An introduction to the *gets* package[1]

Felix Pretis[2], J. James Reade[3] and Genaro Sucarrat[4]

This version: 3rd September 2022

[Original version: 3 September 2018]

---

[1]This vignette is a minor modification of Pretis *et al.* (2018).

[2]University of Victoria, Department of Economics & INET at the Oxford Martin School, University of Oxford. URL: http://www.felixpretis.org/

[3]Programme for Economic Modelling at the Oxford Martin School & Department of Economics, University of Reading. URL: https://sites.google.com/site/jjamesreade/

[4]Department of Economics, BI Norwegian Business School. URL: https://www.sucarrat.net/

Summary

This vignette provides an overview of the R package **gets**, which contains facilities for automated general-to-specific (GETS) modeling of the mean and variance of a regression, and indicator saturation (IS) methods for the detection and modeling of outliers and structural breaks. The mean can be specified as an autoregressive model with covariates (an "AR-X" model), and the variance can be specified as an autoregressive log-variance model with covariates (a "log-ARCH-X" model). The covariates in the two specifications need not be the same, and the classical linear regression model is obtained as a special case when there is no dynamics, and when there are no covariates in the variance equation. The four main functions of the package are `arx()`, `getsm()`, `getsv()` and `isat()`. The first function estimates an AR-X model with log-ARCH-X errors. The second function undertakes GETS modeling of the mean specification of an 'arx' object. The third function undertakes GETS modeling of the log-variance specification of an 'arx' object. The fourth function undertakes GETS modeling of an indicator-saturated mean specification allowing for the detection of outliers and structural breaks. The usage of two convenience functions for export of results to EViews and STATA are illustrated, and LaTeX code of the estimation output can readily be generated.

1 Introduction

General-to-specific (GETS) modeling combines well-known ingredients: backwards elimination, single and multiple hypothesis testing, goodness-of-fit measures and diagnostics tests. The way these are combined by GETS modeling enables rival theories and models to be tested against each other, ultimately resulting in a parsimonious, statistically valid model that explains the characteristics of the data being investigated. The methodology thus provides a systematic and coherent approach to model development and maintenance, cumulative research and scientific progress. This paper provides an overview of the R (R Core Team, 2018) package **gets** (Sucarrat *et al.*, 2018), which contains facilities for automated general-to-specific (GETS) modeling of the mean and variance of cross-sectional and time-series regressions, and indicator saturation (IS) methods for the detection and modeling of outliers and structural breaks in the mean.

The origins of GETS modeling can be traced back to Denis Sargan and the London School of Economics (LSE) during the 1960s, see Hendry (2003) and Mizon (1995). However, it was not until the 1980s and 1990s that the methodology gained widespread acceptance and usage in economics, with David F. Hendry in particular being a main proponent, see the two-volume article collection by Campos *et al.* (2005) for a comprehensive overview of the GETS methodology. An important software-contribution to GETS modeling was made in 1999, when Hoover and Perez (1999) re-visited the data-mining experiment of Lovell (1983). Hoover and Perez (1999) showed that automated multi-path GETS modeling substantially improved upon the then (in economics) popular model selection strategies. In the study of Hoover and Perez (1999), purpose-specific but limited MATLAB (The MathWorks Inc., 2017) code was used in the simulations.[5] Subsequently, further improvements were achieved in the commercial software packages **PcGets** (Hendry and Krolzig, 2001) and in its successor **Autometrics** (Doornik and Hendry, 2007a). In particular, indicator-saturation methods for the detection of outliers and structural breaks proposed by Hendry *et al.* (2007) were added to **Autometrics** in 2008, see Doornik (2009). Another milestone was reached in 2011, when the R package **AutoSEARCH** (Sucarrat, 2015a) was published on the Comprehensive R Archive Network (CRAN). The package, whose code was developed based on Sucarrat and Escribano (2012), offered automated GETS modeling of conditional variance specifications within the log-ARCH-X class of models. The R package **gets**, available from CRAN since October 2014, is the successor of **AutoSEARCH**. The **gets** package, at the time of writing, is the only statistical software that offers GETS modeling of the conditional variance of a regression, in addition to GETS modeling of the mean of a regression, and indicator saturation (IS) methods for the detection of breaks of outliers structural breaks in the mean of a regression using impulses (IIS), step (SIS; see Castle *et al.* 2015) as well as trend indicators (TIS).

This paper provides an overview of the **gets** package. The main model class under consideration is the autoregressive (AR) model with exponential autoregressive conditional heteroscedastic (ARCH) variance, possibly with additional covariates in the mean or variance equations, or in both. In short, the AR-X model with a log-ARCH-X error term, where the "X" refers to the covariates (the covariates need not be the same in the mean and variance specifications). It should be underlined, however, that **gets** is not limited to time-series models

---

[5]The code is limited in that it allows for a maximum of 10 paths to be searched, and because there is no user manual nor help-system available. The data and MATLAB code is available from `http://www.feweb.vu.nl/econometriclinks/journal/volume2/HooverKD_PerezSJ/data_and_code/`.

(see Section 2.3): Static models (e.g., cross-sectional or panel) can be estimated by specifying the regression without dynamics. The next section, Section 2, provides an overview of GETS modeling and its alternatives, and outlines the principles that guides the development of **gets**. Section 3 contains a note on the advantage of providing the data with time-series attributes – if the data are indeed time-series, since this is useful for the estimation of dynamic models, output and graphing. Section 4 contains an overview of the AR-X model with log-ARCH-X errors, explains how it can be simulated, and illustrates how it can be estimated with the `arx` function. Section 5 illustrates how GETS modeling can be undertaken with the `getsm` and `getsv` functions. The first undertakes GETS modeling of the mean specification, whereas the second undertakes GETS modeling of the log-variance specification. Section 6 introduces the `isat` function for indicator saturation methods. Section 7 illustrates how two convenience functions, `eviews` and `stata`, facilitate GETS modeling by users of EViews (IHS Markit, 2017) or STATA (StataCorp, 2017), i.e., the two most popular commercial software packages in econometrics. The section also briefly alludes to how estimation output can readily be converted into LaTeX code. Finally, Section 8 concludes.

2 An overview, alternatives and development principles

2.1 GETS modeling

It is convenient to provide an overview of GETS modeling in terms of the linear regression model

$$y_t = \beta_1 x_{1t} + \cdots + \beta_k x_{kt} + u_t, \qquad t = 1, 2, \ldots, n, \tag{1}$$

where y_t is the dependent variable, the β's are slope coefficients, the x's are the regressors and u_t is a zero mean error term. GETS modeling assumes there exists at least one "local" data generating process (LDGP) nested in (1). By philosophical assumption *the* DGP is not contained in the simple model above, see Sucarrat (2010) and Hendry and Doornik (2014, Sections 6.2–6.3). The qualifier "local" thus means it is assumed that there exists a specification within (1) that is a statistically valid representation of *the* DGP. Henceforth, for notational and theoretical convenience, we will assume there exists only a single LDGP, but this is not a necessary condition.

A variable x_{jt}, $j \in \{1, \ldots, k\}$, is said to be relevant if $\beta_j \neq 0$ and irrelevant if $\beta_j = 0$. Let $k_{\text{rel}} \geq 0$ and $k_{\text{irr}} \geq 0$ denote the number of relevant and irrelevant variables, respectively, such that $k_{\text{rel}} + k_{\text{irr}} = k$. Of course, both k_{rel} and k_{irr} are unknown to the investigator. GETS modeling aims at finding a specification that contains as many relevant variables as possible, and a proportion of irrelevant variables that corresponds to the significance level α chosen by the investigator. Put differently, if $\widehat{k}_{\text{rel}}$ and $\widehat{k}_{\text{irr}}$ are the retained number of relevant and irrelevant variables, respectively, then GETS modeling aims at satisfying

$$\mathsf{E}(\widehat{k}_{\text{rel}}/k_{\text{rel}}) \to 1 \quad \text{and} \quad \mathsf{E}(\widehat{k}_{\text{irr}}/k_{\text{irr}}) \to \alpha \quad \text{as} \quad n \to \infty, \tag{2}$$

when $k_{\text{rel}}, k_{\text{irr}} > 0$. If either $k_{\text{rel}} = 0$ or $k_{\text{irr}} = 0$, then the criteria are modified in the obvious ways: If $k_{\text{rel}} = 0$, then $\mathsf{E}(\widehat{k}_{\text{rel}}) = 0$, and if $k_{\text{irr}} = 0$, then $\mathsf{E}(\widehat{k}_{\text{irr}}) = 0$. The proportion of spuriously retained variables, i.e., $\widehat{k}_{\text{irr}}/k_{\text{irr}}$, is also referred to as *gauge* in the GETS literature, with distributional results on the gauge for a specific case (the variables being impulses as in IIS)

4

Experiment	k_{rel}	k_{irr}	Algorithm	n	$m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$	$m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$	$\widehat{p}(\mathrm{DGP})$
HP1	0	40	**gets**	139		0.053	0.269
			AutoSEARCH			0.049	0.239
			HP1999			0.045	0.292
			PcGets			≈ 0.04	≈ 0.45
HP2'	1	39	**gets**	139	1.000	0.056	0.254
			AutoSEARCH		1.000	0.050	0.252
			HP1999		1.000	0.107	0.000
			PcGets		≈ 0.97	≈ 0.05	≈ 0.32
			Autometrics		1.000	0.063	0.119
HP7'	3	37	**gets**	138	0.999	0.055	0.232
			AutoSEARCH		1.000	0.051	0.232
			HP1999		0.967	0.082	0.040
			PcGets		≈ 1.00	≈ 0.04	≈ 0.37
			Autometrics		0.999	0.066	0.111

Table 1: Variable selection properties of GETS algorithms. The table is essentially Table 2 in Sucarrat and Escribano (2012, p. 724) augmented by the properties of **gets**, see Appendix A for more details on the simulations. The variable selection is undertaken with a nominal regressor significance level of 5%. $m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$, average proportion of relevant variables $\widehat{k}_{\mathrm{rel}}$ retained relative to the actual number of relevant variables k_{rel}. $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$, average proportion of irrelevant variables $\widehat{k}_{\mathrm{irr}}$ retained relative to the actual number of irrelevant variables k_{irr} in the GUM. $\widehat{p}(\mathrm{DGP})$, proportion of times the exact DGP is found. The properties of the HP1999 algorithm are from Hoover and Perez (1999, Table 4 on p. 179). The properties of the **PcGets** algorithm are from Hendry and Krolzig (2005, Figure 1 on p. C39), and the properties of the **Autometrics** algorithm are from Doornik (2009, Section 6).

provided in Johansen and Nielsen (2016). The relevance proportion, i.e., $\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}}$, is also referred to as *potency* in the GETS literature. Table 1 contains a comparison of the variable selection properties of GETS software packages for some well-known experiments. As the results show, **gets** performs as expected in the experiments, since the irrelevance proportion corresponds well to the nominal regressor significance level α, and since the relevance proportion is 1. Additional simulations, and comparisons against alternative algorithms, are contained in Section 2.2.

GETS modeling combines well-known ingredients from the model-selection literature: backwards elimination, tests on the β_j's (both single and multiple hypothesis tests), diagnostics tests, and fit-measures (e.g., information criteria). Specifically, GETS modeling may be described as proceeding in three steps:

1. Formulate a general unrestricted model (GUM) that passes a set of chosen diagnostic tests.[6] Each non-significant regressor in the GUM constitutes the starting point of a

---

[6]Currently, the standard diagnostic tests available in **gets** are tests for serial correlation and ARCH in the standardized residuals, and a test for non-normality. In addition, the user may add her or his own test or set of tests via the `user.diagnostics` argument.

backwards elimination path, and a regressor is non-significant if the p value of a two-sided t-test is lower than the chosen significance level α.

2. Undertake backwards elimination along multiple paths by removing, one-by-one, non-significant regressors as determined by the chosen significance level α. Each removal is checked for validity against the chosen set of diagnostic tests, and for parsimonious encompassing (i.e., a multiple hypothesis test) against the GUM.

3. Select, among the terminal models, the specification with the best fit according to a fit-criterion, e.g., the Schwarz (1978) information criterion.

For k candidate variables, there are 2^k possible models. As k becomes large the number of models becomes computationally infeasible, thus, a structured search is required. GETS provides such a structured search by starting with a general model (the GUM), and subsequently removing variables along search paths while checking the diagnostics at each removal.

2.2   A comparison of GETS and gets with alternatives

When comparing the R package **gets** to alternatives, it is important to differentiate the methodological approach of GETS modeling relative to other modeling approaches, from different software implementations within the GETS methodology. Here, we denote the broader field of GETS modeling by GETS, and the R package by **gets**. First we briefly review and compare alternative approaches to GETS modeling, then we discuss alternative implementations of GETS.

2.2.1   GETS compared to alternative methods – a feature-based comparison

Numerous model and variable selection methods have been proposed, and an even larger number of implementations are available. Focusing on variable selection, Table 2 contains a feature-based comparison of **gets** against some common alternatives in R. The `ar` function in **stats** (R Core Team, 2018) searches for the best $\text{AR}(P)$ model using the AIC. The `step` function, also in **stats**, offers both forward and backward step-wise search. The packages **lars** (Hastie and Efron, 2013) and **glmnet** (Friedman *et al.*, 2010), provide shrinkage-based search methods for variable selection.

As is clear from the table, GETS may be viewed as being more general than many of its competitors. This comes at a cost: computational speed. Relying on multiple path searches implies that the required computational time increases non-linearly with the number of potential candidate regressors selected over. This is a particular concern when using indicator saturation (Section 6), where the number of candidate variables scales linearly with the number of observations and subsequently implies a non-linear increase in required computational time. For example, selection over k (irrelevant) candidate regressors in **gets** (in a sample of $n = 200$ observations) on a 1.8GHz processor requires approximately 0.8 seconds (s) for $k = 10$, 2.9s for $k = 20$, 15s for $k = 40$, and 114s for $k = 80$. By contrast, the identical experiment with $k = 80$ requires 0.16s using the Lasso in **glmnet**, 0.41s in **lars**, and 0.3s using `step` (backward).

	ar	step (forward)	step (backward)	**lars**	**glmnet**	**gets**
AR-terms	Yes	Yes	Yes	Yes	Yes	Yes
Covariates ("X")		Yes	Yes	Yes	Yes	Yes
More variables than observations		Yes		Yes	Yes	Yes
Variance-modeling						Yes
Regressor tests during search						Yes
Diagnostics tests during search						Yes
Computational cost (relative)	Low	Low	Low	Low	Low	High

Table 2: A variable-selection focused feature-based comparison of **gets** against the `ar` and `step` functions in the R package **stats** (R Core Team, 2018), and against the R packages **lars** (Hastie and Efron, 2013) and **glmnet** (Friedman *et al.*, 2010).

