood approach like the AIC, the proposed MIC relies instead on the literal interpretation of the KL distance as the inefficiency of compressing real data using modelled probabilities, and therefore uses the output of a universal compression algorithm to obtain an estimate of the KL distance. Several Monte Carlo tests are carried out in order to (a) confirm the performance of the algorithm and (b) evaluate the ability of the MIC to identify the true data-generating process from a set of alternative models.

Keywords AIC · Minimum description length · Markov process · Model selection

JEL Classification B41 · C15 · C52 · C63

1 Introduction

The rapid growth in computing power over the last couple of decades, combined with the development of user-friendly programming languages and an improvement of fun-

✉ Sylvain Barde
s.barde@kent.ac.uk

1 School of Economics, Keynes College, University of Kent, Canterbury CT2 7NP, UK

2 Observatoire Français des Conjonctures Economiques, Paris, France

damental statistical and algorithmic knowledge have lead to a widening of the range of the computational methods available to researchers, from formal modelling to estimation, calibration or simulation methodologies. While this multiplication of available methods has offered a greater modelling flexibility, allowing for the investigation of richer dynamics, complex systems, model switching, time varying parameters, etc., it has come at the cost of complicating the problem of comparing the predictions or performance of models from radically different methodological classes. Two recent examples of this, which are by no means exclusive, are the development of the dynamic stochastic general equilibrium (DSGE) approach in economics, and the increase in the popularity of what is generally referred to as agent-based modelling (ABM), which uses agent-level simulations as a method of modelling complex systems, and for which even the issue of bringing models to the empirical data can prove to be a problem.

Within the DSGE literature on model validation and comparison, one of the first to identify and address this problem in a direct and systematic manner is Schorfheide (2000), who introduces a loss function-based method for evaluating DSGE models. This is then complemented by the DSGE-VAR procedure of Del Negro and Schorfheide (2006) and Negro et al. (2007), which explicitly sets out to answer the question ‘How good is my DSGE model?’ (p. 28). The procedures gradually developed over time in this literature are summarised in the section on DSGE model evaluation of Negro and Schorfheide (2011), which outlines several methods for evaluating DSGE performance, such as posterior odds ratios, predictive checks and the use of VAR benchmarking.

Similar concerns relating to model evaluation and comparison also exist in the ABM literature, to the extent that two special journal issues have been published in order to identify and address them. Fagiolo et al. (2007), as part of the special issue on empirical validation in ABM of *Computational Economics*, provide a very good review of the existing practices and provide advice as to how to approach the problem of validating an agent-based simulation model. The most obvious example of the need for validation methodologies is the recent development of several large-scale agent-based frameworks that allow the investigation of key macroeconomic questions, such as Keynes/Schumpeter model developed in Dosi et al. (2010, 2013, 2015), which allows for analysis of fiscal and monetary policies, or the European-wide EURACE collaboration of Deissenberg et al. (2008), Van Der Hoog et al. (2008) and Holcombe et al. (2013), which aims to allow large-scale modelling of economic systems. As pointed out by Fagiolo and Roventini (2012), these would greatly benefit from being compared to the standard DSGE macroeconomic workhorse models. Nevertheless, as outlined by Dawid and Fagiolo (2008) in the introduction of the special issue of the *Journal of Economic Behavior and Organization* on adapting ABM for policy design, finding effective procedures for empirical testing, validation and comparison of such models is still very much an active field of research. Some of the recent developments in this regard are related to the approach suggested here, for example the state similarity measure of Marks (2010, 2013), which aims to measure the distance between two time-series vectors, as is the case here. Similarly, ongoing work by Lamperti (2015) also explores the possibility of comparing models on the basis of simulated data alone with an information based measure, using however a very different measurement approach. Finally, another approach of note is Fabretti (2014), which treats the simulated data from an implementation of the Kirman (1993) model as a Markov chain in order to estimate its parameters. While again the objective and methodology used are different from what is proposed here, the idea of mapping the simulated data to a Markov process is very similar in spirit.

The paper aims to contribute to this general issue of comparing different lineages of models by providing a proof-of-concept for a Markovian information criterion (MIC) that generalises the Akaike (1974) information criterion (AIC) to any class of model able to generate simulated data. Like the AIC, the proposed criterion is fundamentally an estimate of the Kullback and Leibler (1951) (KL) distance between two sets of probability densities. The AIC uses the maximised value of the likelihood function as an indirect estimate of the KL distance, however, this obviously requires the model to have a parametric likelihood function, which is no longer straightforward for many classes of modelling methodologies. The proposed criterion overcomes this problem by relying instead on the original data compression interpretation of the KL distance as the inefficiency resulting from compressing a data series using conditional probabilities that are an estimate or approximation of the true data generating process. This fundamental equivalence between data compression and information criteria has led to the emergence of what is known as the *minimum description length* (MDL) principle, which relies on the efficiency of data compression as a measure of the accuracy of a model's prediction. Grünewald (2007) provides a good introduction to the MDL principle and its general relation to more traditional information criteria, while Hansen and Yu (2001) explore the use of MDL within a model selection framework, concluding that "MDL provides an objective umbrella under which rather disparate approaches to statistical modelling can coexist and be compared" (Hansen and Yu 2001, p. 772).

The proposed methodology provides three key contributions compared to existing methods. The first is to provide a standardised measurement, as the procedure places all models on an equal footing, regardless of their numerical methodology or structure, by treating the simulated data they produce as the result of a Nth order Markov process, where the number of lags is chosen to capture the time dependency of the data. As pointed out by Rissanen (1986), Markov processes of arbitrary order form a large subclass (denoted FSMX) of finite-state machines (FSMs), i.e., systems where transitions are governed by a fixed, finite transition table. By mapping every model to be compared to its FSM representation and scoring these transition probabilities on the empirical data, the MIC is able to overcome differences in modelling methodologies and produce a standardised criterion for any model reducible to a Markov process. This is designed to take advantage of the fact, pointed out by Tauchen (1986a, b) and Kopecky and Suen (2010), that many economic variables and modelling approaches can in fact be approximated by a Markov process.[1] A second contribution is that the algorithm used to obtain the transition table possess a guaranteed optimal performance

[1] This is because, as was pointed out by Shannon (1948) at the very start of the data compression literature, "any stochastic process which produces a discrete sequence of symbols chosen from a finite set may be considered a discrete source" and modelled by a Markov process. As shown by Shannon using the example of the English language, even systems that are not strictly generated through a Markov process can be successfully approximated by one.

over Markov processes of arbitrary order, which as will be shown below, allows an accurate measurement for the MIC. Finally, the algorithm measures cross-entropy at the observation level, producing a vector which sums up to the MIC which enables the reliability of a measurement to be tested statistically.

Because purpose of the approach is to use the MIC to compare a set of models $\{M_1, M_2, \ldots, M_m\}$ against a fixed-size data set, it is important to also highlight the data snooping problem identified by White (2000) and the reality check procedures that must be carried out to avoid it. Essentially, because statistical tests always have a probability of type I error, repeated testing of a large (and possibly increasing) set of models on a fixed amount of data creates the risk of incorrectly selecting a model that is not truly the best model in the set. White (2000) therefore proposes a procedure that takes into account the size of the model comparison set $\{M_1, M_2, \ldots, M_m\}$ when testing for performance against a benchmark model M_0. A recent development in this literature is the model confidence set (MCS) methodology of Hansen et al. (2011), which differs from White's reality check in that it does not test against a benchmark model, but instead identifies the subset $\hat{\mathcal{M}}_{1-\alpha}$ of models in the set which cannot be distinguished from each other at significance level α. This is well suited to the model-specific vectors of scores produced by the MIC, therefore the MCS is included in the Monte Carlo analyses presented below.

The remainder of the paper is organised as follows. Section 2 first discusses the use of universal data compression as an empirical tool for evaluating prediction accuracy and details the theoretical properties of the MIC. A Monte Carlo analysis is then performed in Sect. 3 in order to compare the MIC against the AIC benchmark in an ARMA-ARCH setting and evaluate the criterion's performance. Section 4 discusses the use of the methodology in practical settings and Sect. 5 concludes.

2 The MIC: Motivation and Theoretical Properties

Before examining the information-theoretical motivation for the MIC methodology and the core properties that justify the choice of algorithms, it is important to first briefly clarify the terminology and notation that will be used throughout the paper.

First of all, we define a *prediction* as a conditional probability mass function over the discrete states a system can occupy, given knowledge of the system's history. A *model* is very loosely defined as any device that can produce a complete set of predictions, i.e., a prediction for every acceptable history. This is very similar to the loose definition adopted by Hansen et al. (2011) for their MCS procedure. Furthermore, no assumption is made on the quality of the predictions: a uniform distribution over the system's states is acceptable. Conceptually, this set of predictions corresponds to the state transition table of a FSMX or, equivalently, as the transition matrix of a Markov process. This loose definition of a model is intended to be very general: anything from personal belief systems to formal analytical models, as well as calibrated simulations or estimated econometric specifications are reducible to this class of processes, as all can provide a predictive density for future observations given the history of the system.

Regarding notation, the binary logarithm will be clearly identified as '$\log_2$', while the natural logarithm will be 'ln'. X_t is an unobserved, real-valued random variable describing the state of a system at time t and x_t its observed realisation. Data series are denoted as $x_1^t = \{x_1, x_2, \ldots, x_t\}$. $\underline{X}_t$, $\underline{x}_t$ and $\underline{x}_1^t$ are the discretised versions of the same variables, with r being the number of bits of resolution used for the discretisation and $\Omega = 2^r$ the resulting number of discrete states the system can occupy. Because of the binary discretisation used, $\underline{x}_t$ will refer to both the numerical value of the observation and the corresponding r-length binary string encoding it. When necessary, the kth bit of a given observation $\underline{x}_t$ will be identified as $\underline{x}_t\{k\}$. $P_{dgp}(\underline{X}_t | \underline{x}_1^{t-1})$ is the true probability distribution over the Ω states at time t conditional on the past realisations of the variable. $P_{M_i}(\underline{X}_t | \underline{x}_{t-L}^{t-1})$ is a corresponding conditional probability distribution predicted by a model M_i at the same time and over the same state space, using a limited number of lags L. Using the chain rule for conditional probabilities, $P(\underline{x}_1^t) = P(\underline{X}_t | \underline{x}_1^{t-1}) P(\underline{x}_1^{t-1})$, the model predictions and true conditional probabilities can be used recursively to build the overall probabilities for the series $P_{M_i}(\underline{x}_1^t)$ and $P_{dgp}(\underline{x}_1^t)$.

2.1 Information Criteria and Minimum Description Length

Given a model M_i, a reasonable metric for evaluating the accuracy of the overall prediction $P_{M_i}(x_1^t)$ with respect to $P_{dgp}(x_1^t)$ is the Kullback and Leibler (1951) (KL) distance measure between the two distributions, which was developed as an extension of the fundamental concept of information entropy introduced in Shannon (1948).

$$D_{KL}\left(P_{M_i}\left(x_1^t\right) \| P_{dgp}\left(x_1^t\right)\right) = E_{dgp}\left[\ln \frac{P_{dgp}(x_1^t)}{P_{M_i}(x_1^t)}\right]. \tag{1}$$

In terms of notation, the $E_{dgp}[\ldots]$ operator indicates that the expectation is taken with respect to the true distribution $P_{dgp}(x_1^t)$. The first obvious consequence of (1) is that the KL divergence D_{KL} is zero whenever $P_{M_i}(x_1^t) = P_{dgp}(x_1^t)$. As shown by Cover and Thomas (1991), by taking into account the strict concavity of the logarithm and applying Jensen's inequality to the expectation term in (1) one can show that the KL distance is strictly positive for $P_{M_i}(x_1^t) \neq P_{dgp}(x_1^t)$, making it a *strictly proper* scoring rule in the sense of Gneiting and Raftery (2007). This property underpins the use of the KL distance as a conceptual criterion for determining the accuracy of a model, as minimising the KL distance with respect to the choice of prediction model should theoretically lead to the identification of the true model.

While the KL distance is a desirable measure of accuracy in theory, it suffers from not being directly computable in practice, as this would require knowledge of P_{dgp}. The key insight of Akaike (1974) was to identify that it is possible to use the maximum likelihood estimation of the model M_i, to obtain an estimate of the following cross entropy, without requiring knowledge of the true distribution P_{dgp}:

$$E_{dgp}\left[\ln \frac{1}{P_{M_i}(x_1^t)}\right] = D_{KL}\left(P_{M_i}\left(x_1^t\right) \| P_{dgp}\left(x_1^t\right)\right) + E_{dgp}\left[\ln \frac{1}{P_{dgp}(x_1^t)}\right]. \tag{2}$$

Assuming that the model M_i uses a vector of κ_i parameters θ_i, and that $\hat{\theta}_i$ are the parameter values that maximise the likelihood $\mathcal{L}(\theta_i|x_1^t)$, Akaike (1974) showed the cross entropy between the data and the model can be estimated asymptotically by the following relation, directly leading to the classical definition of the AIC for a set of models:

$$\frac{\text{AIC}_i}{2} = E_{dgp}\left[\ln \frac{1}{P_{\hat{M}_i}(x_1^t)}\right] = -\ln\left[\mathcal{L}\left(\hat{\theta}_i \middle| x_1^t\right)\right] + \kappa_i. \tag{3}$$

The fact that (3) is not directly an estimate of the KL distance (1) but instead of the cross entropy (2) explains why Akaike (1974) recommends looking at the AIC differences between models, $\Delta\text{AIC}_{i,j} = \text{AIC}_i - \text{AIC}_j$, as this removes the model-independent Shannon entropy terms and keep only the relative KL distance $\Delta D_{KL}(P_{M_i}(x_1^t) \| P_{dgp}(x_1^t))_{i,j}$.

As emphasised by the MDL literature, the original interpretation of the KL distance (1) relates to the fundamental theoretical limits to compressibility of data. Given a discretised data series $\underset{\sqcup}{x}{}_1^t$, the binary Shannon entropy $-E_{dgp}[\log_2 P_{dgp}(\underset{\sqcup}{x}{}_1^t)]$ gives the number of bits below which the data series cannot be compressed without loss. Because the true probability over states of nature P_{dgp} is unknown, practical data compression has to rely on a predetermined model of how the data is distributed, P_{M_i}. Intuitively, this should introduce some inefficiency, thus increasing the theoretical limit below which the data cannot be compressed. This higher limit, measured by the cross entropy (2), is the sum of the Shannon entropy and the KL distance between M and the true data generating process. In other words, on top of the number of bits required to encode the true information content of the data, one has to add extra bits to account for the fact that the model distribution P_{M_i} does not exactly match the true distribution P_{dgp}.

The MDL principle is at the core of the proposed MIC precisely because of the flexibility it offers, enabling practical model comparison on the basis of simulated data alone. However, as pointed out by Grünewald (2007), MDL only provides a guiding principle for analysis and does not prescribe a specific methodology. It is therefore important to choose any implementation carefully and verify its efficiency. The context tree weighting (CTW) algorithm proposed by Willems et al. (1995), which forms the basis of the proposed MIC, is chosen specifically because of its desirable theoretical properties, discussed below.

2.2 Theoretical Properties of the MIC Procedure

Discounting a preliminary data preparation step required to convert the real-valued vector of data x_1^t to a discretised vector $\underset{\sqcup}{x}{}_1^t$, the methodology uses a two stage procedure to obtain the MIC. In the first stage, the CTW algorithm processes the simulated data generated by each candidate model M_i and produces a set of tree structures from which model-specific conditional probabilities $P_{M_i}(\underset{\sqcup}{X}{}_t | \underset{\sqcup}{x}{}_{t-L}^{t-1})$ can be recovered. In the second stage, each observation $\underset{\sqcup}{x}{}_t$ of the real data is scored using the corresponding CTW probability, providing a binary log score $\lambda_i(\underset{\sqcup}{X}{}_t | \underset{\sqcup}{x}{}_{t-L}^{t-1})$ which sums to the following cross entropy measure (2), which will form the basis of the MIC:[2]

$$\lambda_i\left(\underline{x}_{L+1}^{t}\right) = \sum_t \lambda_i\left(\underline{X}_t \,\middle|\, \underline{x}_{t-L}^{t-1}\right). \tag{4}$$

The justification for using the CTW algorithm in a two-stage procedure is that it endows the resulting criterion with the following properties. First of all, the measurement of cross-entropy (2) is *optimal*, as the inefficiency resulting from having to learn a model's conditional probability structure from simulated data attains the theoretical minimum and can therefore be controlled for. Importantly, the CTW is *universal* over FSMs, i.e., this optimal performance is guaranteed for all Markov processes of arbitrary order. This key guaranteed performance for all Markov processes underpins the name of the criterion. Finally, because the methodology processes observations *sequentially* in the second stage, it measures entropy at the level of each individual observation. This key property, which is not present in alternative methods which measure aggregate entropy by relying on empirical frequencies over the entire set of observations, allows for both analysis of model fit at the local level and confidence testing of the aggregate MIC value.