2.2.2 GETS compared to alternative methods – a performance-based comparison

Hendry and Doornik (2014, Section 17) together with Castle *et al.* (2011) provide a broad overview of the performance of GETS relative to alternative model selection strategies of the mean of a regression, including step-wise regression, information criteria and penalized shrinkage-based selection using the Lasso (see Tibshirani, 1996). Castle *et al.* (2015) compare GETS in the context of step-shifts against the Lasso using LARS (Efron *et al.*, 2004), and Pretis *et al.* (2016) compare GETS against the Lasso for designed break functions (see Section 6.3 for a more detailed discussion of **gets** in the context of break detection). In both instances shrinkage-based selection is implemented using the R packages **lars** (Hastie and Efron, 2013) and **glmnet** (Friedman *et al.*, 2010). The emerging consensus from these simulation comparisons is that the false-positive rate, or irrelevance proportion or gauge, is erratic and difficult to control in step-wise as well as shrinkage-based selection procedures. When selecting on information criteria only, the implicit significance level of selection results in a high gauge when the number of candidate variables increases relative to the sample size. In contrast, the gauge tends to be well-calibrated around the nominal size of selection α in GETS. While the retention of relevant variables often is high in shrinkage-based approaches (and erratic in step-wise regression), this result comes at the cost of a high gauge and the performance becomes less reliable in the presence of correlation between the candidate variables.

To provide additional comparisons of performance to alternative methods for detecting relevant and discarding irrelevant variables, here we compare **gets** to: shrinkage-based selection, 1-cut selection (where all variables with p values $\leq \alpha$ in the GUM are retained in a single decision), and conducting selection inference starting at the DGP itself. The results are provided in Figure 1 (and Tables 5, 6, and 7 in Appendix B). The simulations cover three correlation structures of regressors: First, in-expectation uncorrelated regressors, second, positively correlated regressors ($\rho = 0.5$), and third, alternating negatively correlated regressors (where $\rho(x_i, x_{i+1}) = 0.5$, $\rho(x_i, x_{i+2}) = -0.5$). We consider a total of $k = 20$ regressors in a sample
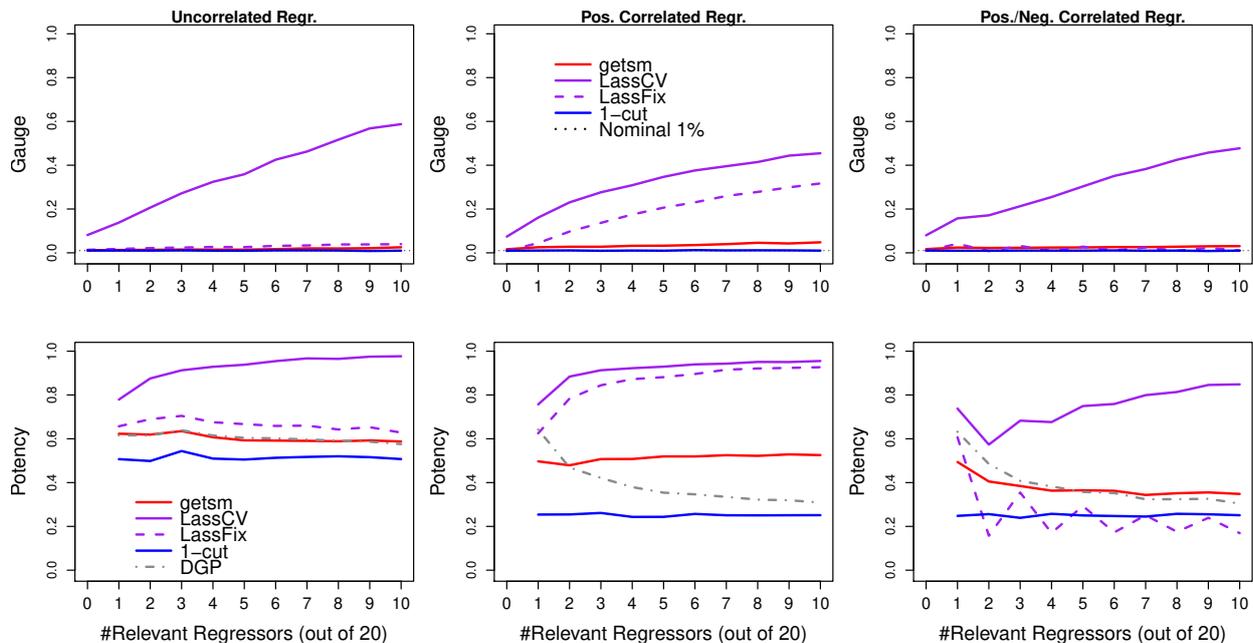
Figure 1: Performance of `getsm` selection algorithm compared against alternatives: cross-validated Lasso (LassCV), Lasso with fixed penalty (LassFix), 1-cut selection, and significance in the DGP itself (DGP). The top row shows the false retention rate (gauge), the bottom row shows the correct retention of relevant variables (potency). Columns show uncorrelated, positively correlated, and alternating positively and negatively correlated regressors. Nominal selection in `getsm` taken place at 1% significance level.

of $n = 500$ observations for 1000 replications. The number of relevant regressors is increased from $k_{rel} = 0$ to $k_{rel} = 10$ with coefficients set to correspond to an expected t-statistic of ≈ 3. The performance of **gets** using the `getsm` function is compared to the cross-validated Lasso in **glmnet** and the Lasso with fixed penalty parameter such that the false-detection rate approximately matches `getsm` under the null (when $k_{rel} = 0$). The significance level of 1-cut selection is chosen to match $\alpha = 1\%$ in `getsm` selection.

The simulation results presented here match the evidence from previous studies: GETS selection yields a false-detection rate close to the nominal size of selection regardless of the correlation structure of regressors considered. While exhibiting high potency, the false detection rate of Lasso is difficult to control when the correlation structure varies and the number of relevant variables is unknown. GETS dominates 1-cut selection when regressors are correlated, and closely matches 1-cut in absence of correlation.

To the best of our knowledge, the only currently publicly available software that provides automated model selection of the variance is **gets**. The reason for this is that **gets** sidesteps the numerical estimation difficulties usually associated with models of the variance thanks to its OLS estimation procedure, see the discussion in Sucarrat and Escribano (2012).

	HP1999	**PcGets**	**Autometrics**	**gets**
More than 10 paths		Yes	Yes	Yes
GETS of mean	Yes	Yes	Yes	Yes
GETS of variance				Yes
Impulse and step IS			Yes	Yes
Trend IS				Yes
IS variance correction				Yes
User-defined diagnostics				Yes
GETS of logit models			Yes	
GETS of systems			Yes	
Menu-based GUI		Yes	Yes	
Free and open source	Yes			Yes

Table 3: A feature-based comparison of GETS software packages; the MATLAB code of Hoover and Perez (1999) (HP1999), **PcGets** version 0.9, **Autometrics** version 7 and **gets** version 0.12.

2.2.3 Alternatives within the field of GETS

There have been different software implementations of GETS modeling – Table 3 summarizes the similarities and differences between these. The main (currently available) alternative to the package **gets** for GETS modeling of the mean in regression models is **Autometrics** (Doornik, 2009) written in Ox (Doornik, 2006) within the software package **PcGive** (Doornik and Hendry, 2007b). **Autometrics** and **gets** share common features in GETS modeling of the mean in regression models, and in the general implementation of impulse- and step-indicator saturation. There are, however, notable differences between the two implementations: The main advantages of **gets** lie in being the only GETS implementation of variance models, the implementation of new and unique features in indicator saturation methods including trend-indicator saturation (TIS), consistency and efficiency corrections of the variance estimates, and testing of the time-varying mean (see Section 6.3 for an in-depth discussion of the differences in indicator saturation between **Autometrics** and **gets**), as well as new features in model selection (e.g., the availability of a direct function to correct for model-selection bias). In turn, selection over systems of equations can be conducted automatically in **Autometrics** while having to be done by one-equation at a time in **gets**.

2.3 Development principles of the package gets

The original motivation behind the precursor of **gets** (i.e., **AutoSEARCH**) was to make GETS modeling methods of the variance (and mean) of a regression freely and publicly available, while being open-source and implementing recent developments in GETS. This principle will continue to guide the development of **gets**. Indicator saturation methods were added to **gets** in version 0.2, and we plan to expand **gets** further to include model classes for which there currently is no GETS software, e.g., spatial models, panel-data, etc. Naturally, we encourage others keen to

develop and publish GETS modeling methods for a wider range of alternatives, either within the **gets** package or as a separate package. Another important development principle is that we would like to enable more user-specified control. User-specified diagnostics, for example, were added in version 0.10, and we also plan to enable user-specified estimation and inference procedures (this is already available in `arx`, but not in `getsm`, `getsv` and `isat`). Finally, we also aim at making the package computationally faster and more user-friendly.

3 Setting time-series attributes

The **gets** package is not limited to time series models and does not require that time-series characteristics are set beforehand (for example if the data at hand are not time series). However, if time series characteristics are not set, and if the data are in fact time series, then graphs and other outputs (e.g., fitted values, residuals, etc.) are not optimal. The **gets** package is optimized to work with Z's ordered observations (ZOO) package **zoo**, see Zeileis and Grothendieck (2005). In fact, the fitted values, residuals, recursive estimates and so on returned by **gets** functions, are all objects of class 'zoo'. The **zoo** package provides a very general and versatile infrastructure for observations that are ordered according to an arbitrary index, e.g., time-series, and **zoo** is adapted to interact well with the less versatile time-series class of the **base** distribution, 'ts': To convert 'ts' objects to 'zoo' objects, simply use `as.zooreg` (preferred) or `as.zoo`. See the help system and webpage of the **zoo** package for several short intros and vignettes: https://CRAN.R-project.org/package=zoo.

4 The AR-X model with log-ARCH-X errors

The specifications considered by **gets** are all contained in the AR-X model with log-ARCH-X errors. This model is made up of two equations, one for the mean and one for the log-variance:

$$y_t = \phi_0 + \sum_{r=1}^{R} \phi_r y_{t-r} + \sum_{s=1}^{S} \eta_s x_{s,t}^m + \epsilon_t, \qquad \epsilon_t = \sigma_t z_t, \quad z_t \sim \text{iid}(0,1), \tag{3}$$

$$\ln \sigma_t^2 = \alpha_0 + \sum_{p=1}^{P} \alpha_p \ln \epsilon_{t-p}^2 + \sum_{q \in Q} \beta_q \ln \text{EqWMA}_{q,t-1}$$

$$+ \sum_{a=1}^{A} \lambda_a (\ln \epsilon_{t-a}^2) I_{\{\epsilon_{t-a}<0\}} + \sum_{d=1}^{D} \delta_d x_{d,t}^v. \tag{4}$$

The conditional mean equation (3) is an autoregressive (AR) specification of order R with S covariates $x_{1,t}^m, \ldots, x_{S,t}^m$ ("X"), AR-X for short. The covariates may contain lags of conditioning variables. The error term ϵ_t is a product of the time-varying conditional standard deviation $\sigma_t > 0$ and the real-valued innovation z_t, where z_t is iid with zero mean and unit variance conditional on the past. The conditional log-variance equation (4) is given by a logarithmic autoregressive conditional heteroscedasticity (log-ARCH) specification of order P with volatility proxies defined as $\text{EqWMA}_{q,t-1} = (\epsilon_{t-1}^2 + \cdots + \epsilon_{t-q}^2)/q$, A logarithmic asymmetry terms (i.e., "leverage") analogous to those of Glosten *et al.* (1993) – so $I_{\epsilon_{t-a}<0}$ is an indicator function equal

to 1 if $\epsilon_{t-a} < 0$ and 0 otherwise, and D covariates $x^v_{1,t}, \ldots, x^v_{D,t}$, log-ARCH-X for short. The covariates may contain lags of conditioning variables, and the covariates in the mean need not be the same as those of the log-variance specification. Hence the superscripts m and v, respectively. The log-proxies $\ln \text{EqWMA}_{q,t-1}$, where EqWMA is short for equally weighted moving average, are intended to proxy lagged log-GARCH terms, e.g., $\ln \sigma^2_{t-1}$. However, it should be noted that the log-proxies can also be given additional interpretation of interest. For example, if $y_t = \epsilon_t$ is a daily financial return, and if the returns are recorded over weekdays only, then $\text{EqWMA}_{5,t-1}$, $\text{EqWMA}_{20,t-1}$ and $\text{EqWMA}_{60,t-1}$ can be interpreted as the "weekly", "monthly" and "quarterly" volatilities, respectively. The log-proxies thus provide great flexibility in modeling the persistence of log-volatility. Also, note that $\text{EqWMA}_{q,t-1} = \ln \epsilon^2_{t-1}$, i.e., the ARCH(1) term, when $q = 1$. Of course, additional volatility proxies can be included via the covariates $x_{d,t}$.

The model (3)–(4) is estimated in two steps.[7] First, the mean specification (3) is estimated by OLS. The default variance-covariance matrix is the ordinary one, but – optionally – this can be changed to either that of White (1980) or that of Newey and West (1987). Second, the nonlinear AR-representation of (4) is estimated, also by OLS. The nonlinear AR-representation is given by

$$\ln \epsilon^2_t = \alpha^*_0 + \sum_{p=1}^P \alpha_p \ln \epsilon^2_{t-p} + \sum_{q \in Q} \beta_q \ln \text{EqWMA}_{q,t-1} \tag{5}$$

$$+ \sum_{a=1}^A \lambda_a (\ln \epsilon^2_{t-a}) I_{\{\epsilon_{t-a} < 0\}} + \sum_{d=1}^D \delta_d x^v_{d,t} + u_t, \tag{6}$$

where $\alpha^*_0 = \alpha_0 + \mathsf{E}(\ln z^2_t)$ and $u_t = \ln z^2_t - \mathsf{E}(\ln z^2_t)$ with $u_t \sim \text{iid}(0, \sigma^2_u)$. This provides consistent estimates of all the parameters in (4) except α_0, under appropriate assumptions. To identify α_0, an estimate of $\mathsf{E}(\ln z^2_t)$ is needed, which depends on the density of z_t. Sucarrat et al. (2016) show that a simple formula made up of the residuals $\widehat{u}_t$ provides a consistent and asymptotically normal estimate under very general and non-restrictive assumptions. The estimator is essentially the negative of the natural log of the smearing estimate of Duan (1983): $\widehat{\mathsf{E}}(\ln z^2_t) = -\ln\left[n^{-1} \sum_{t=1}^n \exp(\widehat{u}_t)\right]$. So the expression in square brackets is the smearing estimate. The log-variance intercept α_0 can thus be estimated by $\widehat{\alpha}^*_0 - \widehat{\mathsf{E}}(\ln z^2_t)$. Finally, the ordinary variance-covariance matrix is used for inference in the log-variance specification, since the error term u_t of the nonlinear AR-representation is iid.

4.1   Simulation

Simulation from an AR(P) process can readily be done with the `arima.sim` function in the **stats** package (part of the base distribution of R). For example, the following code simulates 100 observations from the AR(1) model $y_t = \phi_0 + \phi_1 y_{t-1} + \epsilon_t$ with $\phi_0 = 0$ and $\phi_1 = 0.4$:

---

[7] A multi-step, iterative procedure might improve the finite sample efficiency, but does not necessarily improve the asymptotic efficiency. Joint estimation of the two equations in a single step, e.g., by Gaussian maximum likelihood, is likely to be asymptotically more efficient when z_t is not too fat-tailed, see Francq and Sucarrat (2018). In finite samples, however, it is likely to be less efficient when many parameters are estimated simultaneously due to numerical issues.

```
set.seed(123)
y <- arima.sim(list(ar = 0.4), 100)
```

To simulate from a model with log-ARCH errors, we first need to simulate the errors. This can be achieved with `lgarchSim` from the **lgarch** package (Sucarrat, 2015b):

```
library("lgarch")
```

Next, the following code simulates an error-term ϵ_t that follows the log-ARCH(1) specification $\ln \sigma_t^2 = \alpha_0 + \alpha_1 \ln \epsilon_{t-1}^2$ with $\alpha_0 = 0$ and $\alpha_1 = 0.3$:

```
eps <- lgarchSim(100, arch = 0.3, garch = 0)
```

By default, the standardized error z_t is normal, but this can be changed via the `innovation` argument of the `lgarchSim` function. To combine the log-ARCH error with an AR(1) model with $\phi_0 = 0$ and $\phi_1 = 0.4$ the following code can be used:

```
yy <- arima.sim(list(ar = 0.4), 100, innov = eps)
```

The command `plot(as.zoo(cbind(y, yy, eps)))` plots the three series.

4.2 arx(): Estimation

The function `arx` estimates an AR-X model with log-ARCH-X errors. For example, the following code loads the **gets** package, fits an AR(1) model with intercept to the series `y` generated in Section 4.1, and stores the results in an object called `mod01`:

```
library("gets")
mod01 <- arx(y, ar = 1)
```

To print the estimation results, simply type `mod01`. This returns:

```
Date: Fri Aug 06 10:57:59 2021
Dependent var.: y
Method: Ordinary Least Squares (OLS)
Variance-Covariance: Ordinary
No. of observations (mean eq.): 99
Sample: 2 to 100

Mean equation:

          coef std.error t-stat   p-value
mconst 0.034045  0.091664 0.3714    0.7111
ar1    0.397411  0.095212 4.1740 6.533e-05

Diagnostics and fit:

                Chi-sq df p-value
Ljung-Box AR(2)   0.25922  2  0.8784
```

```
Ljung-Box ARCH(1) 0.26124  1  0.6093

SE of regression    0.90933
R-squared           0.15226
Log-lik.(n=99)   -130.06490
```

The two diagnostic tests are of the standardized residuals $\widehat{z}_t$. The AR and ARCH tests are Ljung and Box (1978) tests for serial correlation in $\widehat{z}_t$ and $\widehat{z}_t^2$, respectively, and the number in parentheses indicates at which lag the test is conducted. R-squared is that of the mean specification, whereas the (Gaussian) log-likelihood is made up of the residuals $\widehat{\epsilon}_t$. If no log-variance specification is fitted, then the conditional variance in the log-likelihood is constant and equal to the sample variance of the residuals. By contrast, if a log-variance specification is fitted, then the conditional variance in the log-likelihood is equal to the fitted conditional variance, which is given by $\widehat{\sigma}_t^2 = \exp(\ln \widehat{\sigma}_t^2)$.

The main optional arguments of the arx function when estimating the mean are:

- mc: TRUE (default) or FALSE. mc is short for "mean constant", so mc = TRUE includes an intercept, whereas FALSE does not.

- ar: integer vector that indicates the AR terms to include, say, ar = 1, ar = 1:4 or ar = c(2, 4).

- mxreg: vector, matrix or 'zoo' object that contains additional regressors to be included in the mean specification.

- vcov.type: the type of variance-covariance matrix used for inference in the mean specification. By default, the ordinary ("ordinary") matrix is used. The other options available are "white", i.e., the heteroscedasticity robust variance-covariance matrix of White (1980), and "newey-west", i.e., the heteroscedasticity and autocorrelation robust variance-covariance matrix of Newey and West (1987).

To make full use of these arguments, let us first generate a set of 5 regressors:

```
mX <- matrix(rnorm(100 * 5), 100, 5)
```

Next, the following code estimates an AR-X model with an intercept, two AR-lags and five regressors, and stores the estimation results in an object called mod02:

```
mod02 <- arx(y, ar = 1:2, mxreg = mX, vcov.type = "white")
```

Estimation of the log-variance specification is also undertaken with the arx function. For example, the following code fits the log-ARCH(1) specification $\ln \sigma_t^2 = \alpha_0 + \alpha_1 \ln \epsilon_{t-1}^2$ to the variable eps generated above:

```
mod03 <- arx(eps, mc = FALSE, arch = 1)
```

Typing mod03 prints the estimation results. The main optional arguments when estimating the log-variance are:

- **arch**: integer vector that indicates the log-ARCH terms to include, say, `arch = 1`, `arch = 1:3` or `arch = c(3, 5)`.

- **asym**: integer vector that indicates the logarithmic asymmetry terms (often referred to as "leverage") to include, say, `asym = 1`, `asym = 1:4`, or `asym = c(2, 4)`.

- **vxreg**: vector, matrix or 'zoo' object that contains additional regressors to be included in the log-volatility specification.

The following code provides an example that makes use of all three arguments:

```
mod04 <- arx(eps, mc = FALSE, arch = 1:3, asym = 2, vxreg = log(mX^2))
```

Again, typing `mod04` prints the results. Finally we give an example where we jointly fit a mean and log-variance equation to the series `yy` generated above, using the variance-covariance matrix of White (1980) for the mean equation:

```
mod05 <- arx(yy, ar = 1:2, mxreg = mX, arch = 1:3, asym = 2,
  vxreg = log(mX^2), vcov.type = "white")
```

4.3 Extraction functions

There are a number of functions available for extracting information from 'arx' objects. The most important of these (most of them S3 methods) are:

```
coef, ES, fitted, logLik, plot, predict, print, recursive, residuals,
rsquared, sigma, summary, toLatex, VaR, vcov
```

Six of these (`coef`, `fitted`, `predict`, `recursive`, `residuals` and `vcov`) have an optional argument that allows you to choose whether to extract information pertaining to the mean or log-variance specification. The `print` function prints the estimation result, `logLik` extracts the (Gaussian) log-likelihood associated with the joint model, `summary` lists the entries of the 'arx' object (a `list`), `plot` plots the fitted values and residuals of the model, `recursive` computes and – optionally – plots the recursive coefficient estimates, `rsquared` and `sigma` extract the R-squared and standard error of regression, respectively, while `ES` and `VaR` extract the conditional expected shortfall and value-at-risk, respectively.

4.4 Example: A model of quarterly inflation with time-varying conditional variance

When Engle (1982) proposed the ARCH-class of models, his empirical application was the uncertainty of UK-inflation. However, the ARCH(4) specification he used to model the conditional variance was severely restricted in order to ensure the positivity of the variance estimates, see Engle (1982, p. 1002). Arguably, this is why (non-exponential) ARCH specifications never became popular in macroeconomics. The log-ARCH class of models, by contrast, does not suffer from the positivity problem, since the conditional variance is specified in logs. To illustrate we fit an AR(4)-X-log-ARCH(4)-X model to a quarterly inflation series, and show that

the conditional variance specification provides a substantial improvement in terms of fit and diagnostics.

The following code imports the data[8] and assigns it quarterly time-series attributes:

```
data("infldata", package = "gets")
infldata <- zooreg(infldata[, -1], frequency = 4, start = c(1989, 1))
```

Note that `[, -1]` removes the first column, since it is not needed. The dataset thus contains four variables: `infl`, `q2dum`, `q3dum` and `q4dum`. The first variable is quarterly Norwegian inflation (year-on-year) in % from 1989(1) to 2015(4), whereas the latter three are seasonal dummies associated with the second, third and fourth quarter, respectively. Initially, to illustrate why a time-varying conditional variance is needed, we estimate only the mean specification:

$$\texttt{infl}_t = \phi_0 + \sum_{r=1}^{4} \phi_r \texttt{infl}_{t-r} + \eta_2 \texttt{q2dum}_t + \eta_3 \texttt{q3dum}_t + \eta_4 \texttt{q4dum}_t + \epsilon_t. \tag{7}$$

That is, an AR(4)-X, where the dummies constitute the X-part. The code

```
inflMod01 <- arx(inflData[, "infl"], ar = 1:4, mxreg = inflData[, 2:4],
  vcov.type = "white")
```

estimates the model using heteroscedasticity-robust coefficient standard errors of the White (1980) type, and typing `inflMod01` prints the estimation results:

```
Date: Fri Aug 06 11:11:17 2021
Dependent var.: y
Method: Ordinary Least Squares (OLS)
Variance-Covariance: White (1980)
No. of observations (mean eq.): 104
Sample: 1990(1) to 2015(4)

Mean equation:

              coef  std.error  t-stat   p-value
mconst  0.8386311  0.2961338   2.8319  0.005637
ar1     0.7257550  0.1300407   5.5810 2.211e-07
ar2     0.0195911  0.1171347   0.1673  0.867523
ar3     0.0350092  0.1385735   0.2526  0.801087
ar4    -0.1676751  0.1336972  -1.2541  0.212836
q2dum  -0.0148892  0.2333917  -0.0638  0.949266
q3dum  -0.0072972  0.2262704  -0.0322  0.974340
q4dum   0.0103990  0.2226772   0.0467  0.962849

Diagnostics and fit:
```

---

[8]The source of the data is Statistics Norway (http://www.ssb.no/). The original untransformed data, a monthly consumer price index (CPI), was retrieved 14 February 2016 via http://www.ssb.no/tabell/08183/.

```
                 Chi-sq df  p-value
Ljung-Box AR(5)    16.3205  5 0.005986
Ljung-Box ARCH(1)   5.9665  1 0.014580


SE of regression     0.72814
R-squared            0.53166
Log-lik.(n=104)   -110.57435
```

The diagnostics suggest the standardized residuals are autocorrelated and heteroscedastic, since the tests for autocorrelation and heteroscedasticity yield p values of 0.6% and 1.5%, respectively. Next, we specify the conditional variance as a log-ARCH(4)-X, where the X-part is made up of the seasonal dummies:

$$\ln \sigma_t^2 = \alpha_0 + \sum_{p=1}^{4} \alpha_p \ln \epsilon_{t-p}^2 + \delta_2 \texttt{q2dum}_t + \delta_3 \texttt{q3dum}_t + \delta_4 \texttt{q4dum}_t. \tag{8}$$

The code

```
inflMod02 <- arx(inflData[, "infl"], ar = 1:4, mxreg = inflData[, 2:4],
   arch = 1:4, vxreg = inflData[, 2:4], vcov.type = "white")
```

estimates the full model with White (1980) standard errors in the mean and ordinary standard errors in the log-variance. Typing `inflMod02` returns

```
Date: Fri Aug 06 11:12:20 2021
Dependent var.: y
Method: Ordinary Least Squares (OLS)
Variance-Covariance: White (1980)
No. of observations (mean eq.): 104
Sample: 1990(1) to 2015(4)


Mean equation:


              coef  std.error  t-stat    p-value
mconst  0.8386311  0.2961338   2.8319   0.005637
ar1     0.7257550  0.1300407   5.5810 2.211e-07
ar2     0.0195911  0.1171347   0.1673   0.867523
ar3     0.0350092  0.1385735   0.2526   0.801087
ar4    -0.1676751  0.1336972  -1.2541   0.212836
q2dum  -0.0148892  0.2333917  -0.0638   0.949266
q3dum  -0.0072972  0.2262704  -0.0322   0.974340
q4dum   0.0103990  0.2226772   0.0467   0.962849


Log-variance equation:


          coef std.error   t-stat   p-value
```

```
vconst   0.95935    0.53464   3.2199 0.072749
arch1    0.16697    0.10352   1.6130 0.110169
arch2    0.12027    0.10335   1.1637 0.247566
arch3    0.14740    0.10332   1.4267 0.157060
arch4    0.05982    0.10515   0.5689 0.570824
q2dum   -1.32860    0.61862  -2.1477 0.034366
q3dum   -0.92707    0.58400  -1.5874 0.115843
q4dum   -1.82736    0.62014  -2.9467 0.004069


Diagnostics and fit:


                   Chi-sq df p-value
Ljung-Box AR(5)    9.1776  5  0.1022
Ljung-Box ARCH(5)  1.7613  5  0.8811


 SE of regression   0.72814
 R-squared          0.53166
 Log-lik.(n=100)   -82.32892
```

The first noticeable difference between `inflMod01` and `inflMod02` is that the diagnostics improve substantially. In `inflMod02`, the AR and ARCH tests of the standardized residuals suggest the standardized error z_t is uncorrelated and homoscedastic at the usual significance levels (1%, 5% and 10%), and the Jarque and Bera (1980) test suggests z_t is normal. The second noticeable improvement is in terms of fit, as measured by the average (Gaussian) log-likelihood. In `inflMod01` the average log-likelihood is $-110.57435/104 = -1.06$, whereas in `inflMod02` the average log-likelihood is $-82.3289/100 = -0.82$. This is a substantial increase. In terms of the Schwarz (1978) information criterion (SC), which favors parsimony, a comparison of the average log-likelihoods can be made by the `info.criterion` function:

```
info.criterion(as.numeric(logLik(inflMod01)), n = 104, k = 8 + 1)
info.criterion(as.numeric(logLik(inflMod02)), n = 100, k = 8 + 8)
```

As is clear, the value falls from 2.53 in `inflMod01` to 2.38 in `inflMod02`. (A comparison of the average log-likelihoods is necessary, since the two models are estimated with a different number of observations. This is the main difference between the `info.criterion` function and `AIC` and `BIC`.) Together, the enhanced fit and diagnostics indicate the log-variance specification provides a notable improvement. Later, in Section 5.4, we will undertake GETS modeling of the mean and variance specifications of `inflMod02`.