The actual measurement of cross entropy using the CTW probabilities is carried out using arithmetic encoding, a simple, elegant and efficient approach to data compression initially outlined by Elias (1975), and further developed by Rissanen (1976) and Rissanen and Langdon (1979) into a practical algorithm. In practice, the cross entropy of an r-bit observation $\underline{x}_t$ is simply the sum of the binary log scores for each bit, where $p_i(k) = P_{M_i}(\underline{X}_t\{k\} = 1 \,|\, \underline{x}_{t-L}^{t-1}, \underline{x}_t\{1, 2, \ldots, k-1\})$ is the probability of the kth bit being a '1' conditional on L lags of history and the previous $k-1$ bits of the observation:

$$\lambda_i\left(\underline{X}_t \,\middle|\, \underline{x}_{t-L}^{t-1}\right) = -\sum_{k=1}^{r}\left[\underline{x}_t\{k\}\log_2 p_i(k) + \left(1-\underline{x}_t\{k\}\right)\log_2\left(1-p_i(k)\right)\right]. \tag{5}$$

One of the key contributions of Elias (1975) is a proof that the difference between the binary log scores summed over the entire length of the data (4) and the theoretical cross entropy (2) is guaranteed to be less than two bits:

$$\lambda_i\left(\underline{x}_{L+1}^{t}\right) - E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{x}_{L+1}^{t})}\right] \leq 2. \tag{6}$$

The inefficiency incurred in the first stage by having to learn model probabilities using the CTW algorithm can similarly be bounded. The general intuition behind CTW is that each $\{0,\ 1\}$ bit in a binary string is treated as the result of a Bernoulli trial. More precisely, all the bits in the series that have the same past historical context $\underline{x}_{t-L}^{t-1}$,

[2] Because the aim of the paper is to present the desirable theoretical properties of the proposed criterion and assess their usefulness in practice, the more technical aspects relating to the algorithmic implementation are detailed in Appendix 2.

identified by the binary string s, and the same initial observation bits $\underbracket{x}_{t}\{1,\ 2, \ldots, k\}$, identified by string o, are governed by the same Bernoulli process with unknown parameter $\theta_{s,o}$. As the training series is processed, each node in the tree maintains a set of internal counters $\{a_{s,o},\ b_{s,o}\}$ for the number of times it has respectively observed a '0' or a '1' after having seen both context s and the first o bits of the current observation. Given these $\{a_{s,o},\ b_{s,o}\}$ counters, the estimator for Bernoulli processes developed by Krichevsky and Trofimov (1981) (henceforth referred to as the KT estimator) can be used to update the underlying probability of the binary sequence, by using the following recursion:

$$\begin{cases} P_e(0,\ 0) = 1, \\ P_e(a_{s,o},\ b_{s,o} + 1) = \frac{b_{s,o}+\frac{1}{2}}{a_{s,o}+b_{s,o}+1} P_e(a_{s,o},\ b_{s,o}), \\ P_e(a_{s,o} + 1,\ b_{s,o}) = \frac{a_{s,o}+\frac{1}{2}}{a_{s,o}+b_{s,o}+1} P_e(a_{s,o},\ b_{s,o}). \end{cases} \tag{7}$$

The ratios in the second and third equations of this recursion can be interpreted as frequency-based estimators of $\theta_{s,o}$ and $1 - \theta_{s,o}$, respectively. Clearly such a learning process has an efficiency cost, as one can see that the KT estimator (7) is initialised with an uninformative prior, where $P_e(1,\ 0) = P_e(0,\ 1) = 0.5$, regardless of the true value of $\theta_{s,o}$. While the frequencies will converge to $\theta_{s,o}$ and $1 - \theta_{s,o}$ as more training data is observed, the cost of compressing the observations sequentially using (7) rather than the true Bernoulli process with parameter $\theta_{s,o}$ can be expressed as:[3]

$$\chi\left(a_{s,o},\ b_{s,o}\right) = \log_2 \frac{(1-\theta_{s,o})^{a_{s,o}}\theta_{s,o}^{b_{s,o}}}{P_e(a_{s,o},\ b_{s,o})}. \tag{8}$$

The key contribution of Willems et al. (1995) is to prove that χ is bounded above by the following term:

$$\chi\left(a_{s,o},\ b_{s,o}\right) \leq \frac{1}{2}\log_2\left(a_{s,o} + b_{s,o}\right) + 1. \tag{9}$$

In order to show that this bound on χ is optimal, Willems et al. (1995) refer to the key contributions of Rissanen (1978, 1984, 1986), which show that if the probabilities P_{M_i} used to encode data come from a Markov process M_i parametrised by a vector θ_i that has to be estimated from the observations, then the effective lower bound on compression is higher than the Shannon entropy alone. This bound, known as the Rissanen bound, is the denominator of the following expression and depends on the number of parameters κ_i used in model M_i and the number of observations N available from which to estimate the parameters:

[3] One should note that the numerator of (8) differs from a standard binomial probability in that it does not contain the binomial coefficient $\binom{a+b}{b}$, which counts the number of ways one can distribute b successes within $a + b$ trials. This is because we are interested in the probability of a *specific* sequence of zeros and ones, therefore the multiplicity is omitted.

$$\lim_{N} \inf \frac{E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(x_1^t)}\right] - E_{dgp}\left[\log_2 \frac{1}{P_{dgp}(x_1^t)}\right]}{\frac{1}{2}\kappa_i \log_2 N} \geq 1. \tag{10}$$

This result, which resembles the Bayesian information criterion (BIC) developed during the same time period by Schwarz (1978), can be used to show that the bound on the inefficiency of the (9) estimator is optimal.[4] Because the KT estimator (7) estimates the parameter of a Bernoulli source, the number of ones and zeros observed must give the total number of observations, so $N_{s,o} = a_{s,o} + b_{s,o}$, and there is only a single parameter $\theta_{s,o}$, so $\kappa_{s,o} = 1$, and the Rissanen bound (10) for the KT estimator can be expressed as:

$$\lim_{a_{s,o}+b_{s,o}} \inf \frac{E_{dgp}[\chi(a_{s,o},\ b_{s,o})]}{\frac{1}{2}\log_2(a_{s,o} + b_{s,o})} \geq 1. \tag{11}$$

As pointed out by Willems et al. (1995), this makes the bound (9) optimal: the inefficiency of the KT estimator has to be greater in expectation than its Rissanen bound $1/2 \times \log_2(a+b)$, yet its realisation is proven to be less than the Rissanen bound plus the smallest possible information increment, i.e., one bit. This optimality argument lead Willems et al. (1995) to use the bound (9) as the measure of the learning cost of each KT estimator (7) in the tree. Because all the nodes in a context tree use the KT estimator (7), the aggregate inefficiency of learning the full set of transition probabilities in a context tree is bounded above by the following term, which simply sums the KT bound (9) over all contexts s and observations o, using a continuation term $\gamma(x)$ that ensures that nodes that have never been observed do not contribute to the expression:

$$\mathrm{X}_i = \sum_s \sum_o \gamma\left(a_{s,o} + b_{s,o}\right) \quad \gamma(x) = \begin{cases} x & \text{for } 0 \leq x < 1, \\ \frac{1}{2}\log_2(x) + 1 & \text{for } x \geq 1. \end{cases} \tag{12}$$

Crucially, Willems et al. (1995) prove that because all FSMX sources can be mapped to a context tree, and all nodes in the tree use the KT estimator (7), the aggregate learning cost for the full tree (12) is itself optimal for all FSMX sources, in particular all Markov process of arbitrary order L. This proven optimal performance over the very general class of Markov processes corresponds to the second important property mentioned above, universality, and justifies the choice of this algorithm for the proposed information criterion, rather than other existing Markov approximation methods such as the ones proposed by Tauchen (1986a, b) or reviewed in Kopecky and Suen (2010). Furthermore, in line with the MDL literature, this aggregate cost (12) can be used as a measure of the complexity of a model for the purpose of penalisation, which will be discussed further in Sect. 4.

[4] This similarity becomes even clearer when one considers that, as shown by Akaike (1974), the maximised likelihood function is a good estimator of the cross entropy (2). Rissanen (1984) is very aware of the similarity of the bound (10) with the BIC, and refers several time to the lineage of his work with Akaike (1974).

The final property of interest is the fact that the empirical observations are scored sequentially in the second stage, generating an observation-specific score $\lambda_{M_i}(\underset{\smile}{X}_t | \underset{\smile}{x}_{t-L}^{t-1})$, which sums up to the total length of the code string (4). This has three important implications, the first of which is the ability to assess the relative accuracy of models both at the global and local level, which will be illustrated in Sec. 3.3. Secondly, the availability of an observation-by-observation score provides the basis for statistical analysis and confidence testing of the MIC measurements obtained. In fact, as will be shown in Sect. 3.2, this makes the MIC methodology naturally suited to the MCS approach. Finally, the node counters $a_{s,o}$, $b_{s,o}$ which are used to calculate the transition probabilities (7) can also be used to produce a correction term for the measurement bias incurred by using these probabilities to calculate the entropy of the observation in the log score (5).

This bias occurs due to the fact that the KT estimator (7) uses the $a_{s,o}$ and $b_{s,o}$ counts to approximate the true parameters $\theta_{s,o}$, and this approximation may be poor, especially for transitions that are infrequently observed or for low lengths of training data. As pointed out in the literature initiated by Miller (1955) and Basharin (1959) on biased measurement of entropy, such a use of empirical frequencies instead of true probabilities will create a systematic bias.[5] In the case of the KT estimator, it is possible to take advantage of the optimal bound (9) to correct this bias.

It is important to point out, however, that one cannot directly use expressions (9) or (12) to correct this bias. These measure the learning cost incurred should one try to compress the training data in a single pass, i.e., when sequentially compressing each binary training observation after having used it to update the KT estimator (7). This is not what is done here: only the empirical data is compressed, and this is processed separately from the training data. What is required instead is the size of the learning cost at the margin, for the specific empirical observation being compressed, after having already observed a sequence of training observations. This can be obtained from (9) by calculating how much the overall learning cost of the KT estimator increases when an empirical observation is added to the existing tree built with the training data. Treating (9) as an equality and calculating the increase in the expression after adding a single extra zero or one observation provides the following approximation for the learning cost at the margin:

$$\frac{\Delta\chi(a_{s,o},\ b_{s,o})}{\Delta a_{s,o}} = \frac{\Delta\chi(a_{s,o},\ b_{s,o})}{\Delta b_{s,o}} \approx \frac{1}{2}\log_2 \frac{a_{s,o}+b_{s,o}+1}{a_{s,o}+b_{s,o}}. \tag{13}$$

Because the KT estimator only predicts the value of a single bit, the overall bias for a given data observation, identified by its specific context string $s*$ is calculated by summing (13) over the r set of counters $\{a_{s*,o},\ b_{s*,o}\}$ within node $s*$ that correspond to the bits of the observation string o:

[5] This literature mainly focuses on biases that occur when measuring Shannon entropy. This is discussed further in Appendix 1, which derives the expected bias for the case of a cross entropy measurement, which is what the MIC measures.

$$\epsilon_i\left(\underline{X}_t \middle| \underline{x}_{t-L}^{t-1}\right) = \frac{1}{2}\sum_{o=\varnothing}^{\underline{x}_t} \log_2 \frac{a_{s*,o} + b_{s*,o} + 1}{a_{s*,o} + b_{s*,o}}. \tag{14}$$

Subtracting the observation-level bias vector (14) from the raw score vector (4) results in a bias-corrected score $\lambda_i^{\epsilon c}(\underline{X}_t | \underline{x}_{t-L}^{t-1})$ which sums to the overall MIC:

$$\text{MIC}_i = \sum_t \left[\lambda_i\left(\underline{X}_t \middle| \underline{x}_{t-L}^{t-1}\right) - \epsilon_i\left(\underline{X}_t \middle| \underline{x}_{t-L}^{t-1}\right)\right]. \tag{15}$$

One might legitimately worry about the accuracy of the bias correction term (14) given that (13) is calculated by treating the bound (9) as an equality, rather than an inequality. However, given the tightness of this bound compared to the theoretical expectation (11), the approximation is expected to be accurate. Furthermore, it is shown in Appendix 1 that (14) corresponds to the bias one obtains by generalising Roulston (1999) to cross entropy. The benchmarking exercise carried out in the next section also establishes that (14) closely tracks the bias observed in practice.

2.3 Benchmarking the MIC's Theoretical Efficiency

The MIC (15) aims provide a reliable and statistically testable measurement of the cross entropy (2) between data and a model, for all models that are reducible to a Markov process. What makes this proposition feasible are the desirable theoretical properties of the two algorithms used to generate the MIC, i.e., the fact that difference between the log score provided by the Elias (1975) algorithm $\lambda_i(\underline{x}_1^t)$ and the true cross entropy over an entire sequence of data is tightly bounded by (6) and that bias term (14) provides an effective correction for having imperfectly learnt the probabilities in the first stage. Because these theoretical properties form the central justification for the procedure, it is important to check, as a first step, that they are achieved in practice when the procedure is implemented.

The benchmarking strategy chosen is to run the MIC procedure on a stream of data with known distribution, and therefore known entropy, and to compare the performance achieved in practice to the one expected in theory. In order to provide a reliable test of performance, 1000 random data series of length $N = 2^{19}$ are generated from the following beta distribution.[6]

$$X_t \underset{iid}{\sim} \text{Beta}(2,\ 7). \tag{16}$$

The beta distribution is chosen for its [0, 1] support, which simplifies the process of discretising the observations, and for its asymmetry under the chosen parameters, in order to test the CTW's ability to identify asymmetric distribution shapes. Furthermore,

[6] Because of the binary nature of the data and algorithm, the data lengths used throughout the paper are powers of two as this simplifies calculations requiring the binary logarithm $\log_2$.

Table 1 Algorithm performance on 1000 eight-bit beta distributed data series

	Shannon benchmark	Elias, fixed probability	Elias, CTW probability
λ/N, mean	6.9839	6.9839	6.9852
λ/N, std. dev.	0.0013	0.0013	0.0469
$\lambda/N,\ P_{2.5}$	6.9815	6.9815	6.8942
$\lambda/N,\ P_{97.5}$	6.9864	6.9864	7.0767
$\Delta\lambda/N$, mean	–	1.907×10^{-6}	3.132×10^{-4}
$\Delta\lambda/N$, std. dev.	–	1.327×10^{-12}	0.0016
$\Delta\lambda/N,\ P_{2.5}$	–	1.907×10^{-6}	−0.0026
$\Delta\lambda/N,\ P_{97.5}$	–	1.907×10^{-6}	0.0037
Theoretical bound	–	Elias (6)	Bias (14)
Value	–	3.815×10^{-6}	3.172×10^{-4}

λ/N measured cross-entropy per observation in bits, $\Delta\lambda/N$ measured cross-entropy per observation, relative to Shannon benchmark, in bits

the iid assumption means that each data series is memoryless, ensuring that the best possible compression performance per observation is simply the Shannon entropy of the overall distribution. For the purpose of the analysis, the 1000 series are quantised to an eight-bit level, i.e., $r = 8$.

For each of the 1000 data series, the eight-bit theoretical entropy is calculated by collecting the N observations into a 256 bin histogram on the [0, 1] support, which when normalised by N provides an empirical frequency vector f from which the Shannon entropy $S = -\sum_{i=1}^{256} f_i \log_2 f_i$ can be calculated. This provides the theoretical lower bound for compression and serves as the performance benchmark, the descriptive statistics of which are presented in the first column of Table 1.

The bound on the Elias algorithm (6) is tested by scoring each data series with the algorithm using its corresponding frequency vector f, and comparing the result against the theoretical benchmark. The result, displayed the second column of Table 1, confirms that the difference in performance between arithmetic encoding using the fixed probabilities f and the theoretical benchmark is vanishingly small (on the order of $1/N$) and remains below the Elias bound (6) of $2/N$. One can conclude from this result that as expected the implementation of the Elias algorithm provides an extremely reliable measure of cross-entropy (2).

A second test evaluates the learning cost of the CTW algorithm by training it with an additional, independent, stream of beta distributed data (16) and using the resulting CTW probabilities to compress the 1000 Monte Carlo data series. In order to provide an illustration of the literal learning curve of the CTW algorithm, the Monte Carlo analysis is run for increasing amounts of training data, from $T = 1$ to $T = 2^{19} = N$, the result being illustrated in Fig. 1.

Along with the third column of Table 1, Fig. 1a provides two key conclusions. The first is that, as expected, there is a learning curve: the performance of the algorithm

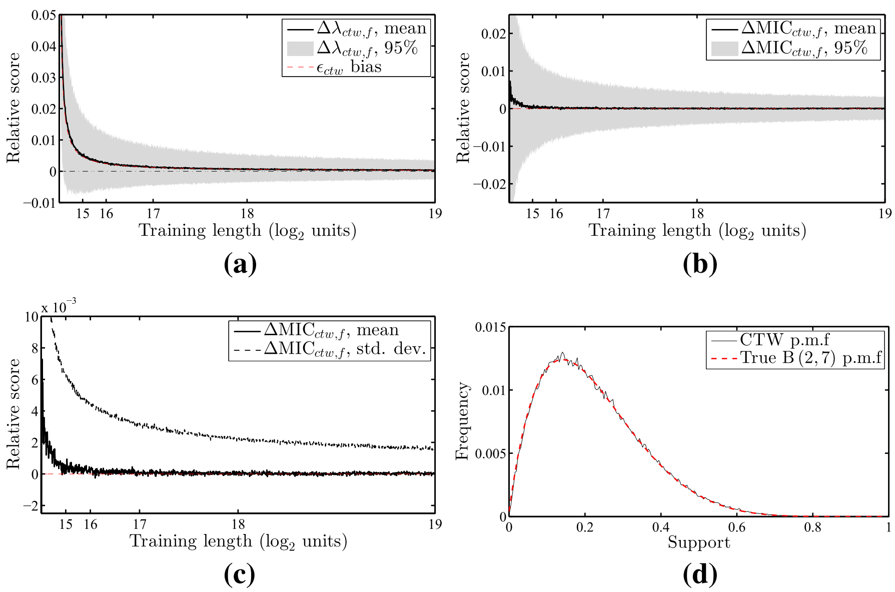

Fig. 1 Effective versus theoretical performance of CTW algorithm. **a** Relative score per observation. **b** Relative MIC per observation. **c** Learning curves. **d** CTW probabilities at $T = 2^{19}$

is poor at very low levels of training but converges to the benchmark as the amount of training data is increased. The second important element is that the bias correction term (14) closely tracks the mean value of the relative log scores (5) for the fixed and CTW probabilities, and is noticeable up until 2^{17}–2^{18} observations. Using the bias correction term, as shown in Fig. 1b, c, therefore ensures that even at low levels of training (around $2^{15} \approx 32,000$ observations) the expected difference between the MIC score with CTW probabilities and the score for fixed probability f is near zero. Beyond that point, additional training data is only useful in reducing the variance of the measurement, albeit with strongly diminishing returns. Figure 1c shows that most of the noise reduction is achieved around $2^{18} \approx 250,000$ observations, and doubling the amount of training data to 2^{19} observations brings little further improvement. A final confirmation of the CTW algorithm's good learning properties is seen in Fig. 1d, which shows that probability distribution learnt by the algorithm closely follows the beta distributed probability mass function (16).

3 Monte Carlo Validation on ARMA–ARCH Models

The practical usefulness of the MIC as an information criterion is tested on a series of ARMA–ARCH models by evaluating its ability to identify the true model as well as correctly rank the alternative models. This will also illustrate the MIC's performance on subsets of data, by attempting to identify portions of the data where the relative explanatory power of two models switches over.

Table 2 ARMA–ARCH model structures, parameter estimates and AIC rankings

	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
a_0	0	−0.048	0	0	0	0	0
a_1	0.7	–	0.957	0.874	0.299	0.694	0.690
a_2	0.25	–	–	0.087	0.642	0.254	0.256
b_1	0.2	0.916	−0.038	–	0.534	0.205	0.212
b_2	0.2	0.544	0.211	–	–	0.215	0.219
c_0	0.25	0.643	0.252	0.265	0.258	1.234	0.405
c_1	0.5	0.275	0.492	0.470	0.486	–	0.665
c_2	0.3	0.552	0.307	0.332	0.315	–	–
$E(\Delta \text{AIC}_{i,0})$	–	9510.25	38.32	424.95	245.40	4608.41	875.53
$std(\Delta \text{AIC}_{i,0})$	–	447.90	12.62	40.60	56.65	1438.50	160.10
AIC rank ρ_i^*	1	7	2	4	3	6	5

3.1 The ARMA–ARCH Model Specification and Monte Carlo Analysis

Because the MIC aims to generalise the AIC to all FSMX models, the analysis uses a set of ARMA models with ARCH errors, as it is possible to obtain the AIC for each of the models and use this as a basis for comparing the rankings produced by the MIC. The general structure for the set of models, presented in Eq. (17), allows for two autoregressive lags, two moving average lags and two ARCH lags.

$$\begin{cases} X_t = a_0 + a_1 X_{t-1} + a_2 X_{t-2} + b_1 \sigma_{t-1}\varepsilon_{t-1} + b_2 \sigma_{t-2}\varepsilon_{t-2} + \sigma_t \varepsilon_t, \\ \sigma_t^2 = c_0 + c_1 \varepsilon_{t-1}^2 + c_2 \varepsilon_{t-2}^2. \end{cases} \tag{17}$$

The various specifications used in the analysis only differ in their parameters, shown in Table 2. Only the parameters for the true model M_0 are chosen *ex-ante* with an aim to generate persistence in the auto-regressive components. The parameters for the alternate models M_1–M_6 are estimated using the following procedure. Firstly, a training data series with $T = 2^{19}$ observations is generated using the parameters for the true model and random draws $\varepsilon_t \underset{iid}{\sim} \text{N}(0,\ 1)$. The parameters for the alternate models are then obtained by using this data series to estimate the ARMA–ARCH equation (17) with the corresponding lag(s) omitted.[7]

Once the parameters are obtained, the various data series required for the Monte Carlo analysis of the MIC can be generated using Eq. (17) parameterised with the relevant column from Table 2 and further random draws $\varepsilon_t \underset{iid}{\sim} \text{N}(0,\ 1)$. T-length training series are generated for each of the six alternate specifications, and used in stage 1 to train the CTW algorithm. Similarly, 1000 ‘real’ data series of $N = 2^{13} = 8192$

[7] This was done in STATA using the ‘arch’ routine. As a robustness check, the true data series was also re-estimated, and the chosen parameters for the true model all fall within the 95 % confidence interval for the estimates.

Table 3 MIC performance on ARMA models, $T = 2^{22}$

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
MIC$_i$, mean	23,983.70	30,703.92	23,989.93	24,136.63	24,071.69	26,300.02	24,261.73
P($\rho_i = \rho_i^*$)	0.605	1	0.602	0.993	0.993	1	0.996
ΔMIC$_{i,0}$, mean	–	6720.22	6.24	152.93	87.99	2316.32	278.03
P(ΔMIC$_{i,0} > 0$)	–	1	0.605	1	0.999	1	1
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.991	0	0.939	0.002	0.110	0	0
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.978	0	0.899	0.001	0.057	0	0

$P(\rho_i = \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with Monte Carlo probability of being positive, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

observations are generated using the parameters for M_0. These are used in a Monte Carlo analysis of the MIC's ability to correctly rank the set of models.

The test benchmark is obtained by estimating the set of models using the 1000 N-length data series and calculating the respective AIC for each model and estimation. The descriptive AIC statistics are shown at the bottom of Table 2 and, as expected, the true model is consistently ranked first. An important point to keep in mind for the following section is that because the two AR parameters for M_0 are chosen so as to approach a unit-root behaviour, the AR(1) model M_2 is ranked an extremely close second. In fact, given the average $\Delta\text{AIC}_{2,0} = 38.32$, normalising by the number of observations N gives a mean AIC gap per observation of 4.7×10^{-3}, making those models difficult to distinguish in practice.

Finally, for the purpose of illustrating the local version of the MIC explored in Sect. 3.3, two additional sets of 1000 N-length data series are generated using model switching. In the first case, the data generating process uses M_0 for the first half of the observations before switching to M_2. In the second case, the data generating process starts with M_5 for half the observations before switching to M_0 for $N/4$ observations and then switching back to M_5 for the remainder of the series.

3.2 MIC Performance on ARMA–ARCH Models

The Monte Carlo analysis carried out on the ARMA–ARCH models follows the protocol outlined in Appendix 2. As a preliminary step, the data and training series are all discretised to a $r = 7$ bit resolution over the $[-30, 30]$ range. In the training stage (stage 1), the CTW algorithm was run on varying lengths of training series with a chosen tree depth of $d = 21$ bits, which corresponds to 3 observation lags L if one accounts for the seven-bit resolution. Finally, in stage 2, the trees are used to compress the 1000 data series, providing a MIC value for each of the models on each of the series.

Table 3 summarises the three main tests that were carried out to evaluate the MIC performance.[8] The first section examines whether the ranking assigned by the MIC to

[8] More detailed tables, which present the results for the MIC (15) at several values of the training length T and including upper/lower tail critical values for 95 % significance levels, are available in Appendix 3.

each model, ρ_i, matches the AIC ranking ρ_i^* in Table 2. This is a relatively strict test because the ranking for a given model i is affected by the performance of the MIC on all the other models, making this a global test of performance on the full model comparison set. Nevertheless, at training lengths $T = 2^{22}$ the probability $\mathrm{P}(\rho_i = \rho_i^*)$ of correctly replicating the AIC relative ranking is high for most models, except for M_0 and M_2, where the MIC does little better than random chance. The second test is less strict, as it instead looks at the probability of correctly selecting the best model in a simple head-to-head competition against the true model M_0. The $\mathrm{P}(\Delta \mathrm{MIC}_{i,0} > 0)$ values reveal that the MIC performs as expected, with a high probability of identifying the true model M_0, in all cases except M_2, where again the MIC does little better than a coin flip.

In addition to using Monte Carlo frequencies to evaluate the MIC's ability to rank models, one can take advantage of the availability of an N-length observation-level score $\lambda_i^{\epsilon c}(\underline{X}_t \,|\, \underline{x}_{t-L}^{t-1})$ which sums to the MIC to directly test the statistical reliability of the overall measurement at the level of each individual series. As stated previously, availability of this score means that the most natural and rigourous testing approach for the MIC is the reality check proposed White (2000) or the MCS of Hansen et al. (2011). The last part of Table 3 reveals the percentage of series for which the ith model is included in the MCS $\hat{\mathcal{M}}_{1-\alpha}$ at the 5 and 10 % confidence levels. The MCS procedure relies on 1000 iterations of the Politis and Romano (1994) stationary bootstrap for each of the Monte Carlo series, the block length for the bootstrap being determined using the optimal block length procedure of Politis and White (2004). Even accounting for the conservative nature of the MCS test, the procedure is able to effectively narrow down the confidence set to the subset of models that have the lowest MIC ranking. The MCS procedure also confirms the MIC's inability to distinguish M_0 and M_2, as they are almost always included in the confidence set $\hat{\mathcal{M}}_{1-\alpha}$.

3.3 Localised MIC Performance on ARMA–ARCH Models

The availability of the observation-level score $\lambda_i^{\epsilon c}(\underline{X}_t \,|\, \underline{x}_{t-L}^{t-1})$ also enables the calculation of a local version of the MIC, allowing models to be compared over subsets of the data. This is illustrated by running the procedure on the two sets of 1000 model-switching series mentioned in Sect. 3.1 using the CTW trees obtained for $T = 2^{22}$ training observations in the previous section. The results, which are presented in Fig. 2, have been smoothed using a 200 observation-wide moving average window. In order to also illustrate the small-sample properties of the MIC, the MCS procedure is run on the resulting 200-observation averages. These are shown in Fig. 2c, d, where the dark gray areas indicate observations for which the confidence set $\hat{\mathcal{M}}_{0.95}$ only includes M_0, and the lighter gray cases where the procedure is unable to separate M_0 from the alternate model (M_2 or M_5 depending on the case) and both remain in the confidence set $\hat{\mathcal{M}}_{0.95}$. The areas in white implicitly identify those data points where the confidence set $\hat{\mathcal{M}}_{0.95}$ is reduced to the alternate model.

In the first case, the localised version of the algorithm is unable to detect the transition from M_0 to M_2, for both the individual data series and the Monte Carlo average.

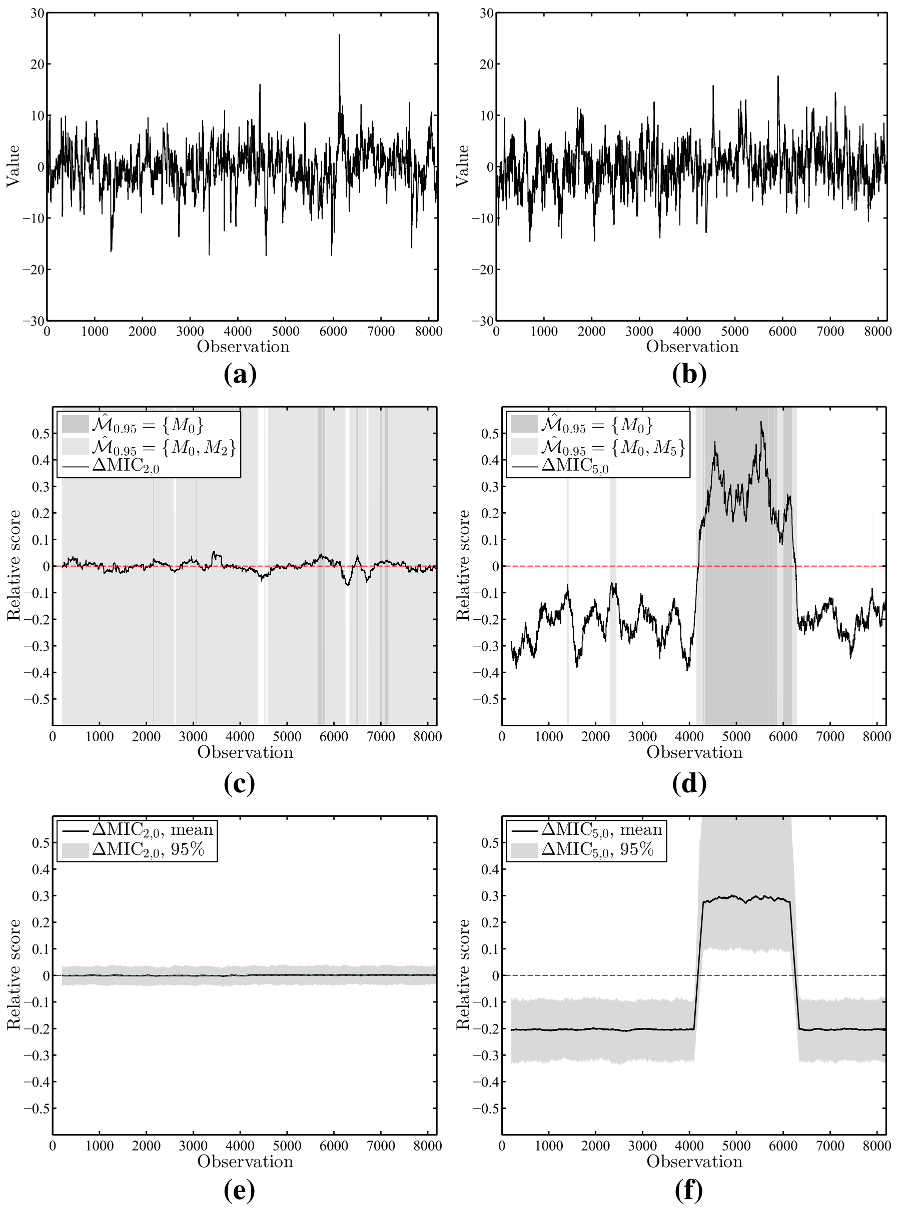

Fig. 2 Localised MIC performance. **a** Single data series, M_0 and M_2 switching. **b** Single data series, M_0 and M_5 switching. **c** Corresponding $\Delta\text{MIC}_{2,0}$. **d** Corresponding $\Delta\text{MIC}_{5,0}$. **e** $\Delta\text{MIC}_{2,0}$, Monte Carlo average. **f** $\Delta\text{MIC}_{5,0}$, Monte Carlo average

This is not surprising given that the MIC is unable to reliably distinguish between M_0 and M_2 at the aggregate level in Table 3. In the second case, however, the temporary switch from M_5 to M_0 is clearly visible in the Monte Carlo average and is detected by the MCS procedure on the individual data series. This localised version of the MIC may prove to be as useful for comparing model performance as the aggregate version presented above. It is indeed reasonable to expect that in practical situations one model may outperform others on aggregate yet may be beaten on some specific features of the data, as is the case in Fig. 2.