4.5   Example: A log-ARCH-X model of daily SP500 volatility

The most common volatility specification in finance are first order GARCH-like specifications. In the log-GARCH class of models, this corresponds to a log-GARCH(1, 1): $\ln \sigma_t^2 = \alpha_0 + \alpha_1 \ln \epsilon_{t-1}^2 + \beta_1 \ln \sigma_{t-1}^2$. Here, we show that a log-ARCH-X model that makes use of commonly available information provides a better fit.

We start by loading a dataset of the Standard and Poor's 500 (SP500) index:

```
data("sp500data", package = "gets")
sp500data <- zoo(sp500data[, -1],
  order.by = as.Date(sp500data[, "Date"]))
```

The dataset contains the daily value of the SP500 index, its highs and lows, and daily volume. We will make use of this information together with day-of-the-week dummies to construct a rich model of SP500 return volatility. But first we shorten the sample, since not all variables are available from the start:

```
sp500data <- window(sp500data, start = as.Date("1983-07-01"))
```

The resulting sample thus goes from 1 July 1983 to 8 March 2016, a total of 8241 observations before differencing and lagging. Next, the following lines of code create a variable equal to the log-return in percent, a lagged range-based volatility proxy, and the lagged log-difference of volume:

```
sp500Ret <- diff(log(sp500data[, "Adj.Close"])) * 100
relrange <- (log(sp500data[, "High"]) - log(sp500data[, "Low"]) ) * 100
volproxy <- log(relrange^2)
volproxylag <- lag(volproxy, k = -1)
volume <- log(sp500data[, "Volume"])
volumediff <- diff(volume) * 100
volumedifflag <- lag(volumediff, k = -1)
```

Finally, we make the day-of-the-week dummies and estimate the full model, a log-ARCH(5)-X specification:

```
sp500Index <- index(sp500Ret)
days <- weekdays(sp500Index)
days <- union(days, days)
dTue <- zoo(as.numeric(weekdays(sp500Index) == days[1]),
  order.by = sp500Index)
dWed <- zoo(as.numeric(weekdays(sp500Index) == days[2]),
  order.by = sp500Index)
dThu <- zoo(as.numeric(weekdays(sp500Index) == days[3]),
  order.by = sp500Index)
dFri <- zoo(as.numeric(weekdays(sp500Index) == days[4]),
  order.by = sp500Index)
sp500Mod01 <- arx(sp500Ret, mc = FALSE, arch = 1:5, log.ewma = c(5, 20, 60, 120),
  asym = 1,
  vxreg = cbind(volproxylag, volumedifflag, dTue, dWed, dThu, dFri))
```

Typing sp500Mod01 returns the following print output:

```
Date: Fri Aug 06 11:17:38 2021
Dependent var.: y
Method: Ordinary Least Squares (OLS)
Sample: 1983-07-05 to 2016-03-08
```

```
Log-variance equation:

                     coef   std.error   t-stat     p-value
vconst          0.0107260  0.0784437   0.0187    0.891241
arch1          -0.0482520  0.0161972  -2.9790    0.002900
arch2           0.0071996  0.0122312   0.5886    0.556127
arch3           0.0256668  0.0122521   2.0949    0.036212
arch4           0.0149581  0.0122145   1.2246    0.220758
arch5           0.0371055  0.0122796   3.0217    0.002521
asym1          -0.0336271  0.0175185  -1.9195    0.054954
logEqWMA(5)     0.0262491  0.0519435   0.5053    0.613334
logEqWMA(20)    0.2817220  0.0713466   3.9486  7.926e-05
logEqWMA(60)    0.1970841  0.1052311   1.8729    0.061122
logEqWMA(120)   0.1936954  0.0865864   2.2370    0.025312
volproxylag     0.2078785  0.0400515   5.1903  2.151e-07
volumedifflag  -0.0030906  0.0014207  -2.1754    0.029630
dTue            0.0978314  0.0834703   1.1720    0.241212
dWed           -0.0804053  0.0853471  -0.9421    0.346171
dThu            0.0838896  0.0843500   0.9945    0.319988
dFri            0.0756869  0.0840118   0.9009    0.367664


Diagnostics and fit:

                    Chi-sq df  p-value
Ljung-Box AR(1)    0.53421   1   0.4648
Ljung-Box ARCH(6) 29.21040   6 5.55e-05


SE of regression       1.13957
R-squared             -0.00069
Log-lik.(n=8120) -10985.79738
```

Later, in Section 5.5, we will simplify this model with the `getsv` function. For now, we provide a comparison with a log-GARCH(1, 1) using the R package **lgarch**, see Sucarrat (2015b). The following code loads the package, estimates the model and stores the estimation results:

```
library("lgarch")
sp500Mod02 <- lgarch(sp500Ret)
```

Extracting the log-likelihood by `logLik(sp500Mod02)` reveals that it is substantially lower, namely −11396.11. To compare the models in terms of the Schwarz (1978) information criterion, it is necessary to undertake the comparison in terms of the average log-likelihoods, since the estimation samples of the two models have a different number of observations:

```
info.criterion(as.numeric(logLik(sp500Mod01)), n = 8120, k = 17)
info.criterion(as.numeric(logLik(sp500Mod02)), n = 8240, k = 3)
```

The value increases from 2.7247 in `sp500Mod01` to 2.7693 in `sp500Mod02`, which indicates that the former specification provides a better fit.

5 GETS modeling

5.1 getsm(): Modeling the mean

GETS modeling of the mean specification in a regression (e.g., a simple time series or cross-sectional model) is undertaken by applying the `getsm` function on an 'arx' object. This conducts GETS variable selection on the regressors included in the initially specified `arx` model. For example, the following code performs GETS model selection on the regressors of the mean specification of `mod05` with default values on all the optional arguments:

```
getsm05 <- getsm(mod05)
```

The results are stored in an object named `getsm05`, and the information produced during the specification search is:

```
GUM mean equation:

        reg.no. keep       coef   std.error  t-stat  p-value
mconst        1     0 -0.0596894  0.0782285 -0.7630   0.4475
ar1           2     0  0.1938157  0.1235456  1.5688   0.1202
ar2           3     0  0.0343803  0.1141559  0.3012   0.7640
mxreg1        4     0  0.1171045  0.0805838  1.4532   0.1496
mxreg2        5     0  0.0116124  0.0865925  0.1341   0.8936
mxreg3        6     0 -0.1087162  0.0815946 -1.3324   0.1861
mxreg4        7     0 -0.2226722  0.1019820 -2.1834   0.0316
mxreg5        8     0  0.0012498  0.0694024  0.0180   0.9857


GUM log-variance equation:

           coef std.error  t-stat p-value
vconst  0.351872  0.438687  0.6434 0.42249
arch1   0.268975  0.107470  2.5028 0.01424
arch2   0.088540  0.159135  0.5564 0.57941
arch3   0.022932  0.115861  0.1979 0.84357
asym2  -0.112941  0.171767 -0.6575 0.51262
vxreg1  0.102181  0.110374  0.9258 0.35718
vxreg2 -0.068873  0.093762 -0.7345 0.46464
vxreg3 -0.032006  0.102597 -0.3120 0.75584
vxreg4  0.029429  0.106865  0.2754 0.78369
vxreg5  0.187176  0.120259  1.5564 0.12332


Diagnostics:
```

```
                   Chi-sq df p-value
Ljung-Box AR(3)    0.18672  3 0.97970
Ljung-Box ARCH(4) 0.43983  4 0.97909


7 path(s) to search
Searching: 1 2 3 4 5 6 7


Path 1: 1 8 5 3 4 6 2
Path 2: 2 8 5 3 1 4 6
Path 3: 3 8 5 1 4 6 2
Path 4: 4 3 5 8 1 6 2
Path 5: 5 8 3 1 4 6 2
Path 6: 6 8 5 3 1 4 2
Path 7: 8 5 3 1 4 6 2


Terminal models:


                  info(sc)      logl  n k
spec 1 (1-cut): 2.285792 -109.7113 98 1


Retained regressors (final model):

 mxreg4
```

To see the estimation results of the final model, type `getsm05`. The first part of the printed results pertains to the GUM, i.e. the starting model. Note in particular that regressors are numbered (the `reg.no` column in the GUM mean equation). This is useful when interpreting paths searched, which indicates in which order the regressors are deleted in each path. Next, the `Terminal models` part contains the distinct terminal specifications. By default, the Schwarz (1978) information criterion (sc) is used to choose among the terminals, but this can be changed (see below). The last part contains the estimation results of the final, simplified model.

The main optional arguments of the `getsm` function are (type `args(getsm)` or `?getsm` for all the arguments):

- `t.pval`: numeric value between 0 and 1 (The default is 0.05). The significance level used for the two-sided t-tests of the regressors.

- `wald.pval`: numeric value between 0 and 1 (the default is `t.pval`). The significance level used for the parsimonious encompassing test (PET) against the general unrestricted model (GUM) at each regressor deletion.

- `do.pet`: logical, `TRUE` (the default) or `FALSE`. If `TRUE`, then a PET against the GUM is undertaken at each regressor removal.

- `ar.LjungB`: a list with two elements named `lag` and `pval`, respectively, or `NULL`. If the list is not `NULL`, then a Ljung and Box (1978) test for serial correlation in the standardized residuals is undertaken at each attempt to remove a regressor. The default, `list(lag`

= NULL, pval = 0.025), means the lag is chosen automatically (as max(ar) + 1), and that a p value of pval = 0.025 is used. If the list is NULL, then the standardized residuals $\hat{z}_t$ are not checked for serial correlation after each removal.

- arch.LjungB: a list with two elements named lag and pval, respectively, or NULL. If the list is not NULL, then a Ljung and Box (1978) test for serial correlation in the squared standardized residuals is undertaken at each attempt to remove a regressor. The default, list(lag = NULL, pval = 0.025), means the lag is chosen automatically (as max(arch) + 1) and that a p value of pval = 0.025 is used. If the list is NULL, then the squared standardized residuals $\hat{z}_t^2$ are not checked for serial correlation after each removal.

- vcov.type: NULL, "ordinary", "white" or "newey-west". If NULL (default), then the type of variance-covariance matrix is automatically determined (the option from the 'arx' object is used). If "ordinary", then the ordinary variance-covariance matrix is used. If "white", then the variance-covariance matrix of White (1980) is used. If "newey-west", then the variance-covariance matrix of Newey and West (1987) is used.

- keep: either NULL or an integer vector. If NULL (default), then no regressors are excluded from removal. Otherwise, the regressors associated with the numbers in keep are excluded from the removal space. For example, keep = 1 excludes the intercept from removal. Retaining variables using the keep argument implements the "theory-embedding" approach outlined in Hendry and Johansen (2015) by "forcing" theory variables to be retained while conducting model discovery beyond the set of forced variables.

- info.method: "sc", "aic" or "hq". If "sc" (default), then the information criterion of Schwarz (1978) is used as tiebreaker between the terminals. If "aic", then the information criterion of Akaike (1974) is used, and if "hq", then the information criterion of Hannan and Quinn (1979) is used.

As an example, the following code uses a lower significance level for the regressor significance tests and the PETs, and turns of diagnostic testing for ARCH in the standardized residuals:

```
getsm05a <- getsm(mod05, t.pval = 0.01, arch.LjungB = NULL)
```

Similarly, the following code restricts the mean intercept from being deleted, even though it is not significant:

```
getsm05b <- getsm(mod05, keep = 1)
```

5.2 getsv(): Modeling the log-variance

GETS modeling of the log-variance specification is undertaken by applying the getsv function to an 'arx' object. For example, the following code performs GETS model selection of the log-variance specification of mod05 with default values on all the optional arguments:

```
getsv05 <- getsv(mod05)
```

Alternatively, the following code undertakes GETS model selection on the log-variance specification of the simplified model getsm05:

```
mod06 <- arx(residuals(getsm05), mc = FALSE, arch = 1:3, asym = 2,
  vxreg = log(mX^2))
getsv06 <- getsv(mod06)
```

Typing `getsv06` prints the results. Note that `vconst`, the log-variance intercept, is forced to enter the `keep` set when `getsv` is used. That is, α_0 is restricted from removal even if it is not significant. This is due to the estimation procedure, which is via the AR-representation. Finally, the main optional arguments of `getsv` are almost the same as those of `getsm` (see above). The main difference is that the only variance-covariance matrix available is the ordinary one, since the error-term of the AR-specification is iid. As an example of how to set some of the options to non-default values, the following code restricts the three log-ARCH terms (in addition to the log-variance intercept) from removal, and turns off diagnostic testing for serial correlation in the standardized residuals:

```
getsv06b <- getsv(mod06, keep = 1:4, ar.LjungB = NULL)
```

5.3 Extraction functions

There are a number of extraction functions available for 'gets' objects, i.e., objects produced by either `getsm` or `getsv`. The most important functions (most of them S3 methods) are:

```
coef, ES, fitted, logLik, paths, plot, predict, print, recursive,
residuals, rsquared, sigma, summary, terminals, toLatex, VaR, vcov
```

All, apart from `paths` and `terminals`, behave in a similar way to the corresponding extraction functions for 'arx' objects. In particular, `coef`, `fitted`, `print` and `residuals` automatically detect whether `getsm` or `getsv` has been used, and behave accordingly. The `paths` function extracts the paths searched, and `terminals` the terminal models.