4 Robustness and Practical Applicability

The Monte Carlo analysis of Sect. 3 is designed in order to establish that the MIC can replicate the rankings of a known experimental setting under ideal conditions, as there is little purpose to a methodology that cannot achieve even this most basic task. It is therefore important to also clarify some of the practical issues related to using the methodology in a more realistic and restricted setting, as well as present some supplementary Monte Carlo analyses corresponding to those more realistic settings. An outline of the overall protocol and the more technical aspects of the algorithms are detailed in Appendix 2, while the extended tables of the Monte Carlo analyses are presented in Appendix 3.

4.1 Transforming and Discretising the Data

Because the MIC protocol operates on a binary description of the data, the first task to be carried out is to discretise the data to a binary resolution. This is done by choosing a set of bounds $\{b_l, b_u\}$ representing the range of variation of the data, and a resolution parameter r which both determines the number of bits used to describe an observation and the number of distinct discrete values the variable can take, i.e., 2^r. The bounds $\{b_l, b_u\}$ on the range of variation can be set in such a way that a small fraction of the observations are allowed to be out of bounds, in which case they are assigned the value of the corresponding bound, in what is essentially a winsorization of the data.

The first issue that discretisation procedure raises is one of stationarity. Clearly, the fact that the data have to be bounded within a range of variation imposes a requirement that the empirical and training data series used be weakly stationary, as is the case for example of the ARMA–ARCH processes used in Sect. 3. This is not the case of many economic variables in their original form, which are typically integrated of degree one. This imposes a prior transformation of data in order to ensure they can be bounded. As is the case for many econometric applications, the recommendation is to take first differences or growth rates (first log-differences) of variables before discretising them, in order to ensure that the number of observations falling out of bounds does not exceed a small fraction of the sample (5 %, for example).

A second, related, issue is the ergodicity and time-homogeneity of the training data. Both are standard assumptions in the data compression literature, starting with Shannon (1948), as they underpin the ability to learn the stable transition probabilities of a data source by taking the time average of its output. While time-homogeneity is not a strict requirement for the CTW algorithm, if a model possesses a tipping point which irreversibly changes the transition probabilities, CTW will learn a mixture of the two regimes. In practice, it might then be preferable to learn two separate sub-models corresponding to each regime. Similarly, if a model possesses absorbing states, it is preferable to run multiple simulations, each cut at the point of absorbtion, in order to improve the learning of the transition probabilities before absorbtion. It is important to point out that neither time homogeneity or ergodicity are required for the empirical data. As shown in Sect. 3.3, any tipping point in the empirical probabilities will be detected by a sharp degradation (or improvement) in the performance of the MIC for a given fixed model.

Finally, one must choose a resolution parameter r for the discretisation, which is also relatively straightforward. As explained in Appendix 2, discretising the data creates a quantisation error, and the resolution parameter should be chosen such that this error is distributed as an iid uniform variable over the discretisation interval. This can be tested statistically, and the recommended procedure is to run the quantisation tests from Appendix 2 on the empirical data and to choose the smallest value of r which satisfies the tests.

4.2 Choosing the Lags and Training Length

Once the resolution parameter has been set the next parameter to choose is the order of the Markov process L, or equivalently, the depth $d = rL$ of the tree. Two conflicting factors have to be considered in this decision. The first is that the worst-case memory requirement of the binary tree is $\mathcal{O}(2^d)$. Even though the implementation detailed in Appendix 2 uses the integer arithmetic of Willems and Tjalkens (1997) to reduce memory costs, this implies that setting $d > 32$ can potentially require an infeasible amount of memory to store the nodes. A related consideration is that the higher the order of the Markov process, the more training data will be required to adequately explore the larger transition space.

Conversely, there is the risk that setting the order of the Markov process too low compared to the true order of the underlying process will distort the measurement. Given the memory constraint previously mentioned, the immediate implication is that situations where the empirical data exhibits long memory will pose a problem for the methodology, and this should be tested for beforehand by looking, for instance, at the number of significant lags of the partial autocorrelation function.[9] The MIC methodology does seem to be robust to L, however, and in situations of non-persistent time dependence it should perform well. As an example, the order is set to $L = 3$ in the Monte Carlo analysis of Sect. 3, which is already less than the correct length $L = 4$ given (17).[10] This is tested further in Tables 8 and 9, which show the results of the ARMA–ARCH Monte Carlo analysis for $L = 2$, which is even more misspecified. As one would expect, this reduces the ability of the MIC to identify the true model, as is visible from the fact the number of models included in the MCS increases and the frequency at which models are correctly ranked falls. Nevertheless, when comparing the specifications head-to-head with the true model, the MIC is still able to correctly

[9] It is important to note, however, that long memory processes are typically problematic regardless of the statistical method used to analyse them.

[10] One can see from (17) that X_t depends on σ_{t-2}, which itself depends on ε_{t-4}.

pick the true model in the great majority of cases. In practice, given resolutions of around 6–8 and having tested for long-range dependence, it is therefore reasonable to set the Markov order L to 3–4.

The final choice for any practical application is the amount of training and empirical data to be used for the analysis. Because the purpose of Sect. 3 is to establish that the MIC rankings can match the AIC rankings in the limit, long training sequences of between 2^{19} and 2^{22} are used, as well as a relatively large 'empirical' sample size $N = 2^{13} = 8192$. This is unrealistic for a practical setting, therefore further Monte Carlo robustness checks are carried out Tables 6, 8 and 10, for shorter training lengths of $2^{15} \approx 32{,}000$ to $2^{18} \approx 250{,}000$ and Tables 10 and 11 for shorter empirical series of $N = 500$, in order to clarify these choices.

The findings for the shorter empirical setting of $N = 500$, first of all, confirm that sample size is important for a correct identification of the model rankings, which is what one would expect intuitively. The MIC is able to correctly rank and reject the two worst models for even relatively low levels of training, but it has trouble distinguishing the remaining models, as seen both by the size of the MCS and the frequency of correct rankings. As was the case for the misspecified $L = 2$ example, however, head-to-head comparisons with the true model do frequently lead to the true model being correctly picked. This is also in line with the analysis in Fig. 2, which used a moving average window of $N = 200$ for its head-to-head analysis.

Finally, the Monte Carlo analyses for $T = 2^{15}$ to 2^{22} help to confirm the learning curve of the MIC for a more complex setting than the Beta benchmarking exercise in Sect. 2.3. Crucially, the main finding is consistent with the benchmarking in that for the three settings examined, the mean MIC measurement converges relatively quickly (stabilising after about $T = 2^{17}$), while the variance (as measured by the MCS size) takes longer to settle down. In most cases, the results seem reliable at some point between $T = 2^{18}$ and 2^{19}, suggesting that 250,000–500,000 training observations provide a good learning performance. An important clarification in this regard is that the training observations need not be generated in a single simulation run, as is the case for the simple ARMA–ARCH example used here. As an example, one could also use a set of 1000 Monte Carlo runs of 250 or 500 observations each. Given that ABM simulations or DSGE stochastic simulations often generate such Monte Carlo sets as part of their internal validation, these can directly be re-used in the MIC procedure.

4.3 Penalising Complexity

At this stage, it is important to mention how the penalisation of model complexity can be handled within the methodology. This has not been done in the main analysis, as its main focus is to test the accuracy of the cross entropy measurement based on the CTW algorithm, and the benchmark for comparison is therefore the AIC, which does not strongly penalise the complexity of the models involved. Furthermore, given the similar parameter dimensions of the ARMA models in Table 2, simply adding a BIC-like penalisation term $\kappa_i/2 \times \log_2 N$ to the MIC (15), where κ_i is the number of parameters in the ith model, will not generate meaningful differences in complexity.

Table 4 MDL complexity X_i^N of ARMA models, $L = 3,\ N = 2^{13}$

$\log_2 T$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
15	73,230.81	77,705.93	74,631.46	72,258.36	71,462.14	92,564.47	77,437.20
16	73,039.02	76,701.42	73,327.23	72,099.97	70,946.11	91,425.17	77,129.25
17	73,186.13	76,516.12	73,792.56	71,622.96	71,028.26	91,008.78	77,200.49
18	73,547.16	76,316.96	73,157.97	71,447.64	71,003.63	90,770.46	77,456.41
19	73,331.10	76,008.85	73,071.52	71,373.93	71,159.21	90,852.24	77,557.02
20	73,050.36	75,810.05	73,169.77	71,472.16	71,157.35	90,715.94	77,619.32
21	72,808.90	75,575.97	72,985.64	71,105.81	70,729.82	90,642.48	77,295.94
22	72,206.85	75,051.22	72,517.48	70,610.63	70,221.86	90,191.76	76,806.39

A more interesting penalisation strategy, following Grünewald (2007) and consistent with the MDL underpinning of the methodology, is to use a two-part code based on the complexity cost of the context tree corresponding to each mode M_i. This accounts not only for the parameter dimension of the models, but also for their stochastic complexity, by taking advantage of the fact this complexity is directly measured by the cost term (12) of the context tree corresponding to each model. As was the case for the calculation of the bias correction term (14), however, one must account for the fact that the length of the training data T and empirical data N are not the same: naively using (12) would make the relative weight of the penalisation and the cross entropy directly dependent on T.[11] This implies that the calculation of the complexity (12) must be modified in order to produce a stable measurement. This can be done by normalising the $(a_{s,o},\ b_{s,o})$ to their expectation values had only N training observations been observed:

$$\mathrm{X}_i^N = \sum_s \sum_o \gamma\left(\frac{N(a_{s,o} + b_{s,o})}{T}\right) \quad \gamma(x) = \begin{cases} 0 & \text{for } 0 \le x < 1, \\ \frac{1}{2}\log_2(x) + 1 & \text{for } x \ge 1. \end{cases} \tag{18}$$

Normalised penalisations (18) for various training lengths T in the main Monte Carlo exercise with $L = 3$ and $N = 2^{13}$ are presented in Table 4.[12] Although these values do seem to decrease with increasing T, the sensitivity is very limited, and the relative complexity of the models is stable. Interestingly, the results suggest that from a complexity viewpoint, the simplest models from the set in Table 2 are the ones with little or no moving average component, while dropping either the autoregressive or ARCH components significantly increases the complexity of the resulting model.

[11] As an example, note the similarity of MIC scores in Table 7 for $T = 2^{21}$ and 2^{22}. One would expect a penalised version of the MIC scores to also be similar in both cases, given the large training lengths involved. However, the naive complexity (12) of the trees for $T = 2^{22}$ will be much higher than for $T = 2^{21}$ even thought the number of empirical observations is unchanged at $N = 8192$, as twice as many $(a_{s,o},\ b_{s,o})$ observations will have been processed.

[12] These would be appropriate for penalising the MIC measurements in Tables 6 and 7.

5 Conclusion

This paper develops a methodology which follows the MDL principle and aims provide an information criterion with practically no formal requirement on model structure, other that it be able to generate a simulated data series. While the MDL-inspired algorithms might be unfamiliar, it is important to emphasise that this methodology nevertheless follows the same logic as the AIC, which is that one can measure the cross entropy between some data and a model without knowing the true conditional probability distributions for the events observed. Because these measurements contain an unobservable constant (the true information entropy of the data generating process), one then has to take differences across models to obtain the difference in KL distance, which is the desired indicator of relative model accuracy. The difference between the AIC and the methodology suggested here rests simply in the choice of method used to measure the cross entropy.

The main benefit of the proposed methodology is that it compares models on the basis of the predictive data they produce, by using the CTW algorithm to estimate the transition tables of the underlying FSMs corresponding to the candidate models. This mapping of models to a general class of FSMs explains why there is no requirement that the candidate models have a specific functional form or estimation methodology, only that they be able to generate a predicted data series. It is this specific aspect which, while unfamiliar, enables the information criterion to compare across classes of models, from regression to simulation.

The Monte Carlo analyses in Sects. 2.3 and 3 provide several validations of the methodology by demonstrating that, given enough training data, the MIC procedure provides model ranking information that is comparable to the AIC, and also illustrates a bias correction procedure that can be used to increase the accuracy of the MIC measurement and a penalisation term based on the complexity of the context tree corresponding to each model. The main drawback of the procedure is that some inefficiency is incurred by the CTW algorithm having to learn the transition table of a model from the training data. A large part of this can be corrected using the known theoretical bounds of the CTW algorithm, however the residual variation creates a "blind spot" which limits the MIC's ability to distinguish similar Markov models. The Monte Carlo analysis shows that this blind spot is reduced by increasing both training and empirical data, and crucially because cross entropy is measured at the observation level, the reliability of any comparison between models can be tested statistically using the MCS approach. This feature also allows comparison of models on subsets of the data, thus detecting data locations where the relative performance of models switches over.

This paper only provides a proof-of-concept, however, and it is important to point how one might extend this methodology to more common settings. Indeed, the work presented here focuses on a univariate time-series specification, where the candidate models attempt to predict the value of a single random outcome conditional on its past realisations, i.e., $P(X_t|x_{t-L}^{t-1})$. While this is reasonable as a starting point for establishing that the methodology works on small-scale problems, it is important to outline how it can be scaled up to larger settings.

First of all, extending the approach to multivariate models poses no conceptual problem, as the current state of a FSM does not have to be restricted to a single variable. Supposing that X_t represents instead a state vector made up of several variables $\{A_t,\ B_t,\ C_t, \ldots\}$, one could use the preliminary step of the protocol in Appendix 2 to discretise each variable at its required resolution $r(a),\ r(b),\ r(c), \ldots$ The binary string for an observation $\underline{x}_t$ is then simply the concatenation of the individual observations $\underline{a}_t,\ \underline{b}_t,\ \underline{c}_t$, and its resolution r is the sum of the individual variable resolutions, i.e., $\Sigma_i r(i)$.

Secondly, in the time-series setting presented here, the observations in x_1^t are used both as the outcome to be predicted (in the case of the current observation) and the conditioning information for the prediction (for past observations). However, there no reason why the outcome and conditioning data could not be separated. Keeping x_1^t as the conditioning data, the methodology can also generate predictions about the state of a separate outcome variable or vector Y_t, i.e., probability distributions of the type $P(Y_t | x_{t-L}^t)$, which can be used to extend the MIC approach beyond time-series analysis.

While both these extensions are conceptually feasible, they create implementation challenges. As stated previously, the larger resolution $\Sigma_i r(i)$ of a multivariate setting implies a correspondingly larger depth d of the binary context tree for any given number of time lags L, increasing the memory requirement for storing the tree nodes. Ongoing work on the implementation of the CTW algorithm is specifically directed at improving the memory efficiency of the algorithm in order to address this last point and turn the proposed methodology into a practical tool.