5.4 Example: A parsimonious model of quarterly inflation

In Section 4.4, we showed that a log-ARCH(4)-X specification of the log-variance improved the fit and diagnostics of an AR(4)-X model of quarterly inflation. Here, we obtain a simplified version by using the `getsm` and `getsv` functions.

The estimation results of the AR(4)-X-log-ARCH(4)-X specification that we fitted was stored as an 'arx' object named `inflMod02`. The following code undertakes GETS modeling of the mean, and stores the results in an object named `inflMod03`:

```
inflMod03 <- getsm(inflMod02)
```

This produces the following during the specification search:

```
GUM mean equation:

        reg.no. keep        coef   std.error  t-stat   p-value
mconst        1    0   0.8386311   0.2961338   2.8319  0.005637
ar1           2    0   0.7257550   0.1300407   5.5810 2.211e-07
ar2           3    0   0.0195911   0.1171347   0.1673  0.867523
```

```
ar3              4   0   0.0350092  0.1385735   0.2526   0.801087
ar4              5   0  -0.1676751  0.1336972  -1.2541   0.212836
q2dum            6   0  -0.0148892  0.2333917  -0.0638   0.949266
q3dum            7   0  -0.0072972  0.2262704  -0.0322   0.974340
q4dum            8   0   0.0103990  0.2226772   0.0467   0.962849


GUM log-variance equation:


           coef std.error   t-stat  p-value
vconst  0.95935   0.53464   3.2199 0.072749
arch1   0.16697   0.10352   1.6130 0.110169
arch2   0.12027   0.10335   1.1637 0.247566
arch3   0.14740   0.10332   1.4267 0.157060
arch4   0.05982   0.10515   0.5689 0.570824
q2dum  -1.32860   0.61862  -2.1477 0.034366
q3dum  -0.92707   0.58400  -1.5874 0.115843
q4dum  -1.82736   0.62014  -2.9467 0.004069


Diagnostics:


                   Chi-sq df p-value
Ljung-Box AR(5)    9.1776  5 0.10219
Ljung-Box ARCH(5)  1.7613  5 0.88109


6 path(s) to search
Searching: 1 2 3 4 5 6


Path 1: 3 7 6 8 4 5 -5
Path 2: 4 7 6 8 3 5 -5
Path 3: 5 7 6 3 8 -8 4 -4
Path 4: 6 7 8 3 4 5 -5
Path 5: 7 6 8 3 4 5 -5
Path 6: 8 7 6 3 4 5 -5


Terminal models:


        info(sc)       logl   n k
spec 1: 1.722352 -82.59571 104 3
spec 2: 1.776284 -83.07798 104 4


Retained regressors (final model):


mconst   ar1   ar4
```

In addition to the intercept, the final model contains the AR(1) and AR(4) terms, but no quarterly dummies. So the level of quarterly year-on-year inflation does not seem to depend on

quarter. Note that, in the searched paths, regressor no. 5 (i.e., the AR(4) term) has a minus sign in front of it in all but one of the searched paths. This means the term has been re-introduced after deletion, since its deletion leads to a violation of one or several of the diagnostics tests. This is the reason the AR(4) term is retained even though it is not significant in the final model. Next, we use the residuals of the simplified model to develop a parsimonious model of the log-variance, storing the results in `inflMod05`:

```
inflMod04 <- arx(residuals(inflMod03), mc = FALSE, arch = 1:4,
   vxreg = inflData[, 2:4])
inflMod05 <- getsv(inflMod04, ar.LjungB = list(lag = 5, pval = 0.025))
```

Note that, to ensure that the diagnostic test for autocorrelation in the standardized residuals is undertaken at the same lag as earlier, the `ar.LjungB` argument has been modified. Next, typing `inflMod05` prints the results, and again we only reproduce selected parts in the interest of brevity:

```
SPECIFIC log-variance equation:

          coef std.error  t-stat  p-value
vconst  0.71311   0.53965  1.7462 0.186355
arch1   0.17438   0.10057  1.7339 0.086217
arch2   0.16822   0.10034  1.6764 0.096975
q2dum  -1.43834   0.62992 -2.2834 0.024662
q3dum  -1.09189   0.60035 -1.8187 0.072135
q4dum  -1.82836   0.60351 -3.0295 0.003163


Diagnostics and fit:

                 Chi-sq df p-value
Ljung-Box AR(5)   8.1224  5  0.1496
Ljung-Box ARCH(5) 7.7418  5  0.1711
```

The results suggest a high impact of the ARCH(1) and ARCH(2) terms – much higher than for financial returns,[9] and that the conditional variance depends on quarter. To obtain an idea of the economic importance of our results, we re-estimate the full, simplified model, and generate out-of-sample forecasts of the conditional standard deviation up to four quarters ahead. The full, simplified model is re-estimated using:

```
inflMod06 <- inflMod06 <- arx(inflData[, "infl"], ar = c(1, 4),
   arch = 1:2, vxreg = inflData[, 2:4], vcov.type = "white")
```

In order to generate out-of-sample forecasts, we first need to generate the out-of-sample values of the retained quarterly dummies:

---

[9]In finance, if ϵ_t is a mean-corrected financial return, then the ARCH(1) term is usually about 0.05, and almost never higher than 0.1.

```
newvxreg <- matrix(0, 4, 3)
colnames(newvxreg) <- c("q2dum", "q3dum", "q4dum")
newvxreg[2, "q2dum"] <- 1
newvxreg[3, "q3dum"] <- 1
newvxreg[4, "q4dum"] <- 1
```

We can now generate the out-of-sample forecasts of the conditional standard deviations:

```
set.seed(123)
predict(inflMod06, n.ahead = 4, spec = "variance", newvxreg = newvxreg)
```

The first command, `set.seed(123)`, is for reproducibility purposes, since a bootstrap procedure is used to generate variance forecasts two or more steps ahead (the number of draws can be changed via the `n.sim` argument). The forecasts for 2016(1) to 2016(4) are:

```
  2016(1)    2016(2)    2016(3)    2016(4)
1.0448239 0.3403215 0.4628250 0.2075531
```

In other words, the conditional variance is forecasted to be about four times higher in 2016(1) than in 2016(4). This has notable economic consequences. For example, if the forecasted inflation in 2016(1) is 2%, then an approximate 95% prediction interval computed as $2 \pm 2 \times \hat{\sigma}_{n+1}$ is given by the range 0% to 4%, which is large. By contrast, an approximate 95% prediction interval for 2016(4) computed as $2 \pm 2 \times \hat{\sigma}_{n+4}$ is given by the range 1.1% to 2.9%, which is much tighter.

5.5   Example: A parsimonious model of daily SP500 volatility

In Section 4.5 we estimated a rich model of daily SP500 return volatility named `sp500Mod01`. Simplification of this model is straightforward with the `getsv` function. Since the model does not fully get rid of the ARCH in the standardized residuals, we will turn off the ARCH diagnostics. Also, for parsimony we will choose a small regressor significance level equal to 0.1%:

```
sp500Mod03 <- getsv(sp500Mod01, t.pval = 0.001, arch.LjungB = NULL)
```

Typing `sp500Mod03` returns (amongst other):

```
SPECIFIC log-variance equation:
```

	coef	std.error	t-stat	p-value
vconst	0.016309	0.044960	0.1316	0.7167940
arch1	-0.064147	0.013740	-4.6687	3.080e-06
arch5	0.038690	0.011324	3.4168	0.0006368
logEqWMA(20)	0.427071	0.053110	8.0413	1.014e-15
logEqWMA(120)	0.327148	0.052734	6.2038	5.782e-10
volproxylag	0.197866	0.036558	5.4124	6.396e-08
dWed	-0.176576	0.064799	-2.7250	0.0064442

```
Diagnostics and fit:
```

```
                  Chi-sq df  p-value
Ljung-Box AR(1)    0.40681  1   0.5236
Ljung-Box ARCH(6) 32.33070  6 1.41e-05


SE of regression      1.14417
Log-lik.(n=8120) -10993.62221
```

In other words, no day-of-the-week dummies are retained and only the first ARCH-term is retained. However, three of the log-proxies are retained, i.e., the weekly, the monthly and the half-yearly, and both the lagged range-based volatility proxy and the lagged log-volume difference are retained. The log-likelihood is now -11131.4, and the following code computes the Schwarz (1978) information criterion value in terms of the average log-likelihood:

```
info.criterion(as.numeric(logLik(sp500Mod03)), n = 8120, k = 7)
```

The value is 2.7155, so so the parsimonious model provides a better fit (according to sc) compared with the GUM (i.e., `sp500Mod01`).

6 Indicator saturation

Indicator saturation has been a crucial development in GETS modeling to address the distorting influence of outliers and structural breaks (changes in parameters) in econometric models. Such parameter changes are generally of unknown magnitudes and may occur at unknown times. Indicator saturation tackles this challenge by starting from a general model allowing for an outlier or shift at every point and removing all but significant ones using GETS selection. This serves both as a method to detect outliers and breaks, as well as a generalized approach to model mis-specification testing – if the model is well-specified, then no outliers/shifts will be detected. The function `isat` conducts multi-path indicator saturation to detect outliers and location-shifts in regression models using impulse indicator saturation (IIS – see Hendry et al. 2007, and Johansen and Nielsen 2016 for a comprehensive asymptotic analysis), step-indicator saturation (SIS – see Castle et al. 2015), trend-indicator saturation (TIS – as applied in Pretis et al. 2015), and user-designed indicator saturation (UIS, or designed break functions in Pretis et al. 2016, and Schneider et al. 2017). Formulating the detection of structural breaks as a problem of model selection, a regression model is saturated with a full set of indicators which are then selected over using the general-to-specific `getsm` algorithm at a chosen level of significance `t.pval`. This approach to break detection imposes no minimum break length, and outliers can be identified jointly with structural breaks. The false-detection rate or gauge in IS is given by αk for k irrelevant indicators selected over, where $k = n$ for IIS and TIS, and $k = n - 1$ for SIS if an intercept is forced. Thus, the false-detection rate can easily be controlled by reducing α at the cost of reduced power of detecting true shifts and outliers. To ensure a low false-detection rate, the rule of thumb of setting $\alpha = \min(0.05, [1/k])$ can be used, which yields one incorrectly retained indicator in expectation for large samples, and aims for a false-detection rate below 5% in small samples. Figure 2 (and Table 8 in Appendix B) show the false-detection rate in IS using `isat` in a simple static simulation for increasing sample sizes.

The respective GUMs for a simple model of the mean of y_t using impulse-, step- and trend-indicator saturation[10] are given by

$$\text{SIS GUM:} \qquad y_t = \mu + \sum_{j=2}^{n} \delta_j 1_{\{t \geq j\}} + u_t, \tag{9}$$

$$\text{IIS GUM:} \qquad y_t = \mu + \sum_{j=1}^{n} \delta_j 1_{\{t=j\}} + u_t, \tag{10}$$

$$\text{TIS GUM:} \qquad y_t = \mu + \sum_{j=1}^{n} \delta_j 1_{\{t>j\}}(t-j) + u_t, \tag{11}$$

where n denotes the total number of observations in the sample. Indicators are partitioned into blocks based on the values of the `ratio.threshold` and `max.block.size` arguments of the `isat` function, where the block size used is the maximum of given by either criterion. Indicators retained in each block are re-combined and selected over to yield terminal models. Additional regressors that are not selected over can be included through the `mxreg` argument, where autoregressive terms in particular, can be included using the `ar` argument. Naturally different indicators can be combined, by specifying both `iis = TRUE` and `sis = TRUE` selection takes place over both impulse- as well as step-indicators. The different regimes made up of indicators (e.g., retained step-functions or impulses) weighted by their estimated coefficients describe shifts in the intercept over time – the coefficient path of the intercept. While the detection of shifts in SIS is focused on time-series analysis, IIS can be used in cross-sectional regression models to detect individual outliers (see e.g., Pretis and Jiao 2017).

The primary arguments for selection of indicators in `isat` carry over from the `getsm` function. The main differences and additional arguments are:

- `t.pval`: numeric value between 0 and 1. The significance level α used for the two-sided t-tests of the indicators in selection. The default is lower than in regular `getsm` model selection and set to 0.001 to control the number of false positives. Under the null of no outliers (or structural breaks), the irrelevance proportion or gauge (or proportion of spuriously retained indicators) is equal to αk where k is the number of indicators selected over. Thus setting $\alpha \approx 1/k$ yields one spuriously retained indicator on average under the null.

- `iis`: logical, `TRUE` or `FALSE`. If `TRUE`, then a full set of impulse indicators is added and selected over.

- `sis`: logical, `TRUE` or `FALSE`. If `TRUE`, then a full set of step indicators is added and selected over.

- `tis`: logical, `TRUE` or `FALSE`. If `TRUE`, then a full set of trend indicators is added and selected over.