Acknowledgements The author is grateful to WEHIA 2013, CFE 2013 and CEF 2014 participants for their comments and suggestions on earlier versions of this paper. The author would like to thank particularly Jagjit Chadha, Stefano Grassi, Philipp Harting, Sander van der Hoog, Sandrine Jacob-Léal, Miguel León-Ledesma, Thomas Lux, Mauro Napoletano, Lionel Nesta, John Peirson and the two anonymous referees whose suggestions greatly helped to improve the manuscript. Finally, a special thanks to James Holdsworth, Steve Sanders and Mark Wallis for their invaluable help in setting up and maintaining the SAL and PHOENIX Computer Clusters on which the Monte Carlo analysis was run. Any errors in the manuscript remain of course the author's.

Appendix 1: Bias Correction in Measured Cross-Entropy

It has been known since Miller (1955) and Basharin (1959) that measuring Shannon entropy using observed frequencies rather than the true underlying probabilities generates a negative bias due to the interaction of Jensen's inequality with the inherent measurement error of the frequencies. A large literature exists calculating analytical expressions for this bias, good examples of which are Carlton (1969), Panzeri and Treves (1996) or Roulston (1999). These expressions can be generalised to the case of measuring the cross entropy between empirical and simulated data, in order to provide a clarification of the bias correction term derived in Sect. 2.2. As will be shown in the derivation below, which generalises the approach of Roulston (1999), in the case of cross entropy the resulting bias is in fact positive.

Letting B be the number of states a system can occupy, q_i the probability of observing the ith state in the empirical data and p_i the corresponding model probability, the theoretical cross entropy is given by:

$$H = -\sum_i^B q_i \ln p_i. \tag{19}$$

In practice, this is measured using frequencies f_i and g_i, where n_i, t_i are the observed realisations for the ith state in the empirical and simulated data, respectively, with N and T as the overall number of observations.

$$\hat{H} = -\sum_i^B \frac{n_i}{N} \ln \frac{t_i}{T} = -\sum_i^B f_i \ln g_i. \tag{20}$$

The count variables n_i and t_i can be considered as binomial distributed variables with parameters q_i and p_i, with the following expectations and variances:

$$\begin{cases} E[n_i] = Nq_i \\ E[t_i] = Tp_i \end{cases} \quad \begin{cases} V[n_i] = Nq_i(1-q_i), \\ V[t_i] = Tp_i(1-p_i). \end{cases} \tag{21}$$

Following Roulston (1999), the frequencies f_i and g_i can be redefined in terms of a measurement error relative to the true underlying probabilities q_i and p_i.

$$\begin{cases} f_i = q_i(1+\eta_i) \\ g_i = p_i(1+\varepsilon_i) \end{cases} \quad \begin{cases} \eta_i = \frac{f_i - q_i}{q_i}, \\ \varepsilon_i = \frac{g_i - p_i}{p_i}. \end{cases} \tag{22}$$

Given the mean and variance of the binomial counts (21), one can see that the measurement errors have mean $E[\eta_i] = E[\varepsilon_i] = 0$ and variances:

$$E\left[\eta_i^2\right] = \frac{1-q_i}{Nq_i} \quad E\left[\varepsilon_i^2\right] = \frac{1-p_i}{Tp_i}. \tag{23}$$

Replacing (22) into the measured cross entropy term (20) allows the bias to be expressed in terms of η_i and ε_i:

$$\hat{H} = H - \sum_i^B q_i \left[\eta_i \ln p_i + (1+\eta_i) \ln(1+\varepsilon_i)\right]. \tag{24}$$

Because it is expected that $|\varepsilon_i| < 1$, Roulston (1999) uses a second order Taylor expansion for the logarithmic term, $\ln(1+\varepsilon_i) \approx \varepsilon_i - \varepsilon_i^2/2$. Replacing in the previous expression and rearranging provides the following expression.

$$\hat{H} \approx H - \sum_i^B q_i \left[\eta_i \ln p_i + \varepsilon_i + \underbrace{\eta_i \varepsilon_i}_{b} - \frac{\varepsilon_i^2}{2} - \eta_i \frac{\varepsilon_i^2}{2}\right]. \tag{25}$$

The critical component of the bias is the product of errors, labeled b. Suppose for the moment that the empirical and simulated data are identical, so that $q_i = p_i$, $f_i = g_i$, $\eta_i = \varepsilon_i$ and $N = T$. In that case the cross entropy term (19) measures the Shannon entropy, b offsets the negative $\varepsilon_i^2/2$ term in equation (25), and one recovers the derivation in Roulston (1999):

$$\hat{H} \approx H - \sum_i^B p_i \left[\varepsilon_i \left(\ln p_i + 1 \right) + \frac{\varepsilon_i^2}{2} - \frac{\varepsilon_i^3}{2} \right]. \tag{26}$$

Given $E[\varepsilon_i] = 0$ and the variance $E[\varepsilon_i^2]$ in (23) the second order approximation for the bias is given by the following expression, where B^* corresponds to the number of states for which $p_i \neq 0$.

$$E[\hat{H}] \approx H - \sum_i^B \frac{p_i E[\varepsilon_i^2]}{2} = H - \frac{B^* - 1}{2N}. \tag{27}$$

This is the finding reported by Roulston (1999), which is that measured Shannon entropy systematically *underestimates* the true Shannon entropy, and to a second order the bias does not depend on the distribution p_i.

This is not the case for the more general situation where $q_i \neq p_i$ and $f_i \neq g_i$. Because it is reasonable to assume that the measurement errors on the real and simulated probabilities are independent, their covariance $E[\eta_i \varepsilon_i]$ will be zero and the bias term b in (25) no longer counteracts the negative $\varepsilon_i^2/2$ term. Furthermore, the weighting of the bias terms by q_i rather than p_i means that the p_i denominator in the variance of ε_i will not cancel out. The expectation of the second order approximation is now:

$$E[\hat{H}] \approx H + \sum_i^B \frac{q_i E[\varepsilon_i^2]}{2} = H + \frac{1}{2T} \left[\sum_i^B \frac{q_i}{p_i} - 1 \right]. \tag{28}$$

This produces a situation which is the opposite of the one observed for Shannon entropy in (27): measured cross entropy systematically *overestimates* the true cross entropy and the bias (28) now depend on the unknown q_i and p_i distributions. We now show that the bias correction term (14) approximates this bias in expectation. Recall that given an observation $\underline{x}_t$, identified by the binary string o and a context $\underline{x}_{t-L}^{t-1}$, identified by the binary string s, the correction is given by:

$$\epsilon \left(\underline{X}_t \middle| \underline{x}_{t-L}^{t-1} \right) = \frac{1}{2} \sum_{o=\varnothing}^{\underline{x}_t} \log_2 \frac{a_{s*,o} + b_{s*,o} + 1}{a_{s*,o} + b_{s*,o}} = \frac{1}{2} \sum_{o=\varnothing}^{\underline{x}_t} \log_2 \left[1 + \frac{1}{a_{s*,o} + b_{s*,o}} \right]. \tag{29}$$

Letting $\tau_{s,o} = a_{s*,o} + b_{s*,o}$ be the number of times the Markov transition identified by the s, o binary strings has been seen in the training data, and taking a first order approximation of the binary logarithm, the correction for that specific observation $\underset{\sqcup}{x}_t$ is approximated by

$$\epsilon\left(\underset{\sqcup}{X}_t \middle| \underset{\sqcup}{x}_{t-L}^{t-1}\right) \approx \frac{\log_2 e}{2} \sum_{o=\varnothing}^{\underset{\sqcup}{x}_t} \frac{1}{\tau_{s*,o}}. \tag{30}$$

Summing this term over all the observations in the empirical data on the left hand side is equivalent to summing over all the contexts and observation strings $s,\ o$ on the right hand side:

$$\sum_{t=L+1}^{N} \epsilon\left(\underset{\sqcup}{X}_t \middle| \underset{\sqcup}{x}_{t-L}^{t-1}\right) \approx \frac{\log_2 e}{2} \sum_{s} \sum_{o} \frac{n_{s*,o}}{\tau_{s*,o}}. \tag{31}$$

Dividing both sides by the number of transitions observed $N - L$ provides the expected correction term for the empirical data set. Note that the total number of observations T is added to both the numerator and denominator of the right hand side:

$$E\left[\epsilon\left(\underset{\sqcup}{X}_t \middle| \underset{\sqcup}{x}_{t-L}^{t-1}\right)\right] \approx \frac{\log_2 e}{2T} \sum_{s} \sum_{o} \frac{n_{s*,o}}{N-L} \frac{T}{\tau_{s*,o}}. \tag{32}$$

The two ratios in the right hand side terms correspond to the observed frequencies of the Markov transition identified by binary strings $s,\ o$ in the empirical and training data, respectively, therefore $E[n_{s*,o}/(N-L)] = q_{s*,o}$ and $E[\tau_{s*,o}/T] = p_{s8,o}$. One can therefore see that allowing for the change in logarithmic base, the expectation of (32) is essentially equivalent to (28). The main advantage of using the error correction term (14) is that the bias is corrected at the level of each observation rather than over the entire empirical data, which facilitates the evaluation of model performance at the local level.

Appendix 2: Technical Appendix—The MIC Protocol

Only two inputs are required in order to run the protocol. The first is a $N \times 1$ data series that provides the empirical benchmark for model comparison. The second input is a set of training series produced from the various fitted, calibrated or simulated models in the comparison set $\{M_1,\ M_2, \ldots, M_m\}$, etc., which should be $T \gg N$ observations in length. An overview of the stages of the methodologies is provided in Table 5, and each stage is discussed in the subsequent subsections

Stage 0: Discretising the State Space and Testing for Quantisation Error

The Krichevsky and Trofimov (1981) estimator, at the core of the CTW and context tree maximising (CTM) algorithms, can only be used to predict the probability distribution of a binary event, and can only therefore deal with binary data. As a result, a preliminary

Table 5 Overview of the methodology stages

Stages	Names	Inputs	Outputs	Parameters	Comments
0	Discretisation	$N \times 1$ real-valued empirical series	$N \times 1$ discretised empirical series	r, resolution	Quantisation tests can be carried out to choose r
		$T \times 1$ real-valued simulated series	$T \times 1$ discretised simulated series	b_l, lower variation bound	
				b_u, upper variation bound	
1	Training	$T \times 1$ discretised simulated series	Context tree structure T_i	d, tree depth	The number of significant lags in the PACF can help choose the Markov order L
					Depth is $d = rL$
2	Scoring	T_i Context tree structure	$N - L \times 1$ log score λ_i	–	
		$N \times 1$ real-valued empirical series	$N - L \times 1$ bias correction ϵ_i		
			Penalisation term X_i		

step is to digitise the data and model training series and convert them into binary strings. It is important, ensure that the no essential information about the variation in the data is lost through the quantisation error.

Data Discretisation and Observation Structure

The discretisation of the data is controlled by two sets of parameters, first a set of bounds $\{b_l,\ b_u\}$ representing the range of variation of the data, and secondly a resolution parameter r, which determines the number of bits used to describe an observation and the number of distinct discrete values the variable can take, i.e., 2^r. It needs to large enough to capture the significant variation in the data and training series, but not so large that it increases the complexity and memory requirements of the CTW algorithm unnecessarily. Figure 3 provides an illustration of the process for $r = 3$.

The digitisation itself is carried out in a straightforward manner: Given a choice of bounds $\{b_l,\ b_u\}$ and a resolution r, the data vector x is first normalised to the $[0,\ 2^{r-1}]$ interval as follows:

$$x' = \frac{x - b_l}{b_u - b_l} 2^{r-1}. \tag{33}$$

The normalised data values x' are then rounded to the nearest integer, with any values outside the bounds being rounded to the bounds themselves.

$$\underset{\llcorner\lrcorner}{x} = \lfloor x' \rceil. \tag{34}$$

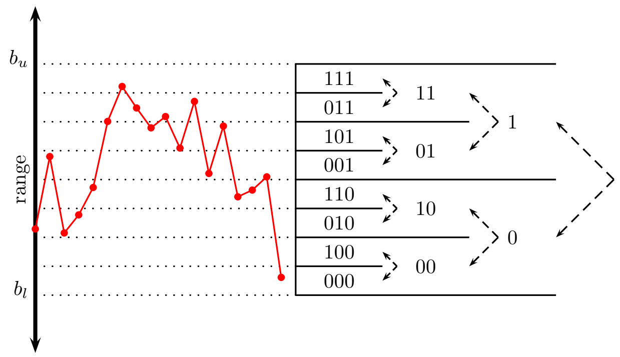

Fig. 3 Converting observations to binary strings

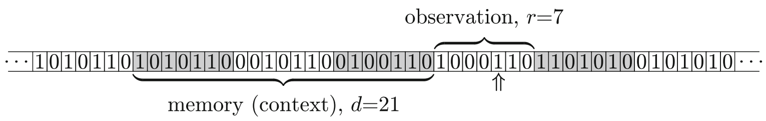

Fig. 4 Structure of a binary data string

The resulting vector of integers $\underset{\smile}{x}$ is mapped to its binary values, using the scheme illustrated in Fig. 3. The most significant bit (MSB), which indicates whether the observation is in the top or bottom half of the range is on the right, with the least significant bit (LSB) on the left. It is this structure which allows the Krichevsky and Trofimov (1981) estimator, which can only deal with Bernoulli processes, to be extended to any arbitrary categorical distribution.

Once each observation is discretised, the binary strings are concatenated, as shown in Fig. 4. The specific configuration illustrated in the figure corresponds to the one used for the ARMA–ARCH analysis presented in the paper: the range of variation was bounded within $[-30,\ 30]$, 7 bits were chosen to represent each observation and the tree depth was set at 21 bits, or 3 lags.

The Choice of r: Testing for Quantisation Error

It should be apparent that the rounding operation (34) creates a quantisation error $\varepsilon = x' - \underset{\smile}{x}$ which given the normalisation (33) is distributed over the unit length interval $[-0.5,\ 0.5[$. Some of the information content of the data x is therefore inevitably discarded by the quantisation (34), and it is important to check that this does not affect the measurements. If the quantisation error ε satisfies the following two assumptions, then the information loss will not affect the MIC:

iid Uniformity: $\varepsilon \underset{iid}{\sim} \mathcal{U}(-0.5,\ 0.5)$,
Uncorrelatedness: $\operatorname{corr}(\varepsilon,\ \underset{\smile}{x}) = 0$.

Proposition *If the quantisation error $\varepsilon = x' - \underset{\smile}{x}$ satisfies iid uniformity and uncorrelatedness at a resolution r^*, then choosing a larger $r = r^* + \phi$ increases the log score $\lambda_i(\underset{\smile}{X})$ by an additive constant for all models M_i.*

Proof Given a data vector x, let r^* be the lowest resolution such that the quantisation error ε^* resulting from the corresponding digitisation $\underline{x}^*$ satisfies the iid uniformity and uncorrelatedness assumptions. Let $\underline{x}$ be an alternative digitisation of x with the same bounds $\{b_l,\ b_u\}$ but a higher resolution $r = r^* + \phi$. Let us calculate the number of bits required to encode the tth observation at resolution r:

$$\lambda_i\left(\underline{X}_t \middle| \underline{x}_{t-L}^{t-1}\right) = E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}_t | \underline{x}_{t-L}^{t-1})}\right]. \tag{35}$$

First, because the conditional probability of each of the r observation bits is conditioned on the values of previous observation bits, the chain rule for conditional probabilities ensures that (35) is simply the sum of the cross entropies for each bit.

$$\begin{aligned} \forall t,i \quad & E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}_t | \underline{x}_{t-L}^{t-1})}\right] \\ & = E_{dgp}\left[\sum_{k=1}^{r} \log_2 \frac{1}{P_{M_i}(\underline{X}_t\{k\} | \underline{x}_{t-L}^{t-1},\ \underline{x}_t\{1,\ 2, \ldots, k-1\})}\right]. \end{aligned} \tag{36}$$

Additivity of the expectations operator $E_{dgp}[\ldots]$ then implies:

$$\begin{aligned} \forall\, t,i \quad & E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}_t | \underline{x}_{t-L}^{t-1})}\right] \\ & = \sum_{k=1}^{r} E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}_t\{k\} | \underline{x}_{t-L}^{t-1},\ \underline{x}_t\{1,\ 2, \ldots, k-1\})}\right]. \end{aligned} \tag{37}$$

Second, the normalisation (33) and rounding (34) operations ensure that by construction the binary string formed by $\underline{x}_t$ contains the shorter $\underline{x}^*_t$ as a substring:

$$\forall\, k \in \{1,\ 2, \ldots, r^*\}, \quad \underline{x}_t\{k\} = \underline{x}^*_t\{k\}. \tag{38}$$

Regarding the first r^* bits in the sum (37), the nesting of $\underline{x}^*_t$ in $\underline{x}_t$ implies that the only difference between predicting $\underline{X}_t\{1,\ 2, \ldots, r^*\}$ and $\underline{X}^*_t\{1,\ 2, \ldots, r^*\}$ is that each of the lagged observations in $\underline{x}_{t-L}^{t-1}$ contains more information than in $\underline{x}^{*\,t-1}_{t-L}$, specifically, ϕ bits of information drawn from the quantisation error ε^*. However, ε^* is iid and uncorrelated with $\underline{x}^*$, therefore this extra information cannot improve the conditioning compared to what is achieved using only the r^*-length substring $\underline{x}^*$:

$$\begin{aligned} \forall\, t,i,k \le r^* \quad & P_{M_i}\left(\underline{X}_t\{k\} = 1 \middle| \underline{x}_{t-L}^{t-1},\ \underline{x}_t\{1,\ 2, \ldots, k-1\}\right) \\ = \quad & P_{M_i}\left(\underline{X}^*_t\{k\} = 1 \middle| \underline{x}^{*\,t-1}_{t-L},\ \underline{x}^*_t\{1,\ 2, \ldots, k-1\}\right). \end{aligned} \tag{39}$$

Predicting the values of the last ϕ bits of $\underline{X}_t$ in (37) is equivalent to predicting the value of ε_t^* at a resolution ϕ. However, ε^* is iid uniformly distributed over $[-0.5,\ 0.5]$ and uncorrelated with $\underline{x}$ therefore:

$$\forall\, t, i, k > r^* \quad P_{M_i}\left(\underline{X}_t\{k\} = 1 \middle| \underline{x}_{t-L}^{t-1},\ \underline{x}_t\{1,\ 2, \ldots, r^*\}\right) = 0.5. \tag{40}$$

Replacing (39) and (40) in the sum of conditional entropies (37) provides the following result:

$$\forall\, t, i \quad E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}_t | \underline{x}_{t-L}^{t-1})}\right] = E_{dgp}\left[\log_2 \frac{1}{P_{M_i}(\underline{X}^*_t | \underline{x}^{*\,t-1}_{t-L})}\right] + \phi. \tag{41}$$

One therefore has $\lambda_i(\underline{X}_t | \underline{x}_{t-L}^{t-1}) = \lambda_i(\underline{X}^*_t | \underline{x}^{*\,t-1}_{t-L}) + \phi$. □

In essence, if the iid uniformity and uncorrelatedness assumptions on ε are satisfied at resolution r^*, then choosing a higher resolution $r^* + \phi$ for the N data observations will simply result in a log score λ equal to the one obtained at resolution r^*, plus an additional $\phi \times N$ extra bits, for all models in the comparison set $\{M_1,\ M_2, \ldots, M_m\}$. This additive term will drop out when taking differences between models.

We therefore suggest an incremental testing procedure, starting at $r = 1$, where the following three tests are carried out to ensure that the quantisation error satisfies the uniformity, independence and uncorrelatedness assumptions above. The correct value of r is the lowest value that satisfies the three following tests:

- A Kolmogorov–Smirnov (KS) test on the quantisation error η, where H_0 is that η is uniformly distributed over the range $[-0.5,\ 0.5[$.
- A Ljung–Box (LB) test on the autocorrelation coefficients of η, where H_0 is that η is independently distributed.
- A second LB test on the cross correlation coefficients of η and the quantised values $\underline{x}$, where H_0 is that η and $\underline{x}$ are independent.

Stage 1: The Context Tree Weighting (CTW) and Maximising (CTM) Algorithms

The general implementation of the CTW and CTM algorithms in the MIC toolbox broadly follows the technical report of Willems and Tjalkens (1997), however, some aspects have been modified. It is recommended that the reader download this report from http://www.ele.tue.nl/ctw for more general background information.

Tree Structure

The information used to reconstruct the $2^d \times 2^r$ Markov transition table of model M_i is stored in a set of $2^r - 1$ binary trees, one of which is illustrated in Fig. 5. Following from the discussion in 2.1 each tree is indexed by a $k \leq r$ length string o corresponding to possible substrings of the observation string $\underline{x}_t$, and each node within a tree is indexed by a substring s of the full context $\underline{x}_{t-L}^{t-1}$. As is illustrated in Fig. 6, each node contains two internal counters $a_{s,o}$ and $b_{s,o}$ which count the number of times a 0 or a 1 have been observed after context s and observation bits o, as well as two internal ratios $\beta_{s,o}$ and $Q_{s,o}$ which are used respectively to weight the probabilities of the node with respect to its children nodes and to select between the node and its children when drawing probabilities. As a result each node in a tree can be used to generate the probability that $\underline{X}_t\{k\} = 1$ conditional on the history of the system, given by the context string s and on the previous $k-1$ bits of the observation string o.

Given a full context of length $d = rL$, one can identify a path in each tree (in bold in Fig. 5) linking the corresponding leaf node to the root node $\varnothing$.[13] Selecting which node on the path to draw the probability from involves a tradeoff between precision and conditioning. Nodes close to the leaf benefit from more conditioning information, as the context string s is longer, but suffer from not being observed very often. Nodes closer to the root will be visited more often and produce more precise probabilities, but suffer from weaker conditioning.[14] The intuition behind the effectiveness of the CTW and CTM algorithms is that they provide a method of optimally resolving this tradeoff and selecting the point on the leaf-to-root path which minimises prediction error.

Updating the Tree During Training

During the training stage, the simulated data from model M_i is used sequentially to update the trees. Starting at the beginning of the training data string, the first L observations $\underline{x}_1^L$ specify the d-length context string with the following r bits making up the observation string $\underline{x}_{L+1}$, as shown in Fig. 4. The bits in o are used to identify the r trees corresponding to the observation and the bits in s identify the nodes making up the leaf to root path.

For each of the r trees, the node on the path is updated in sequence, as illustrated in Fig. 6. Because leaf nodes do not have any children, there is no weighting to be performed, and the probabilities in the node are directly given by the Krichevsky and Trofimov (1981) estimates using counts $a_{s,o}$ and $b_{s,o}$:

$$P_{s,o}^{e}(X_t = 0) = \frac{a_{s,o} + \frac{1}{2}}{a_{s,o} + b_{s,o} + 1} \qquad P_{s,o}^{e}(X_t = 1) = \frac{b_{s,o} + \frac{1}{2}}{a_{s,o} + b_{s,o} + 1}. \tag{42}$$

The probabilities from the leaf node are used to calculate the initial value of the odds ratio η, which is send to the parent of the leaf node.

$$\eta = \frac{P_{s,o}^{e}(X_t = 0)}{P_{s,o}^{e}(X_t = 1)} = \frac{1 - P_{s,o}^{e}(X_t = 1)}{P_{s,o}^{e}(X_t = 1)}. \tag{43}$$

[13] In the interest of simplification, the tree in Fig. 5 is truncated at a depth $d = 3$.

[14] In particular, the root node $\varnothing$ is visited every time the tree is updated, however it corresponds to an unconditional probability as no historical conditioning information is used.

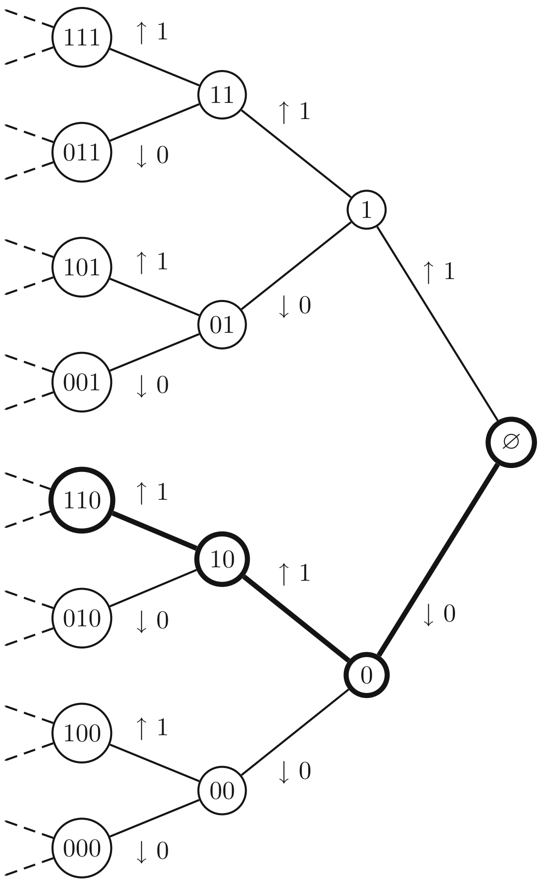

Fig. 5 A context tree, truncated at $d = 3$

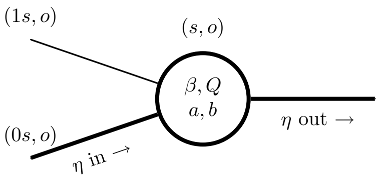

Fig. 6 Updating a tree node identified by context s and observation o

The incoming odds ratio are used to calculate the weighted probability given by the child node:[15]

$$P^w_{0s,o}(X_t = 0) = \frac{\eta_{in}}{1 + \eta_{in}} \quad P^w_{0s,o}(X_t = 1) = \frac{\eta_{in}}{1 + \eta_{in}}. \tag{44}$$

[15] For the parent of the leaf node, it is the case of course that $P^e_{0s,o} = P^w_{0s,o}$, however for nodes deeper in the tree this will no longer be the case.

These allow the η odds ratio to be updated, using the internal mixing ratio $\beta_{s,o}$, which weight the probabilities that $\underline{X}_t\{k\} = 1$ as measured in the current node (42) and as measured in the child node (44):

$$\eta_{out} = \frac{\beta_{s,o} P^e_{s,o}(X_t = 0) + P^w_{0s,o}(X_t = 0)}{\beta_{s,o} P^e_{s,o}(X_t = 1) + P^w_{0s,o}(X_t = 1)}. \tag{45}$$

The outgoing odds ratio η_{out} will become the incoming odds ratio η_{in} for the next node on the leaf to root path. Before proceeding to the next node, the algorithm updates the internal $\beta_{s,o}$, $Q_{s,o}$ ratios and $a_{s,o}$, bs, o counters. First of all, the internal $\beta_{s,o}$ ratio is updated as follows, in order to reflect the new observation, and the internal counter corresponding to the observation is incremented by one.

$$\begin{cases} \beta_{s,o} = \beta_{s,o} \dfrac{P^e_{s,o}(X_t = 0)}{P^w_{0s,o}(X_t = 0)}, & a_{s,o} = a_{s,o} + 1 \quad \text{if } x_t = 0, \\ \beta_{s,o} = \beta_{s,o} \dfrac{P^e_{s,o}(X_t = 1)}{P^w_{0s,o}(X_t = 1)}, & b_{s,o} = b_{s,o} + 1 \quad \text{if } x_t = 1. \end{cases} \tag{46}$$

Secondly, the Q-ratio is updated using $\beta_{s,o}$ and the Q-ratios of the two children of node (s, o), following the CTM procedure of Willems et al. (2006):

$$Q_{s,o} = \max\left(\frac{Q_{0s,o} Q_{1s,o}}{\beta_{s,o} + 1}, \frac{\beta_{s,o}}{\beta_{s,o} + 1}\right). \tag{47}$$

Once the node has been updated, the algorithm passes the updated value of η (45) to the next node on the path and updates it using (44)–(47). This process continues until the root node is reached for all r trees, at which point the algorithm moves on to the next context/observation pair $\underline{x}_2^{L+1}$, $\underline{x}_{L+2}$, the procedure continuing until all the training data has been processed.

Stage 2: Scoring the Empirical Data

The $N - L$ empirical observations are processed sequentially, starting with the first context/ observation pair $\underline{x}_2^{L+1}$, $\underline{x}_{L+2}$, and finishing with the last one, $\underline{x}_{N-L}^{N-1}$, $\underline{x}_N$. The conditional probabilities $P_{M_i}(\underline{X}_t = \underline{x}_t | \underline{x}_{t-L}^{t-1})$ required to score each observation are drawn from the context trees corresponding to model M_i.

Drawing Probabilities from the Tree

As was the case for the training data in stage 1, each context/observation pair is used to identify the r trees corresponding to the observation string and the leaf-to-root path corresponding to the context. Because each node on this path can potentially be used to generate a conditional probability, the context tree maximisation approach of Willems et al. (2006) is used to select the correct node on the path. Starting at the root of the tree, the $\beta_{s,o}$ ratio of the node is compared to product of the Q ratio of the children nodes as follows:

$$\begin{cases} \text{if } Q_{0s,o}Q_{1s,o} > \beta_{s,o} & \text{move to child node on path,} \\ \text{if } Q_{0s,o}Q_{1s,o} < \beta_{s,o} & \text{halt } \rightarrow \text{ current node optimal.} \end{cases} \tag{48}$$

Once the best node on the path s^* is identified, its $a_{s^*,o},\ b_{s^*,o}$ counters are used to calculate the probability of the kth bit being a ‘1’, $P_{M_i}(X_t\{k\} = 1 | \underset{\sqcup}{x}\,{}_{t-L}^{t-1} = s^*,\ \underset{\sqcup}{x}\,{}_t\{1, 2, \ldots, k-1\} = o)$, using the Krichevsky and Trofimov (1981) estimator. This is carried out for all r bits in the observation string, providing a r length vector of probabilities p:

$$p_i(k) = P_{M_i}\left(\underset{\sqcup}{X}\,{}_t\{k\}{=}1 \middle| \underset{\sqcup}{x}\,{}_{t-L}^{t-1},\ \underset{\sqcup}{x}\,{}_t\{1, 2, \ldots, k-1\}\right) = \frac{b_{s^*,o} + \frac{1}{2}}{a_{s^*,o} + b_{s^*,o} + 1}. \tag{49}$$

Scoring the Observations

The nested binary structure of the r-bits in an observation, explained in Sect. 1, means that the log score for the observation is simply sum of the individual binary log scores obtained using the vector of model probabilities p and observation bits $\underset{\sqcup}{x}\,{}_t\{k\}$:

$$\lambda_i\left(\underset{\sqcup}{X}\,{}_t \middle| \underset{\sqcup}{x}\,{}_{t-L}^{t-1}\right) = -\sum_{k=1}^{r}\left[\underset{\sqcup}{x}\,{}_t\{k\}\log_2 p_i(k) + \left(1 - \underset{\sqcup}{x}\,{}_t\{k\}\right)\log_2\left(1 - p_i(k)\right)\right]. \tag{50}$$

In addition to this, the $a_{s^*,o},\ b_{s^*,o}$ counters used to generate the vector of probabilities (49) are used to generate the marginal bound of the KT estimator (13) which is used in the MIC procedure to correct for the measurement bias induced by the stage 1 learning process:

$$\epsilon_i\left(\underset{\sqcup}{X}\,{}_t \middle| \underset{\sqcup}{x}\,{}_{t-L}^{t-1}\right) = \frac{1}{2}\sum_{o=\varnothing}^{\underset{\sqcup}{x}\,{}_t}\log_2\frac{a_{s*,o} + b_{s*,o} + 1}{a_{s*,o} + b_{s*,o}}. \tag{51}$$

Finally, an MDL complexity penalisation can be obtained by summing the log of the normalised counter observations for all the nodes in the tree:

$$\mathrm{X}_i^N = \sum_s\sum_o \gamma\left(\frac{N(a_{s,o}+b_{s,o})}{T}\right) \quad \gamma(x) = \begin{cases} 0 & \text{for } 0 \le x < 1, \\ \frac{1}{2}\log_2(x) + 1 & \text{for } x \ge 1. \end{cases} \tag{52}$$