---

[10]Note that specifications of step-functions are possible in SIS. Here we specify the steps as in (9), and thus for interpretation every additional step is added to the previous ones. In contrast, the paper introducing SIS (Castle *et al.*, 2015) works with step-indicators of the form $\sum_{j=2}^{n} \delta_j 1_{\{t \leq j\}}$, in which case the steps have to be subtracted from the previous sum of steps to interpret the coefficients.
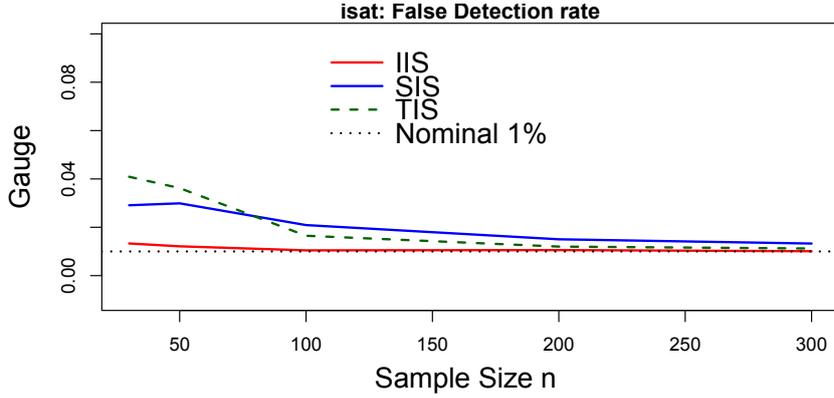
Figure 2: Gauge (false-detection rate) in IIS, SIS, TIS in `isat` under the null of no structural breaks or outliers for varying sample sizes n at nominal selection significance of $\alpha = 1\%$, using a static DGP of $y_t = u_t$ where $u_t \sim N(0, \sigma^2)$ with 1000 replications. The gauge approaches the nominal significance level of selection as $n \to \infty$.

- `uis`: matrix object that contains designed break functions to be selected over.

- `ratio.threshold`: numeric, between 0 and 1. Minimum ratio of variables in each block to total observations to determine the block size, default equals 0.8. Block size used is the maximum of given by either the `ratio.threshold` and `max.block.size`.

- `max.block.size`: an integer of at least 2. Maximum size of block of variables to be selected over, default equals 30. Block size used is the maximum of given by either the `ratio.threshold` and `max.block.size`.

- `parallel.options`: either `NULL` or an integer. The integer denotes the number of cores to be used to search over blocks in parallel. If the argument is `NULL` then no parallel computation is used. This option can speed up computation when the number of blocks of indicators to be searched over is large.

6.1 Example: Structural breaks in the growth rate of UK SO$_2$ emissions

Annual emissions of the pollutant sulphur dioxide (SO$_2$) in the UK have declined in the latter half of the 20th century due to policy interventions and changes in energy production. Here we assess whether there have been significant shifts in the growth rate $(\Delta \log(SO_2)_t)$ of sulphur dioxide emissions between 1946 and 2005, using SIS and the emission time series compiled by Smith *et al.* (2011). Setting `t.pval` to 0.01 yields an approximate gauge of $0.01k$ under the null hypothesis of no shifts for k spuriously included variables. Inclusion of a full set of indicators implies that $k = n$ for IIS, and $k = n - 1$ for SIS, and thus $0.01(n - 1) = 0.01 \times 59$. This suggests less than one indicator being retained spuriously on average in a well-specified model under the null of no shifts or outliers. Estimating an `isat` model using SIS (`sis = TRUE` is default):

```
options(plot = TRUE)
so2 <- data("so2data", package = "gets")
yso2 <- zoo(so2data[, "DLuk_tot_so2"], order.by = so2data[, "year"])
(sis <- isat(yso2, t.pval = 0.01))


SIS block 1 of 2:
30 paths to search
Searching: 1 2 3 4 ...


SIS block 2 of 2:


26 paths to search
Searching: 1 2 3 4 ...


GETS of union of retained SIS variables...
2 paths to search
Searching: 1 2


...


SPECIFIC mean equation:

                coef    std.error    t-stat      p-value
mconst    0.01465385 0.007931984   1.847438 7.026836e-02
sis1972  -0.04332051 0.011866458  -3.650669 5.990412e-04
sis1993  -0.11693333 0.020126141  -5.810023 3.625832e-07
sis1998   0.12860000 0.044305650   2.902564 5.382516e-03
sis1999  -0.28400000 0.057198348  -4.965178 7.505854e-06
sis2000   0.24550000 0.045219264   5.429102 1.441154e-06
sis2004  -0.11550000 0.035026692  -3.297485 1.746083e-03


Diagnostics:


                  Chi-sq df p-value
Ljung-Box AR(1)   0.61553  1 0.43271
Ljung-Box ARCH(1) 1.44153  1 0.22989
Jarque-Bera       0.57302  2 0.75088


SE of regression   0.04045
R-squared          0.73021
Log-lik.(n=60)   110.83192
```

The above output shows multiple detected step-shifts (labeled `sis1972`–`sis2004`) in the time series. If plotting is active (`plot = TRUE`), `isat` also displays the output as in Figure 3 plotting the observed and fitted values, together with the coefficient path (the time-varying intercept through the regimes detected using SIS) as well as the standardized residuals. There is a
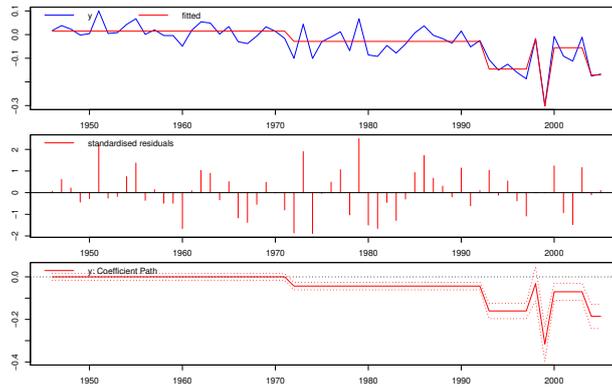
Figure 3: Annual UK SO$_2$ emission growth rate: step-indicator saturation model results. The top panel shows observed (blue) and fit time series (red). The middle panel shows the standardized residuals, the bottom panel shows the coefficient path relative to the intercept and its approximate 95% confidence interval.

downward step-shift detected in the growth rate in 1972, outlying observations are detected through two subsequent step-indicators with opposite-signs (e.g., in 1998/1999), as well as a downward step-shift at the end of the sample in 2004. This example demonstrates the flexibility of the SIS approach – step-shifts are easily identified even at the end of the sample while outliers can be detected simultaneously. The model can easily be extended to include autoregressive terms using the `ar` argument, for example we could estimate an AR(1) model with step-indicator saturation writing `isat(yso2, ar = 1, t.pval = 0.01)`. Detection of outliers and structural breaks can be directly parallelized to increase computational speed when there are a large number of blocks searched over by setting the argument `parallel.options` equal to the number of cores available for processing. For example, `isat(yso2, t.pval = 0.01, parallel.options = 2)` estimates the above model in parallel using two cores.

Additional covariates can be included in an IS regression model by including them in the `mxreg` argument. If fixed regressors entering through `mxreg` induce perfect collinear with break functions in IS, then indicators are removed automatically before selection. For example, consider forcing a hypothesized step-shift in 1972 to be retained while simultaneously searching for additional shifts throughout the sample:

```
x1972 <- zoo(sim(so2data[, "year"])[, 26], order.by = so2data[, "year"])
isat(yso2, t.pval = 0.01, mxreg = x1972)
```

The resulting estimation does not select over the fixed step-shift in 1972, though for this particular example the estimated terminal model with a forced step shift matches the SIS results of a general search.

6.2 Testing and bias correcting post-model selection in indicator saturation

The coefficient path describes how the value of a coefficient on a particular variable evolves over time. The coefficient path of the intercept of the 'isat' object can be extracted using the

31

`isatvar` function. The function returns the coefficient path both as the time-varying intercept (`const.path`) and as deviation relative to the full-sample intercept (`coef.path`), together with the approximate variance of the coefficient path computed using the approach in Pretis (2017). When the model is specified to include autoregressive terms, then `isatvar` (setting `lr = TRUE`) also returns the static long-run solution of the dynamic model with its approximate variance.

```
sisvar <- isatvar(sis)
sisvar

        coef.path  const.path    const.var     const.se
1946  0.00000000  0.01465385 6.291637e-05 0.007931984
1947  0.00000000  0.01465385 6.291637e-05 0.007931984
...
```

Indicator saturation may result in an under-estimation of the error variance as observations are "dummied out" resulting in a truncation of the distribution of the error terms. The magnitude of this effect depends on the level of significance of selection and is generally small for low values of α. This effect manifests itself in an under-estimation of the error variance, and an under-estimation of the variance of regressors not selected over. Both can be corrected when using IIS through consistency and efficiency correction factors derived in Johansen and Nielsen (2016). These correction factors are implemented in **gets** as functions `isvarcor` which corrects the estimated error variance, and `isvareffcor` for an additional correction on the estimated variance of fixed regressors. The correction factors can be applied manually to estimation results, or specified as arguments (`conscorr = TRUE` and `effcorr = TRUE`) within the `isatvar` function. This is demonstrated below running IIS on an autoregressive model with one lag (`ar = 1`) on the growth rate of SO_2 emissions. The estimated variance of the coefficient path is higher once consistency and efficiency corrections are applied:

```
iis <- isat(yso2, ar = 1, sis = FALSE, iis = TRUE, t.pval = 0.05)
isatvar(iis, conscorr = TRUE, effcorr = TRUE)

        coef.path   const.path    const.var     const.se
1947  0.00000000 -0.006210179 7.225479e-05 0.008500282
1948  0.00000000 -0.006210179 7.225479e-05 0.008500282
...

isatvar(iis,  conscorr = FALSE, effcorr = FALSE)

        coef.path   const.path    const.var     const.se
1947  0.00000000 -0.006210179 4.483453e-05 0.006695859
1948  0.00000000 -0.006210179 4.483453e-05 0.006695859
...
```

The terminal models of `isat` are the result of model selection, and may therefore lead to a selection bias in the coefficient estimates of selected variables. Post-selection bias-correction for orthogonal variables can be conducted using the method proposed in Hendry and Krolzig (2005). This is implemented as the function `biascorr`. Following Pretis (2017), bias-correction

of the coefficients in a SIS model can be directly applied to the coefficient path without prior orthogonalization. Bias-correcting the coefficient path of the above model of the growth rate of SO_2 yields the one- and two-step bias-corrected coefficients:

```
bcorr <- biascorr(b = sisvar[, "const.path"], b.se = sisvar[, "const.se"],
  p.alpha = 0.01, T = length(sisvar[, "const.path"]))

          beta   beta.1step  beta.2step
...
1997 -0.14560000 -0.14560000 -0.14560000
1998 -0.01700000 -0.01700000 -0.01700000
1999 -0.30100000 -0.30099983 -0.30099983
2000 -0.05550000 -0.04043232 -0.03000334
2001 -0.05550000 -0.04043232 -0.03000334
...
```

Bias-correction reduces the magnitude of the estimated coefficients slightly to account for potential selection bias.

The function `isattest` makes it possible to conduct hypothesis tests on the coefficient path of the intercept of an 'isat' object. This test is described in Pretis (2017) and builds on Ericsson (2017) and Pretis *et al.* (2015) who use indicator saturation as a test for time-varying forecast accuracy. The main arguments of the `isattest` function are:

- `hnull`: numeric. The null-hypothesis value to be tested against.

- `lr`: logical. If TRUE and the 'isat' object to be tested contains autoregressive terms, then the test is conducted on the long-run equilibrium coefficient path.

- `ci.pval`: numeric, between 0 and 1. The level of significance for the confidence interval and hypothesis test.

- `conscorr`: logical. If TRUE then the estimated error variance in IIS is consistency-corrected using the results in Johansen and Nielsen (2016).

- `effcorr`: logical. If TRUE then the estimated variance of fixed regressors in IIS is efficiency corrected using the results in Johansen and Nielsen (2016).

- `biascorr`: logical. If TRUE then the coefficient path is bias-corrected prior to testing. This is only valid for a non-dynamic (no auto-regressive terms) test without additional covariates.

Here we test the time-varying mean (as determined using SIS) of the annual growth rate of UK SO_2 emissions against the null hypothesis of zero-growth using `isattest`:

```
isattest(sis, hnull = 0, lr = FALSE, ci.pval = 0.99, plot.turn = TRUE,
  biascorr = TRUE)

          ci.low     ci.high bias.high   bias.low
```
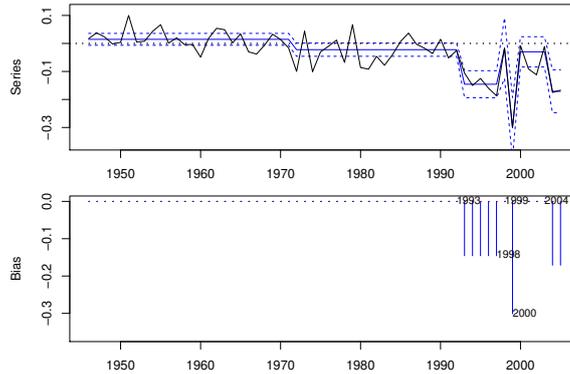
Figure 4: Hypothesis test on the annual UK SO$_2$ emission growth rate following step-indicator saturation. The top panel shows observed (black) and bias-corrected fit (blue). The bottom panel shows the periods where the null-hypothesis is rejected, together with the dates of the significant breaks.

```
1946 -0.006539007  0.035846700        0  0.0000000
1947 -0.006539007  0.035846700        0  0.0000000
1948 -0.006539007  0.035846700        0  0.0000000
...
```

The results are shown in the automatically-generated plot given in Figure 4 (the `plot.turn = TRUE` argument automatically adds the break dates into the plot in the lower panel). When testing at 1% and using bias-correction this suggests that the detected shift in 1972 does not significantly move the growth-rate away from zero. Similarly, the upward shift in 2000 moves the growth rate back to zero. This change, however, is off-set by the shift at the end of the sample which shows the growth rate turning significantly negative in 2004.

6.3 Comparison of `isat` with other methods

Indicator saturation formulates the detection of breaks and outliers as a problem of model (or variable) selection. Here we first provide an overview of software implementing indicator saturation, followed by a discussion of `isat` in **gets** relative to other existing break detection packages.