Appendix 3: Extended Monte Carlo Results

Table 6 Monte Carlo analysis of MIC on ARMA models, $L = 3$, short T

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{15}$							
MIC$_i$, mean	25,226.93	30,933.42	25,186.39	25,248.77	25,235.21	26,383.53	25,298.7
P($\rho_i = \rho_i^*$)	0.137	1	0.227	0.287	0.298	1	0.704
P($\rho_i > \rho_i^*$)	0.863	–	0.206	0.144	0.292	0	0
P($\rho_i < \rho_i^*$)	–	0	0.567	0.569	0.41	0	0.296
ΔMIC$_{i,0}$, mean	–	5706.49	−40.54	21.84	8.28	1156.60	71.77
ΔMIC$_{i,0}$, $P_{2.5}$	–	5150.72	−157.41	−97.77	−113.83	884.81	−57.41
ΔMIC$_{i,0}$, $P_{97.5}$	–	6285.01	61.33	228.30	178.28	1509.14	206.68
P(ΔMIC$_{i,0} > 0$)	–	1	0.231	0.557	0.506	1	0.881
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.974	0	0.998	0.967	0.975	0	0.623
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.931	0	0.995	0.928	0.941	0	0.498
$T = 2^{16}$							
MIC$_i$, mean	24,950.47	30,748.75	24,982.44	25,003.14	24,993.01	26,357.2	24,986.35
P($\rho_i = \rho_i^*$)	0.504	1	0.227	0.302	0.195	1	0.23
P($\rho_i > \rho_i^*$)	0.496	–	0.609	0.332	0.48	0	0
P($\rho_i < \rho_i^*$)	–	0	0.164	0.366	0.325	0	0.77
ΔMIC$_{i,0}$, mean	–	5798.27	31.97	52.67	42.53	1406.73	35.87
ΔMIC$_{i,0}$, $P_{2.5}$	–	5231.67	−58.52	−39.57	−54.23	1093.81	−51.24
ΔMIC$_{i,0}$, $P_{97.5}$	–	6373.71	133.72	157.08	171.59	1855.40	127.10
P(ΔMIC$_{i,0} > 0$)	–	1	0.748	0.871	0.782	1	0.767
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	1	0	0.967	0.917	0.977	0	0.950
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.998	0	0.939	0.856	0.945	0	0.917
$T = 2^{17}$							
MIC$_i$, mean	24,736.78	30,646.67	24,735.29	24,786.95	24,774.35	26,423.05	24,794.21
P($\rho_i = \rho_i^*$)	0.387	1	0.308	0.371	0.269	1	0.446
P($\rho_i > \rho_i^*$)	0.613	–	0.253	0.292	0.401	0	0
P($\rho_i < \rho_i^*$)	–	0	0.439	0.337	0.33	0	0.554
ΔMIC$_{i,0}$, mean	–	5909.88	−1.49	50.16	37.57	1686.26	57.42
ΔMIC$_{i,0}$, $P_{2.5}$	–	5353.38	−93.61	−41.06	−61.91	1331.06	−26.55
ΔMIC$_{i,0}$, $P_{97.5}$	–	6470.10	98.61	140.95	182.14	2199.09	146.06
P(ΔMIC$_{i,0} > 0$)	–	1	0.468	0.884	0.719	1	0.91
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.984	0	0.991	0.847	0.952	0	0.813
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.970	0	0.981	0.768	0.903	0	0.728
$T = 2^{18}$							
MIC$_i$, mean	24,495.31	30,745.8	24,515.46	24,598.97	24,569.58	26,449.52	24,633.98
P($\rho_i = \rho_i^*$)	0.703	1	0.633	0.598	0.65	1	0.761

Table 6 continued

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$\mathrm{P}(\rho_i > \rho_i^*)$	0.297	–	0.092	0.192	0.254	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.275	0.21	0.096	0	0.239
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6250.49	20.15	103.66	74.27	1954.22	138.67
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	5676.61	−53.52	26.11	−4.29	1566.46	57.56
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	6833.33	98.56	185.57	158.62	2549.27	221.66
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.719	0.994	0.969	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.995	0	0.949	0.412	0.661	0	0.192
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.988	0	0.913	0.288	0.532	0	0.125

$P(\rho_i \overset{\leq}{\underset{>}{=}} \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

Table 7 Monte Carlo analysis of MIC on ARMA models, $L = 3$, long T

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{19}$							
MIC_i, mean	24,308.29	30,762.66	24,326.00	24,445.68	24,404.13	26,395.72	24,499.15
$\mathrm{P}(\rho_i = \rho_i^*)$	0.687	1	0.665	0.78	0.85	1	0.887
$\mathrm{P}(\rho_i > \rho_i^*)$	0.313	–	0.024	0.106	0.122	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.311	0.114	0.028	0	0.113
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6454.38	17.71	137.39	95.84	2087.43	190.86
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	5878.78	−48.45	63.14	26.37	1654.28	115.25
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	7042.87	85.21	212.30	169.73	2704.73	272.93
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.687	0.999	0.993	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.998	0	0.924	0.094	0.324	0	0.011
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.988	0	0.876	0.049	0.230	0	0.003
$T = 2^{20}$							
MIC_i, mean	24,171.54	30,708.71	24,181.76	24,308.26	24,260.10	26,328.56	24,391.47
$\mathrm{P}(\rho_i = \rho_i^*)$	0.626	1	0.612	0.921	0.924	1	0.977
$\mathrm{P}(\rho_i > \rho_i^*)$	0.374	–	0.015	0.021	0.059	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.373	0.058	0.017	0	0.023
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6537.17	10.21	136.72	88.56	2157.02	219.93
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	5947.40	−47.96	68.34	22.15	1725.87	141.59
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	7162.27	72.85	203.58	158.51	2820.34	302.93
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.627	1	0.996	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.995	0	0.946	0.041	0.309	0	0
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.976	0	0.902	0.023	0.210	0	0

Table 7 continued

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{21}$							
MIC$_i$, mean	24,061.74	30,709.54	24,067.33	24,210.05	24,152.46	26,285.81	24,316.84
$\mathrm{P}(\rho_i = \rho_i^*)$	0.575	1	0.574	0.972	0.981	1	0.99
$\mathrm{P}(\rho_i > \rho_i^*)$	0.425	–	0.001	0.01	0.018	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.425	0.018	0.001	0	0.01
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6647.80	5.59	148.31	90.77	2224.08	255.11
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	6025.87	−45.61	83.31	32.68	1795.72	173.36
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	7282.78	59.00	211.67	149.48	2870.99	342.03
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.575	1	1	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.986	0	0.959	0.008	0.170	0	0
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.966	0	0.921	0.003	0.115	0	0
$T = 2^{22}$							
MIC$_i$, mean	23,983.70	30,703.92	23,989.93	24,136.63	24,071.69	26,300.02	24,261.73
$\mathrm{P}(\rho_i = \rho_i^*)$	0.605	1	0.602	0.993	0.993	1	0.996
$\mathrm{P}(\rho_i > \rho_i^*)$	0.395	–	0.003	0.004	0.003	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.395	0.003	0.004	0	0.004
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6720.22	6.24	152.93	87.99	2316.32	278.03
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	6080.26	−36.15	93.82	37.51	1853.71	201.62
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	7367.55	52.34	211.11	142.37	3018.52	361.99
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.605	1	0.999	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.991	0	0.939	0.002	0.110	0	0
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.978	0	0.899	0.001	0.057	0	0

$P(\rho_i \lesseqgtr \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

Table 8 Monte Carlo analysis of MIC on ARMA models, $L = 2$, short T

$r = 7$ $d = 14$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{15}$							
MIC$_i$, mean	25,237.59	30,950.94	25,192.1	25,245.42	25,240.6	26,386.74	25,295.39
$\mathrm{P}(\rho_i = \rho_i^*)$	0.1	1	0.211	0.24	0.291	1	0.652
$\mathrm{P}(\rho_i > \rho_i^*)$	0.9	–	0.224	0.137	0.314	0	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0	0.565	0.623	0.395	0	0.348
$\Delta\mathrm{MIC}_{i,0}$, mean	–	5713.35	−45.49	7.83	3.01	1149.15	57.80
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	5162.28	−166.16	−114.43	−116.38	878.39	−72.27
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	6297.20	57.85	216.45	172.40	1504.34	195.88

Table 8 continued

$r = 7$ $d = 14$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
P(ΔMIC$_{i,0} > 0$)	–	1	0.198	0.472	0.473	1	0.835
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.966	0	0.998	0.982	0.977	0	0.691
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.910	0	0.995	0.951	0.942	0	0.567
$T = 2^{16}$							
MIC$_i$, mean	24,970.78	30,759.39	24,989.92	24,997.91	24,997.63	26,360.39	24,973.88
P($\rho_i = \rho_i^*$)	0.306	1	0.179	0.274	0.184	1	0.148
P($\rho_i > \rho_i^*$)	0.694	–	0.652	0.267	0.505	0	0
P($\rho_i < \rho_i^*$)	–	0	0.169	0.459	0.311	0	0.852
ΔMIC$_{i,0}$, mean	–	5788.60	19.14	27.13	26.85	1389.60	3.10
ΔMIC$_{i,0}$, $P_{2.5}$	–	5222.24	−68.30	−66.12	−66.46	1079.23	−82.85
ΔMIC$_{i,0}$, $P_{97.5}$	–	6364.14	120.25	123.74	152.44	1855.98	86.12
P(ΔMIC$_{i,0} > 0$)	–	1	0.657	0.724	0.667	1	0.534
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.994	0	0.980	0.958	0.988	0	0.987
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.986	0	0.957	0.921	0.970	0	0.975
$T = 2^{17}$							
MIC$_i$, mean	24,801.11	30,732.69	24,791.17	24,799.82	24,795.46	26,420.98	24,806.15
P($\rho_i = \rho_i^*$)	0.148	1	0.225	0.227	0.135	1	0.297
P($\rho_i > \rho_i^*$)	0.852	–	0.512	0.194	0.341	0	0
P($\rho_i < \rho_i^*$)	–	0	0.263	0.579	0.524	0	0.703
ΔMIC$_{i,0}$, mean	–	5931.58	−9.94	−1.28	−5.65	1619.87	5.04
ΔMIC$_{i,0}$, $P_{2.5}$	–	5379.29	−90.51	−88.71	−96.19	1266.63	−73.04
ΔMIC$_{i,0}$, $P_{97.5}$	–	6491.13	81.18	76.45	138.82	2126.06	83.33
P(ΔMIC$_{i,0} > 0$)	–	1	0.377	0.496	0.389	1	0.555
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.973	0	0.993	0.963	0.994	0	0.956
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.942	0	0.976	0.930	0.989	0	0.919
$T = 2^{18}$							
MIC$_i$, mean	24,630.97	30,806.13	24,659.9	24,659.23	24,654.89	26,447.94	24,690.49
P($\rho_i = \rho_i^*$)	0.596	1	0.221	0.274	0.25	1	0.634
P($\rho_i > \rho_i^*$)	0.404	–	0.668	0.123	0.324	0	0
P($\rho_i < \rho_i^*$)	–	0	0.111	0.603	0.426	0	0.366
ΔMIC$_{i,0}$, mean	–	6175.16	28.93	28.27	23.92	1816.97	59.52
ΔMIC$_{i,0}$, $P_{2.5}$	–	5605.29	−30.71	−35.79	−41.49	1421.16	−9.22
ΔMIC$_{i,0}$, $P_{97.5}$	–	6765.68	94.65	91.59	88.77	2415.88	128.09
P(ΔMIC$_{i,0} > 0$)	–	1	0.837	0.803	0.771	1	0.953
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.998	0	0.953	0.939	0.965	0	0.782
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.996	0	0.912	0.885	0.922	0	0.676

$P(\rho_i \lesseqgtr \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

Table 9 Monte Carlo analysis of MIC on ARMA models, $L = 2$, long T

$r = 7$ $d = 14$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{19}$							
MIC$_i$, mean	24,537.20	30,870.74	24,547.77	24,560.59	24,562.01	26,469.34	24,600.05
P($\rho_i = \rho_i^*$)	0.519	1	0.314	0.31	0.285	1	0.784
P($\rho_i > \rho_i^*$)	0.481	–	0.409	0.084	0.424	0	0
P($\rho_i < \rho_i^*$)	–	0	0.277	0.606	0.291	0	0.216
ΔMIC$_{i,0}$, mean	–	6333.54	10.57	23.39	24.81	1932.14	62.84
ΔMIC$_{i,0}$, $P_{2.5}$	–	5744.34	−37.37	−32.78	−31.30	1508.02	−0.56
ΔMIC$_{i,0}$, $P_{97.5}$	–	6925.50	62.52	79.29	80.81	2564.46	120.17
P(ΔMIC$_{i,0} > 0$)	–	1	0.647	0.793	0.821	1	0.972
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.998	0	0.976	0.932	0.934	0	0.685
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.992	0	0.955	0.880	0.885	0	0.573
$T = 2^{20}$							
MIC$_i$, mean	24,471.66	30,868.36	24,479.13	24,484.99	24,490.77	26,462.23	24,531.14
P($\rho_i = \rho_i^*$)	0.471	1	0.329	0.26	0.247	1	0.862
P($\rho_i > \rho_i^*$)	0.529	–	0.425	0.034	0.479	0	0
P($\rho_i < \rho_i^*$)	–	0	0.246	0.706	0.274	0	0.138
ΔMIC$_{i,0}$, mean	–	6396.70	7.47	13.33	19.11	1990.57	59.48
ΔMIC$_{i,0}$, $P_{2.5}$	–	5807.51	−35.65	−28.95	−29.64	1561.01	2.07
ΔMIC$_{i,0}$, $P_{97.5}$	–	7006.12	49.96	58.07	67.26	2636.28	114.04
P(ΔMIC$_{i,0} > 0$)	–	1	0.637	0.728	0.779	1	0.98
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.999	0	0.981	0.945	0.938	0	0.597
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.989	0	0.963	0.902	0.870	0	0.488
$T = 2^{21}$							
MIC$_i$, mean	24,428.22	30,917.28	24,432.73	24,440.82	24,447.83	26,445.25	24,497.17
P($\rho_i = \rho_i^*$)	0.471	1	0.348	0.25	0.258	1	0.953
P($\rho_i > \rho_i^*$)	0.529	–	0.337	0.009	0.53	0	0
P($\rho_i < \rho_i^*$)	–	0	0.315	0.741	0.212	0	0.047
ΔMIC$_{i,0}$, mean	–	6489.06	4.51	12.60	19.61	2017.03	68.95
ΔMIC$_{i,0}$, $P_{2.5}$	–	5883.99	−28.48	−24.82	−19.88	1592.25	21.36
ΔMIC$_{i,0}$, $P_{97.5}$	–	7113.15	40.78	52.43	58.81	2657.47	117.99
P(ΔMIC$_{i,0} > 0$)	–	1	0.589	0.736	0.836	1	0.997
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.995	0	0.983	0.926	0.913	0	0.369
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.993	0	0.961	0.886	0.834	0	0.268
$T = 2^{22}$							
MIC$_i$, mean	24,408.03	31,018.51	24,413.63	24,426.75	24,431.41	26,476.42	24,480.06
P($\rho_i = \rho_i^*$)	0.59	1	0.433	0.324	0.263	1	0.979
P($\rho_i > \rho_i^*$)	0.41	–	0.251	0.004	0.586	0	0
P($\rho_i < \rho_i^*$)	–	0	0.316	0.672	0.151	0	0.021

Table 9 continued

$r = 7$ $d = 14$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$\Delta\mathrm{MIC}_{i,0}$, mean	–	6610.48	5.60	18.72	23.39	2068.39	72.03
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	5996.51	−23.45	−10.61	−7.85	1603.77	27.79
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	7239.11	33.91	49.49	55.43	2758.82	119.04
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.661	0.896	0.923	1	1
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.999	0	0.959	0.828	0.779	0	0.251
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.993	0	0.921	0.742	0.686	0	0.171