The only other currently existing software implementing indicator saturation is **Autometrics**. IIS and SIS are available in both **Autometrics** and **gets**, however, a crucial difference within SIS is the construction and subsequent interpretation of step-indicators. In **gets** steps are constructed as forward-steps: $\mathbf{S} = \left(1_{\{t \geq j\}}, j = 1, \ldots, n\right)$, where $1_{\{t \geq j\}}$ denotes the indicator function such that $1_{\{t \geq j\}} = 1$ for observations from j onwards, and zero otherwise. Thus the signs of the coefficients in the retained regression model in **gets** correspond to the direction of the step: positive (negative) coefficients imply an upward (downward) step, and the coefficient path begins with the regression intercept where for each subsequent regime the coefficient on the step indicator is added to the full sample intercept. **Autometrics** relies on backward-steps: $\mathbf{S} = \left(1_{\{t \leq j\}}, j = 1, \ldots, n\right)$ and thus step-coefficients have to be interpreted as opposite-signed relative to the reported regression coefficients. **Autometrics** currently has no implementation

34

of the computation of the coefficient path and its approximate variance, thus testing on the different regimes is non-trivial. This is directly implemented in **gets** by automatically plotting the coefficient path (if `plot = TRUE`), which can be extracted using `isatvar`. The variance estimates in **Autometrics** are currently not consistency or efficiency corrected when using IS. This is implemented in **gets** and – together with the extraction of the coefficient path and its variance – enables testing on the coefficient path using the `isattest` function, together with automatic bias-correction if specified. Further, automatic trend-indicator saturation (TIS) is currently only available in **gets**. Both **Autometrics** and **gets** enable the selection over designed break functions – through the argument `uis` in **gets** and the general more variables than observations model selection algorithm in **Autometrics**.

In the broader field of detection of breaks or changepoints, the main difference to existing methods (e.g., Bai and Perron 1998, Bai and Perron 2003, Perron 2006 implemented in **strucchange** by Kleiber *et al.* 2002) or detection of changepoints in general (as in the package **changepoint** – see Killick and Eckley 2014) is the model-selection approach to break detection in indicator saturation (for discussion of methodological differences see Castle *et al.* 2015, as well as Hendry *et al.* 2007, and Johansen and Nielsen 2016). This makes it possible to detect outliers (single period shifts) jointly with structural breaks (multiple period shifts), further it is also possible to detect breaks using designed functions (Pretis *et al.*, 2016) which is not possible in conventional structural break methods or changepoint analysis.

Where the indicator saturation methodology overlaps in applications with existing methods is the detection of shifts in the intercept of time series regression models, for example using `breakpoints` in **strucchange**. Relative to **strucchange** and the Bai and Perron least-squares approach in changes in the mean, `isat` in **gets** does not impose a minimum break length and can therefore conduct outlier detection jointly with shifts in the intercept, further there is no implicit upper limit on the number of breaks, and it is thus possible to identify outliers or shifts in the mean at the start or end of samples as no trimming is required. Changes in regression coefficients on random variables can be detected in `isat` using designed break functions through the `uis` argument by interacting a full set of step-indicators with the random variable. This, however, is computationally expensive as each additional variable whose coefficient is allowed to break over time adds a set of n variables to be selected over the GUM. The function `breakpoints` in **strucchange** estimates a pure structural change model where all coefficients change, `isat` in **gets** is a partial model where the coefficients on variables included through `mxreg` are not allowed to break, and only breaks in the mean (or specified coefficients through inclusion in `uis`) are permitted – making it possible to pre-specify constancy. A partial structural change model using the Bai and Perron least-squares approach can be estimated using available MATLAB code.[11]

Relative to **changepoint**, `isat` in **gets** is focused on regression modeling and structural breaks in the intercept of regression models jointly with outlier detection. As the authors of **changepoint** themselves note, **changepoint** does not focus on changes in regression models. The function `isat` directly enables the inclusion of covariates through `mxreg` or `ar` within `isat`, only if no additional covariates are specified then `isat` searches for changes in the mean of a time series as in the models used in the **changepoint** package while, however, simultaneously detecting outliers.

---

[11]Available online at http://people.bu.edu/perron/code.html.

7 Exporting results to **EViews**, **STATA** and LaTeX

The two most popular commercial econometric software packages are EViews (IHS Markit, 2017) and STATA (StataCorp, 2017), but none of these provide GETS modeling capabilities. To facilitate the usage of GETS modeling for EViews and STATA users, we provide two functions for this purpose, `eviews` and `stata`. Both functions work in a similar way, and both can be applied on either 'arx', 'gets' or 'isat' objects. For example, typing `eviews(getsm05)` yields the following print output:

```
EViews code to estimate the model:

  equation getsm05.ls(cov = white) yy mxreg4

R code (example) to export the data of the model:

  eviews(getsm05, file = 'C:/Users/myname/Documents/getsdata.csv')
```

In other words, the code to estimate the final model in EViews, and – if needed – a code-suggestion for how to export the data of the model. The need to export the data of the final model is likely to be most relevant subsequent to the use of `isat`. The `stata` function works similarly. Note that both the `eviews` and `stata` functions are only applicable to conditional mean specifications, since neither EViews nor STATA offer the estimation of dynamic log-variance models.

The objects returned by `arx`, `getsm`, `getsv` and `isat` are lists. The entries in these lists that contain the main estimation output are objects of class 'data.frame'. That means the R package **xtable** of Dahl (2016) can be used to generate LaTeX code of the data frames.

8 Conclusions

The R package **gets** provides a toolkit for general-to-specific modeling through automatic variable selection in regression specifications of the mean and the variance, as well as indicator saturation methods to detect outliers and structural breaks. Starting with a general candidate set of variables unknown to be relevant or irrelevant, selection using `getsm` or `getsv` can yield parsimonious terminal models that satisfy a set of chosen diagnostic criteria, where parameter changes and outlying observations are detected using `isat`.

References

Akaike H (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control*, **19**(6), 716–723. https://doi.org/10.1109/tac.1974.1100705.

Bai J, Perron P (1998). "Estimating and Testing Linear Models with Multiple Structural Changes." *Econometrica*, **66**(1), 47–78. https://doi.org/10.2307/2998540.

Bai J, Perron P (2003). "Computation and Analysis of Multiple Structural Change Models." *Journal of Applied Econometrics*, **18**(1), 1–22. https://doi.org/10.1002/jae.659.

Campos J, Hendry DF, Ericsson NR (eds.) (2005). *General-to-Specific Modeling. Volumes 1 and 2.* Edward Elgar Publishing, Cheltenham.

Castle JL, Doornik JA, Hendry DF (2011). "Evaluating Automatic Model Selection." *Journal of Time Series Econometrics*, **3**(1), 1–31. https://doi.org/10.2202/1941-1928.1097.

Castle JL, Doornik JA, Hendry DF, Pretis F (2015). "Detecting Location Shifts During Model Selection by Step-Indicator Saturation." *Econometrics*, **3**(2), 240–264. https://doi.org/10.3390/econometrics3020240.

Dahl DB (2016). **xtable**: *Export Tables to LaTeX or HTML*. R package version 1.8-2, https://CRAN.R-project.org/package=xtable.

Doornik J (2006). *Ox: An Object Oriented Matrix Programming Language.* 5th edition. Timberlake Consultants Ltd., London.

Doornik J (2009). "**Autometrics**." In JL Castle, N Shephard (eds.), *The Methodology and Practice of Econometrics: A Festschrift in Honour of David F. Hendry*, pp. 88–121. Oxford University Press, Oxford.

Doornik JA, Hendry DF (2007a). *Empirical Econometric Modelling –* **PcGive** *12: Volume I.* Timberlake Consultants Ltd., London.

Doornik JA, Hendry DF (2007b). *Empirical Econometric Modelling –* **PcGive** *12: Volume II.* Timberlake Consultants Ltd., London.

Duan N (1983). "Smearing Estimate: A Nonparametric Retransformation Method." *Journal of the Americal Statistical Association*, **78**(383), 605–610. https://doi.org/10.2307/2288126.

Efron B, Hastie T, Johnstone I, Tibshirani R (2004). "Least Angle Regression." *The Annals of Statistics*, **32**(2), 407–499. https://doi.org/10.1214/009053604000000067.

Engle R (1982). "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflations." *Econometrica*, **50**(4), 987–1008. https://doi.org/10.2307/1912773.

Ericsson NR (2017). "How Biased Are US Government Forecasts of the Federal Debt?" *International Journal of Forecasting*, **33**(2), 543–559. https://doi.org/10.1016/j.ijforecast.2016.09.001.

Francq C, Sucarrat G (2018). "An Exponential Chi-Squared QMLE for Log-GARCH Models Via the ARMA Representation." *Journal of Financial Econometrics*, **16**(1), 129–154. https://doi.org/10.1093/jjfinec/nbx032.

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. https://doi.org/10.18637/jss.v033.i01.

Glosten LR, Jagannathan R, Runkle DE (1993). "On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks." *Journal of Finance*, **48**(5), 1779–1801. https://doi.org/10.2307/2329067.

Hannan EJ, Quinn BG (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society B*, **41**(2), 190–195.

Hastie T, Efron B (2013). **lars***: Least Angle Regression, Lasso and Forward Stagewise.* R package version 1.2, https://CRAN.R-project.org/package=lars.

Hendry DF (2003). "J. Denis Sargan and the Origins of LSE Econometric Methodology." *Econometric Theory*, **19**(3), 457–480. https://doi.org/10.1017/s0266466603193061.

Hendry DF, Doornik J (2014). *Empirical Model Discovery and Theory Evaluation.* The MIT Press, London.

Hendry DF, Johansen S (2015). "Model Discovery and Trygve Haavelmo's Legacy." *Econometric Theory*, **31**(1), 93–114. https://doi.org/10.1017/s0266466614000218.

Hendry DF, Johansen S, Santos C (2007). "Automatic Selection of Indicators in a Fully Saturated Regression." *Computational Statistics*, **23**(2), 317–335. https://doi.org/10.1007/s00180-007-0054-z.

Hendry DF, Krolzig HM (2001). *Automatic Econometric Model Selection Using* **PcGets**. Timberlake Consultants Press, London.

Hendry DF, Krolzig HM (2005). "The Properties of Automatic Gets Modelling." *The Economic Journal*, **115**(502), C32–C61. https://doi.org/10.1111/j.0013-0133.2005.00979.x.

Hoover KD, Perez SJ (1999). "Data Mining Reconsidered: Encompassing and the General-to-Specific Approach to Specification Search." *The Econometrics Journal*, **2**(2), 167–191. https://doi.org/10.1111/1368-423x.00025.

IHS Markit (2017). *EViews Version 9.5.* IHS Markit, Irvine, CA. http://www.eviews.com/.

Jarque C, Bera A (1980). "Efficient Tests for Normality, Homoskedasticity, and Serial Independence." *Economics Letters*, **6**(3), 255–259. https://doi.org/10.1016/0165-1765(80)90024-5.

Johansen S, Nielsen B (2016). "Asymptotic Theory of Outlier Detection Algorithms for Linear Time Series Regression Models." *Scandinavian Journal of Statistics*, **43**(2), 321–348. https://doi.org/10.1111/sjos.12174.

Killick R, Eckley I (2014). "**changepoint**: An R Package for Changepoint Analysis." *Journal of Statistical Software*, **58**(3), 1–19. https://doi.org/10.18637/jss.v058.i03.

Kleiber C, Hornik K, Leisch F, Zeileis A (2002). "**strucchange**: An R Package for Testing for Structural Change in Linear Regression Models." *Journal of Statistical Software*, **7**(2), 1–38. https://doi.org/10.18637/jss.v007.i02.

Ljung GM, Box GEP (1978). "On a Measure of Lack of Fit in Time Series Models." *Biometrika*, **65**(2), 297–303. https://doi.org/10.2307/2335207.

Lovell MC (1983). "Data Mining." *The Review of Economics and Statistics*, **65**(1), 1–12. https://doi.org/10.2307/1924403.

Mizon G (1995). "Progressive Modeling of Macroeconomic Time Series: The LSE Methodology." In KD Hoover (ed.), *Macroeconometrics. Developments, Tensions and Prospects*, pp. 107–169. Kluwer Academic Publishers, Dordrecht.

Newey W, West K (1987). "A Simple Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix." *Econometrica*, **55**(3), 703–708. https://doi.org/10.2307/1913610.

Perron P (2006). "Dealing with Structural Breaks." In H Hassani, TC Mills, K Patterson (eds.), *Palgrave Handbook of Econometrics: Volume 1*, pp. 278–352. Palgrave MacMillan, Hampshire, UK.

Pretis F (2017). "Classifying Time-Varying Predictive Accuracy in Using Bias-Corrected Indicator Saturation." University of Oxford Economics Discussion Paper.

Pretis F, Jiao X (2017). "Testing the Presence of Outliers to Assess Misspecification in Regression Models." University of Oxford Economics Discussion Paper.

Pretis F, Mann ML, Kaufmann RK (2015). "Testing Competing Models of the Temperature Hiatus: Assessing the Effects of Conditioning Variables and Temporal Uncertainties Through Sample-Wide Break Detection." *Climatic Change*, **131**(4), 705–718. https://doi.org/10.1007/s10584-015-1391-5.

Pretis F, Reade JJ, Sucarrat G (2018). "Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks." *Journal of Statistical Software*, **86**(3), 1–44. https://doi.org/10.18637/jss.v086.i03.

Pretis F, Schneider L, Smerdon JE, Hendry DF (2016). "Detecting Volcanic Eruptions in Temperature Reconstructions by Designed Break-Indicator Saturation." *Journal of Economic Surveys*, **30**(3), 403–429. https://doi.org/10.1111/joes.12148.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Schneider L, Smerdon J, Pretis F, Hartl-Meier C, Esper J (2017). "A New Archive of Large Volcanic Events Over the Past Millennium Derived from Reconstructed Summer Temperatures." *Environmental Research Letters*, **12**(9), 094005. https://doi.org/10.1088/1748-9326/aa7a1b.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2), 461–464. https://doi.org/10.1214/aos/1176344136.