$P(\rho_i \lesseqgtr \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

Table 10 Monte Carlo analysis of MIC on ARMA models, $L = 3$, short T, $N = 500$

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{15}$							
MIC_i, mean	1538.93	1887.44	1536.64	1540.54	1539.76	1609.25	1543.40
$\mathrm{P}(\rho_i = \rho_i^*)$	0.198	0.999	0.223	0.221	0.196	0.995	0.362
$\mathrm{P}(\rho_i > \rho_i^*)$	0.802	–	0.484	0.178	0.365	0.001	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0.001	0.293	0.601	0.439	0.004	0.638
$\Delta\mathrm{MIC}_{i,0}$, mean	–	348.52	−2.28	1.61	0.83	70.33	4.47
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	222.22	−32.21	−25.42	−24.70	22.18	−22.95
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	489.37	20.86	26.36	27.96	165.52	29.93
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.426	0.534	0.496	0.999	0.645
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.980	0	0.991	0.967	0.985	0.219	0.918
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.965	0	0.978	0.936	0.966	0.152	0.875
$T = 2^{16}$							
MIC_i, mean	1522.30	1876.45	1523.84	1525.84	1524.39	1607.48	1524.36
$\mathrm{P}(\rho_i = \rho_i^*)$	0.248	0.996	0.193	0.262	0.193	0.996	0.242
$\mathrm{P}(\rho_i > \rho_i^*)$	0.752	–	0.586	0.25	0.372	0.004	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0.004	0.221	0.488	0.435	0	0.758
$\Delta\mathrm{MIC}_{i,0}$, mean	–	354.16	1.54	3.54	2.10	85.19	2.06
$\Delta\mathrm{MIC}_{i,0},\ P_{2.5}$	–	229.07	−20.32	−18.47	−20.50	33.87	−20.61
$\Delta\mathrm{MIC}_{i,0},\ P_{97.5}$	–	495.76	24.00	26.50	31.68	199.75	23.97
$\mathrm{P}(\Delta\mathrm{MIC}_{i,0} > 0)$	–	1	0.552	0.641	0.548	1	0.581
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.992	0	0.987	0.963	0.987	0.159	0.951
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.973	0	0.974	0.92	0.968	0.096	0.916
$T = 2^{17}$							
MIC_i, mean	1509.10	1870.21	1508.91	1512.25	1511.24	1611.38	1512.26
$\mathrm{P}(\rho_i = \rho_i^*)$	0.259	0.995	0.239	0.235	0.201	0.995	0.293

Table 10 continued

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
P($\rho_i > \rho_i^*$)	0.741	–	0.494	0.251	0.398	0.005	0
P($\rho_i < \rho_i^*$)	–	0.005	0.267	0.514	0.401	0	0.707
ΔMIC$_{i,0}$, mean	–	361.11	−0.19	3.15	2.15	102.28	3.17
ΔMIC$_{i,0}$, $P_{2.5}$	–	240.21	−20.49	−17.63	−20.52	45.31	−17.70
ΔMIC$_{i,0}$, $P_{97.5}$	–	501.82	21.03	23.05	29.58	238.17	23.71
P(ΔMIC$_{i,0} > 0$)	–	1	0.486	0.619	0.564	1	0.603
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.991	0	0.991	0.944	0.980	0.119	0.941
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.977	0	0.978	0.901	0.962	0.07	0.905
$T = 2^{18}$							
MIC$_i$, mean	1494.14	1876.38	1495.38	1500.99	1499.08	1612.77	1502.50
P($\rho_i = \rho_i^*$)	0.362	0.993	0.27	0.29	0.235	0.993	0.412
P($\rho_i > \rho_i^*$)	0.638	–	0.429	0.278	0.415	0.007	0
P($\rho_i < \rho_i^*$)	–	0.007	0.301	0.432	0.35	0	0.588
ΔMIC$_{i,0}$, mean	–	382.24	1.25	6.85	4.94	118.64	8.37
ΔMIC$_{i,0}$, $P_{2.5}$	–	253.41	−16.34	−13.47	−14.78	55.92	−12.32
ΔMIC$_{i,0}$, $P_{97.5}$	–	524.57	21.05	28.29	25.66	279.69	30.81
P(ΔMIC$_{i,0} > 0$)	–	1	0.543	0.767	0.684	1	0.789
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.993	0	0.982	0.904	0.945	0.105	0.885
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.983	0	0.968	0.857	0.896	0.067	0.831

$P(\rho_i \overset{\leq}{\underset{>}{=}} \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

Table 11 Monte Carlo analysis of MIC on ARMA models, $L = 3$, long T, $N = 500$

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{20}$							
MIC$_i$, mean	1482.34	1877.35	1484.14	1491.46	1489.17	1608.99	1494.72
P($\rho_i = \rho_i^*$)	0.465	0.991	0.326	0.359	0.292	0.991	0.51
P($\rho_i > \rho_i^*$)	0.535	–	0.349	0.261	0.419	0.009	0
P($\rho_i < \rho_i^*$)	–	0.009	0.325	0.38	0.289	0	0.49
ΔMIC$_{i,0}$, mean	–	395.01	1.80	9.12	6.83	126.64	12.37
ΔMIC$_{i,0}$, $P_{2.5}$	–	264.05	−15.51	−9.47	−10.80	60.53	−8.01
ΔMIC$_{i,0}$, $P_{97.5}$	–	545.30	19.24	28.48	26.38	294.40	34.23
P(ΔMIC$_{i,0} > 0$)	–	1	0.598	0.839	0.778	1	0.885
P($M_i \in \hat{\mathcal{M}}_{0.95}$)	0.987	0	0.982	0.865	0.918	0.102	0.844
P($M_i \in \hat{\mathcal{M}}_{0.9}$)	0.979	0	0.965	0.791	0.860	0.062	0.756

Table 11 continued

$r = 7$ $d = 21$	M_0 True	M_1 No AR	M_2 No AR-2	M_3 No MA	M_4 No MA-2	M_5 No ARCH	M_6 No ARCH-2
$T = 2^{20}$							
MIC$_i$, mean	1474.23	1873.88	1475.47	1483.05	1480.34	1604.53	1488.38
$\mathrm{P}(\rho_i = \rho_i^*)$	0.453	0.991	0.336	0.403	0.345	0.991	0.606
$\mathrm{P}(\rho_i > \rho_i^*)$	0.547	–	0.318	0.229	0.349	0.009	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0.009	0.346	0.368	0.306	0	0.394
ΔMIC$_{i,0}$, mean	–	399.65	1.24	8.82	6.11	130.30	14.15
ΔMIC$_{i,0}$, $P_{2.5}$	–	267.28	−14.49	−8.86	−10.49	65.20	−3.66
ΔMIC$_{i,0}$, $P_{97.5}$	–	549.11	16.54	25.55	23.60	303.07	34.49
P(ΔMIC$_{i,0} > 0$)	–	1	0.567	0.842	0.767	1	0.921
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.996	0	0.983	0.852	0.926	0.090	0.798
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.990	0	0.967	0.766	0.875	0.057	0.694
$T = 2^{21}$							
MIC$_i$, mean	1467.89	1873.92	1468.53	1477.45	1473.38	1601.49	1483.64
$\mathrm{P}(\rho_i = \rho_i^*)$	0.444	0.993	0.357	0.462	0.418	0.993	0.663
$\mathrm{P}(\rho_i > \rho_i^*)$	0.556	–	0.264	0.248	0.299	0.007	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0.007	0.379	0.29	0.283	0	0.337
ΔMIC$_{i,0}$, mean	–	406.03	0.65	9.56	5.49	133.61	15.75
ΔMIC$_{i,0}$, $P_{2.5}$	–	269.33	−12.65	−7.01	−9.24	69.17	−3.04
ΔMIC$_{i,0}$, $P_{97.5}$	–	558.25	14.93	25.41	20.90	265.24	36.96
P(ΔMIC$_{i,0} > 0$)	–	1	0.549	0.890	0.785	1	0.942
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.993	0	0.978	0.799	0.903	0.085	0.745
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.977	0	0.954	0.699	0.840	0.052	0.636
$T = 2^{22}$							
MIC$_i$, mean	1463.30	1873.51	1463.78	1473.18	1468.70	1602.26	1480.55
$\mathrm{P}(\rho_i = \rho_i^*)$	0.45	0.99	0.378	0.501	0.454	0.99	0.712
$\mathrm{P}(\rho_i > \rho_i^*)$	0.55	–	0.228	0.233	0.261	0.01	0
$\mathrm{P}(\rho_i < \rho_i^*)$	–	0.01	0.394	0.266	0.285	0	0.288
ΔMIC$_{i,0}$, mean	–	410.21	0.48	9.88	5.40	138.96	17.25
ΔMIC$_{i,0}$, $P_{2.5}$	–	268.40	−10.94	−4.91	−7.51	71.72	−0.94
ΔMIC$_{i,0}$, $P_{97.5}$	–	568.72	12.12	25.18	18.41	295.64	39.28
P(ΔMIC$_{i,0} > 0$)	–	1	0.528	0.903	0.797	1.000	0.965
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.95})$	0.992	0	0.985	0.740	0.895	0.092	0.724
$\mathrm{P}(M_i \in \hat{\mathcal{M}}_{0.9})$	0.981	0	0.962	0.650	0.818	0.053	0.584

$P(\rho_i \lesseqqgtr \rho_i^*)$ Monte Carlo probabilities of MIC rank ρ being equal to, greater or smaller than AIC rank ρ^*, $\Delta MIC_{i,0}$ mean MIC difference between M_i and the true model M_0, with 2.5 and 97.5 % percentiles, $P(M_i \in \hat{\mathcal{M}}_{1-\alpha})$ Monte Carlo probability of M_i being included in the MCS at α% confidence

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *AC–19*, 716–723.

Basharin, G. P. (1959). On a statistical estimate for the entropy of a sequence of independent random variables. *Theory of Probability and Its Applications*, *4*(3), 333–336.

Carlton, A. (1969). On the bias of information estimates. *Psychological Bulletin*, *71*(2), 108–109.

Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory*. New York: Wiley.

Dawid, H., & Fagiolo, G. (2008). Agent-based models for economic policy design: Introduction to the special issue. *Journal of Economic Behavior and Organization*, *67*, 351–354.

Deissenberg, C., Van Der Hoog, S., & Dawid, H. (2008). EURACE: A massively parallel agent-based model of the European economy. *Applied Mathematics and Computation*, *204*(2), 541–552.

Del Negro, M., & Schorfheide, F. (2006). How good is what you've got? DGSE-VAR as a toolkit for evaluating DGSE models. *Economic Review: Federal Reserve Bank of Atlanta*, *91*, 21–337.

Del Negro, M., & Schorfheide, F. (2011). Chapter: Bayesian Macroeconometrics. In *The Oxford handbook of Bayesian econometrics*. Oxford: Oxford University Press.

Del Negro, M., Schorfheide, F., Smets, F., & Wouters, R. (2007). On the fit of new Keynesian models. *Journal of Business and Economic Statistics*, *25*, 123–143.

Dosi, G., Fagiolo, G., & Roventini, A. (2010). Schumpeter meeting Keynes: A policy-friendly model of endogenous growth and business cycles. *Journal of Economic Dynamics and Control*, *34*(9), 1748–1767.

Dosi, G., Fagiolo, G., Napoletano, M., & Roventini, A. (2013). Income distribution, credit and fiscal policies in an agent-based Keynesian model. *Journal of Economic Dynamics and Control*, *37*(8), 1598–1625.

Dosi, G., Fagiolo, G., Napoletano, M., Roventini, A., & Treibich, T. (2015). Fiscal and monetary policies in complex evolving economies. *Journal of Economic Dynamics and Control*, *52*, 166–189.

Elias, P. (1975). Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, *IT–21*, 194–203.

Fabretti, A. (2014). A Markov chain approach to ABM calibration. In F. J. Miguel Quesada, F. Amblard, J. A. Barcel, & M. Madella (Eds.), *Advances in computational social science and social simulation*. Barcelona: Autonomous University of Barcelona.

Fagiolo, G., & Roventini, A. (2012). Macroeconomic policy in DSGE and agent-based models. *Revue de l'OFCE*, *5*, 67–116.

Fagiolo, G., Moneta, A., & Windrum, P. (2007). A critical guide to empirical validation of agent-based models in economics: Methodologies, procedures, and open problems. *Computational Economics*, *30*, 195–226.

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*, 359–378.

Grünewald, P. D. (2007). *The minimum description length principle*. Cambridge: MIT Press.

Hansen, M. M., & Yu, B. (2001). Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, *96*, 746–774.

Hansen, P. R., Lunde, A., & Nason, J. M. (2011). The model confidence set. *Econometrica*, *79*, 453–497.

Holcombe, M., Chin, S., Cincotti, S., Raberto, M., Teglio, A., Coakley, S., et al. (2013). Large-scale modelling of economic systems. *Complex Systems*, *22*(2), 175–191.

Kirman, A. (1993). Ants, rationality and recruitment. *Quarterly Journal of Economics*, *108*, 137–156.

Kopecky, K. A., & Suen, R. M. (2010). Finite state Markov-chain approximations to highly persistent processes. *Review of Economic Dynamics*, *13*(3), 701–714.

Krichevsky, R. E., & Trofimov, V. K. (1981). The performance of universal encoding. *IEEE Transactions on Information Theory*, *IT–27*, 629–636.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, *22*, 79–86.

Lamperti, F. (2015). *An information theoretic criterion for empirical validation of time series models*. LEM working paper series 2015/02.

Marks, R. (2010). Comparing two sets of time-series: The state similarity measure. In *2010 Joint statistical meetings proceedings—Statistics: A key to innovation in a data-centric world* (pp. 539–551). Alexandria, VA: Statistical Computing Section, American Statistical Association.

Marks, R. E. (2013). Validation and model selection: Three similarity measures compared. *Complexity Economics*, *2*(1), 41–61.

Miller, G. A. (1955). Note on the bias of information estimates. *Information Theory in Psychology: Problems and Methods*, *2*, 95–100.

Panzeri, S., & Treves, A. (1996). Analytical estimates of limited sampling biases in different information measures. *Network: Computation in Neural Systems*, *7*, 87–107.

Politis, D. N., & Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical Association*, *89*, 1303–1313.

Politis, D. N., & White, H. (2004). Automatic block-length selection for the dependent bootstrap. *Econometric Reviews*, *23*, 53–70.

Rissanen, J. (1976). Generalized Kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, *20*, 198–203.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*, 465–471.

Rissanen, J. (1984). Universal coding, information, prediction and estimation. *IEEE Transactions on Information Theory*, *IT–30*, 629–636.

Rissanen, J. (1986). Complexity of strings in the class of Markov sources. *IEEE Transactions on Information Theory*, *IT–32*, 526–532.

Rissanen, J., & Langdon, G. G. J. (1979). Modeling by shortest data description. *IBM Journal of Research and Development*, *28*, 149–162.

Roulston, M. S. (1999). Estimating the errors on measured entropy and mutual information. *Physica D*, *125*, 285–294.

Schorfheide, F. (2000). Loss function-based evaluation of DSGE models. *Journal of Applied Econometrics*, *15*, 645–670.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*, 461–464.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*, 379–423.

Tauchen, G. (1986a). Finite state Markov-chain approximations to univariate and vector autoregressions. *Economics Letters*, *20*(2), 177–181.

Tauchen, G. (1986b). Statistical properties of generalized method-of-moments estimators of structural parameters obtained from financial market data. *Journal of Business and Economic Statistics*, *4*(4), 397–416.

Van Der Hoog, S., Deissenberg, C., & Dawid, H. (2008). Production and finance in EURACE. In *Complexity and artificial markets* (pp. 147–158). Berlin: Springer.

White, H. (2000). A reality check for data snooping. *Econometrica*, *68*, 1097–1126.

Willems, F. M. J., & Tjalkens, T. J. (1997). *Complexity reduction of the context-tree weighting algorithm: A study for KPN research*. EIDMA Report RS.97.01.

Willems, F. M. J., Shtarkov, Y. M., & Tjalkens, T. J. (1995). The context-tree weighting method: Basic properties. *IEEE Transactions on Information Theory*, *IT–41*, 653–664.

Willems, F. M. J., Tjalkens, T. J., & Ignatenko, T. (2006) Context-tree weighting and maximizing: Processing betas. In *Proceedings of the inaugural workshop of the ITA (information theory and its applications).*