Smith SJ, van Aardenne J, Klimont Z, Andres RJ, Volke A, Delgado Arias S (2011). "Anthropogenic Sulfur Dioxide Emissions: 1850–2005." *Atmospheric Chemistry and Physics*, **11**(3), 1101–1116. https://doi.org/10.5194/acp-11-1101-2011.

StataCorp (2017). *STATA Statistical Software: Release 15*. StataCorp LLC, College Station, TX. http://www.stata.com/.

Sucarrat G (2010). "Econometric Reduction Theory and Philosophy." *Journal of Economic Methodology*, **17**(1), 53–75. https://doi.org/10.1080/13501780903528978.

Sucarrat G (2015a). **AutoSEARCH**: *General-to-Specific (GETS) Modelling*. R package version 1.5, https://CRAN.R-project.org/package=AutoSEARCH.

Sucarrat G (2015b). **lgarch**: *Simulation and Estimation of Log-GARCH Models*. R package version 0.6-2, https://CRAN.R-project.org/package=lgarch/.

Sucarrat G, Escribano Á (2012). "Automated Model Selection in Finance: General-to-Specific Modelling of the Mean and Volatility Specifications." *Oxford Bulletin of Economics and Statistics*, **74**(5), 716–735. https://doi.org/10.1111/j.1468-0084.2011.00669.x.

Sucarrat G, Grønneberg S, Escribano Á (2016). "Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density Is Unknown." *Computational Statistics & Data Analysis*, **100**, 582–594. https://doi.org/10.1016/j.csda.2015.12.005.

Sucarrat G, Pretis F, Reade J (2018). **gets**: *General-to-Specific (GETS) Modelling and Indicator Saturation Methods*. R package version 0.15, https://CRAN.R-project.org/package=gets.

The MathWorks Inc (2017). *MATLAB – The Language of Technical Computing, Version R2017b*. Natick, Massachusetts. http://www.mathworks.com/products/matlab/.

Tibshirani R (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society B*, **58**(1), 267–288.

White H (1980). "A Heteroskedasticity-Consistent Covariance Matrix and a Direct Test for Heteroskedasticity." *Econometrica*, **48**(4), 817–838. https://doi.org/10.2307/1912934.

Zeileis A, Grothendieck G (2005). "**zoo**: S3 Infrastructure for Regular and Irregular Time Series." *Journal of Statistical Software*, **14**(6), 1–27. https://doi.org/10.18637/jss.v014.i06.

Label	k_{rel}	k_{irr}	Simulation DGP	GUM
HP1	0	40	$r_t = \epsilon_t,$ $\epsilon_t = 130 z_t, \quad z_t \overset{\mathrm{iid}}{\sim} N(0,1)$	$r_t = \sum_{k=1}^{41} \psi_k x_{kt}^{\mathrm{HP}} + \epsilon_t,$ where $x_{37t}^{\mathrm{HP}} = r_{t-1},$ $x_{38t}^{\mathrm{HP}} = r_{t-2},\ x_{39t}^{\mathrm{HP}} = r_{t-3},$ $x_{40t}^{\mathrm{HP}} = r_{t-4}$ and $x_{41t}^{\mathrm{HP}} = 1$
HP2'	1	39	$r_t = 0.75 r_{t-1} + \epsilon_t,$ $\epsilon_t = 85.99 z_t, \quad z_t \overset{\mathrm{iid}}{\sim} N(0,1)$	Same as in HP1.
HP7'	3	37	$r_t = 0.75 r_{t-1} + 1.33 x_{11t}^{\mathrm{HP}} - 0.9975 x_{29t}^{\mathrm{HP}} + \epsilon_t,$ $\epsilon_t = 6.44 z_t, \quad z_t \overset{\mathrm{iid}}{\sim} N(0,1)$	Same as in HP1.

Table 4: List of experiments.

A Hoover and Perez (1999) simulations

Table 4 contains the list of experiments. The design of the experiments HP1, HP2' and HP7' are based on Hoover and Perez (1999, see Table 3 on p. 174), and make use of their data $x_{1t}^{\mathrm{HP}}, \ldots, x_{36t}^{\mathrm{HP}}$. It should be noted that there are two typos in their Table 3. The value $\sqrt{(7/4)}$ should instead be $\sqrt{7}/4$ in the model of the autoregressive error, and the value 6.73 should instead be 6.44 in model 7', see also Doornik (2009). The number of relevant variables in the GUM is k_{rel}, the number of irrelevant variables in the GUM is k_{irr} and the total number of variables (the constant included) in the GUM is $k = k_{\mathrm{rel}} + k_{\mathrm{irr}} + 1$. Nominal regressor significance level used is 5%, and 1000 replications. The term $m(\hat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$ is the average proportion of relevant variables $\hat{k}_{\mathrm{rel}}$ retained relative to the actual number of relevant variables k_{rel} in the DGP. The term $m(\hat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$ denotes the average proportion of irrelevant variables $\hat{k}_{\mathrm{irr}}$ retained relative to the actual number of irrelevant variables k_{irr} in the GUM. The estimate $\hat{k}_{\mathrm{irr}}$ includes both significant and insignificant retained irrelevant variables (this is in line with Hendry and Krolzig (2005), and Doornik (2009), but counter to HP which only includes significant irrelevant variables in the estimate). $\hat{p}(\mathrm{DGP})$ is the proportion of times the DGP is found exactly. The properties of the HP algorithm are from Hoover and Perez (1999, Table 4 on p. 179). The properties of the **PcGets** algorithm are from Hendry and Krolzig (2005, Figure 1 on p. C39), and the properties of the Autometrics algorithm are from Doornik (2009, Section 6). For **AutoSEARCH**, a constant is included in all the regressions but ignored in the evaluation of successes and failures (this is in line with Hoover and Perez (1999) but counter to Hendry and Krolzig (2005), and Doornik (2009)), in the diagnostic checks both the AR and ARCH test of the standardized residuals were undertaken at lag 2 using a significance level of 2.5% for each, and as tiebreaker the Schwarz information criterion is used with a Gaussian log-likelihood made up of the standardized residuals of the mean specification.

B Simulation tables

Tables 5, 6, 7 and 8 present the simulation results comparing **gets** to alternative variable selection methods, and the properties of `isat` under the null of no breaks.

$\rho = 0$		Gauge, $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$				Potency, $m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$				
k_{rel}	k_{irr}	getsm	LassCV	LassFix	1-cut	getsm	LassCV	LassFix	1-cut	DGP
0	20	0.012	0.081	0.014	0.010					
1	19	0.013	0.138	0.018	0.011	0.623	0.779	0.657	0.507	0.616
2	18	0.014	0.206	0.021	0.010	0.619	0.875	0.690	0.498	0.616
3	17	0.015	0.272	0.024	0.011	0.635	0.913	0.705	0.544	0.639
4	16	0.015	0.324	0.027	0.010	0.607	0.929	0.675	0.509	0.615
5	15	0.014	0.359	0.026	0.010	0.593	0.938	0.667	0.505	0.604
6	14	0.017	0.425	0.032	0.011	0.592	0.955	0.659	0.513	0.602
7	13	0.020	0.463	0.034	0.011	0.591	0.967	0.660	0.517	0.597
8	12	0.020	0.517	0.038	0.011	0.589	0.965	0.642	0.520	0.591
9	11	0.021	0.568	0.038	0.008	0.592	0.975	0.653	0.516	0.587
10	10	0.025	0.587	0.040	0.010	0.588	0.977	0.628	0.507	0.576

Table 5: Variable selection properties of getsm algorithm in **gets** compared to alternative algorithms when regressors are uncorrelated in expectation. Variable selection in getsm is undertaken with a nominal significance level of 1%. For a total number of $k = 20$ variables, $m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$ defines the average proportion of relevant variables $\widehat{k}_{\mathrm{rel}}$ retained relative to the actual number of relevant variables k_{rel}. $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$, average proportion of irrelevant variables $\widehat{k}_{\mathrm{irr}}$ retained relative to the actual number of irrelevant variables k_{irr} in the GUM. LassCV denotes the the cross-validated LASSO using **glmnet**, LassFix denotes the LASSO where the penalty parameter is chosen to match the gauge of getsm under the null. Column 1-cut denotes 1-cut selection of the GUM where all variables with p values $< 1\%$ are retained in a single decision, DGP denotes the proportion of variables retained if the DGP is correctly estimated and t-tests are conducted at the 1% level. Simulation undertaken on a sample of $n = 100$ observations using 1000 replications.

$\rho = 0.5$		Gauge, $m(\widehat{k}_{\text{irr}}/k_{\text{irr}})$				Potency, $m(\widehat{k}_{\text{rel}}/k_{\text{rel}})$				
k_{rel}	k_{irr}	getsm	LassCV	LassFix	1-cut	getsm	LassCV	LassFix	1-cut	DGP
0	20	0.018	0.074	0.009	0.011					
1	19	0.025	0.154	0.043	0.009	0.502	0.752	0.616	0.257	0.654
2	18	0.026	0.232	0.095	0.010	0.505	0.874	0.782	0.258	0.498
3	17	0.029	0.273	0.139	0.009	0.508	0.911	0.847	0.248	0.418
4	16	0.031	0.307	0.169	0.010	0.525	0.921	0.870	0.262	0.388
5	15	0.032	0.338	0.202	0.010	0.522	0.932	0.887	0.247	0.353
6	14	0.038	0.378	0.229	0.011	0.525	0.938	0.899	0.255	0.344
7	13	0.036	0.394	0.249	0.009	0.536	0.946	0.912	0.263	0.336
8	12	0.043	0.421	0.281	0.010	0.523	0.947	0.915	0.251	0.328
9	11	0.046	0.440	0.305	0.011	0.527	0.954	0.928	0.250	0.315
10	10	0.048	0.447	0.304	0.011	0.526	0.958	0.933	0.254	0.308

Table 6: Variable selection properties of getsm algorithm in **gets** compared to alternative algorithms when regressors are positively correlated ($\rho = 0.5$) in expectation. Variable selection in getsm is undertaken with a nominal significance level of 1%. For a total number of $k = 20$ variables, $m(\widehat{k}_{\text{rel}}/k_{\text{rel}})$ defines the average proportion of relevant variables $\widehat{k}_{\text{rel}}$ retained relative to the actual number of relevant variables k_{rel}. $m(\widehat{k}_{\text{irr}}/k_{\text{irr}})$, average proportion of irrelevant variables $\widehat{k}_{\text{irr}}$ retained relative to the actual number of irrelevant variables k_{irr} in the GUM. LassCV denotes the the cross-validated LASSO using **glmnet**, LassFix denotes the LASSO where the penalty parameter is chosen to match the gauge of getsm under the null. Column 1-cut denotes 1-cut selection of the GUM where all variables with p values $< 1\%$ are retained in a single decision, DGP denotes the proportion of variables retained if the DGP is correctly estimated and t-tests are conducted at the 1% level. Simulation undertaken on a sample of $n = 100$ observations using 1000 replications.

$\rho = \pm 0.5$		Gauge, $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$				Potency, $m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$				
k_{rel}	k_{irr}	getsm	LassCV	LassFix	1-cut	getsm	LassCV	LassFix	1-cut	DGP
0	20	0.018	0.080	0.010	0.010					
1	19	0.023	0.156	0.041	0.011	0.505	0.756	0.602	0.255	0.635
2	18	0.022	0.165	0.009	0.009	0.376	0.558	0.172	0.241	0.457
3	17	0.025	0.218	0.035	0.008	0.393	0.695	0.360	0.254	0.409
4	16	0.023	0.259	0.008	0.010	0.375	0.685	0.168	0.253	0.385
5	15	0.026	0.309	0.026	0.009	0.375	0.751	0.296	0.262	0.369
6	14	0.024	0.353	0.008	0.010	0.369	0.765	0.181	0.255	0.354
7	13	0.027	0.387	0.026	0.009	0.356	0.807	0.259	0.249	0.336
8	12	0.029	0.418	0.012	0.011	0.339	0.806	0.164	0.249	0.316
9	11	0.032	0.464	0.023	0.010	0.355	0.846	0.244	0.255	0.322
10	10	0.031	0.484	0.012	0.010	0.347	0.849	0.174	0.246	0.306

Table 7: Variable selection properties of getsm algorithm in **gets** compared to alternative algorithms when regressors are alternatingly positively and negatively correlated ($\rho = \pm 0.5$) in expectation. Variable selection in getsm is undertaken with a nominal significance level of 1%. For a total number of $k = 20$ variables, $m(\widehat{k}_{\mathrm{rel}}/k_{\mathrm{rel}})$ defines the average proportion of relevant variables $\widehat{k}_{\mathrm{rel}}$ retained relative to the actual number of relevant variables k_{rel}. $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$, average proportion of irrelevant variables $\widehat{k}_{\mathrm{irr}}$ retained relative to the actual number of irrelevant variables k_{irr} in the GUM. LassCV denotes the the cross-validated LASSO using **glmnet**, LassFix denotes the LASSO where the penalty parameter is chosen to match the gauge of getsm under the null. Column 1-cut denotes 1-cut selection of the GUM where all variables with p values $\leq 1\%$ are retained in a single decision, DGP denotes the proportion of variables retained if the DGP is correctly estimated and t-tests are conducted at the 1% level. Simulation undertaken on a sample of $n = 100$ observations using 1000 replications.

No. breaks	Gauge, $m(\widehat{k}_{\mathrm{irr}}/k_{\mathrm{irr}})$		
Sample n	IIS	SIS	TIS
30	0.013	0.028	0.042
50	0.012	0.029	0.037
100	0.011	0.019	0.016
200	0.011	0.015	0.011
300	0.010	0.013	0.011

Table 8: Gauge (false-detection rate) in IIS, SIS, TIS in isat under the null of no structural breaks or outliers for varying sample sizes n at nominal selection significance of $\alpha = 1\%$ using a static DGP of $y_t = u_t$ where $u_t \sim N(0, \sigma^2)$ with 1000 replications.