Electronic Journal of Statistics
Vol. 16 (2022) 5534–5569
ISSN: 1935-7524
https://doi.org/10.1214/22-EJS2068

Casting vector time series: algorithms for forecasting, imputation, and signal extraction

Tucker McElroy

Research and Methodology Directorate U.S. Census Bureau, 4600 Silver Hill Road, Washington, D.C. 20233
e-mail: tucker.s.mcelroy@census.gov

Abstract: Recursive algorithms, based upon the nested structure of Toeplitz covariance matrices arising from stationary processes, are presented for the efficient computation of multi-step ahead forecast error covariances for non-stationary vector time series. Further, we discuss time reversal to forecast the past, and a filtering algorithm for imputation of missing values. These quantities are required to quantify multi-step ahead forecast error and signal extraction error. An information filter is presented, which provides imputations for arbitrary patterns of missing values (such as ragged edge patterns occurring in mixed frequency data). The methods are applied to multivariate retail data exhibiting trend dynamics and seasonality.

MSC2020 subject classifications: Primary 62M10.
Keywords and phrases: Durbin-Levinson algorithm, missing values, ragged edge, Wiener-Kolmogorov.

Received November 2021.

1. Introduction

This paper has been motivated by the inadequacies of standard algorithms for signal extraction of moderate dimension high-frequency data. In Spring 2016 the U.S. Census Bureau acquired daily retail data from FirstData – having tabulated the daily regional time series from credit card transaction data – with the intention of obtaining a superior understanding of the impact of holidays and calendrical effects upon associated monthly time series. These daily retail series could not be adequately modeled with SARIMA models, because the autocorrelation patterns for each of the seven weekly day-of-week series are substantially different. This behavior prompted modeling the embedded series (treating it as a seven-variate weekly series), but additional algorithms for the resulting "ragged edges" ([9] and [19]) were required. Moreover, it seemed desirable to apply *ad hoc* filters to separate the components, given that fitting component models would likely require many parameters (the SARIMA model already has 126 parameters); however, for measures of uncertainty it is required to obtain error covariances of the forecasts. Hence, new algorithmic research became necessary, providing the genesis for this paper.

A key application of time series analysis is the estimation of projections under a linear process framework [14]. Recursive computation of one-step ahead predictors is the foundation of the Durbin-Levinson algorithm for likelihood evaluation, and can similarly be utilized to provide forecasts and aftcasts[1] of the time series sample. While this framework is powerful, there are a few algorithmic gaps in the case of processes not embeddable in a state space form (SSF). First, efficient computation of multi-step ahead forecast and aftcast error variances is needed to quantify prediction error; second, the casting error covariances are needed to quantify signal extraction error; third, efficient computation of midcasts (i.e., predictions of missing values) and their prediction error variances is needed both for pre-processing and to correctly quantify signal extraction error. This paper provides new algorithms applicable to difference-stationary multivariate time series (i.e., processes that are covariance stationary after the application of a matrix differencing polynomial).

Many processes can indeed be embedded in SSF, and algorithms – such as the Kalman filter and state space smoother – are available to efficiently provide estimates and error variances ([15] and [14]). However, without making special adaptations of these algorithms the error covariances are not available, which are featured in the error variance for *ad hoc* filters applied to forecast extended data; such an application cannot be simply handled in SSF. Moreover, when many latent components are present the state vector of the SSF has a high dimension, which tends to slow algorithmic speed. (For instance, daily time series may likely involve models requiring at least 365 components in the state vector.) On the other hand, some processes of interest cannot be embedded in SSF (without some truncation), e.g., long memory processes and Vector Exponential processes [17]. The aim of this paper is to provide efficient computation for all these cases.

The approach of this paper relies on an older methodology – going back to Levinson – that focuses on first computing the autocovariances of a data process (this may be obtained by appropriately aggregating the autocovariances of the component processes) followed by recursive computation of multi-step ahead predictors as well as error covariances. While recursive procedures for multi-step ahead predictors are well-known ([23] provides a review), this paper provides new recursions where there are fairly arbitrary[2] patterns of missing values. The algorithm only needs to store predictions and prediction error covariances for casted values, corresponding to either those times interior to the sample – where an imputation is needed (i.e., a midcast) – or those times exterior to the sample – where a forecast/aftcast is needed. A novel feature is that a backward pass, or sweep, through the sample is needed whenever the first d observations contain some missing values; this backward pass utilizes a time reversal of the process' dynamics by conjugating the spectral density.

With these predictions and prediction error covariances one can obtain all desired casts, and as a corollary obtain the Gaussian divergence (-2 times the

[1] Also called backcasts; here we utilize the term aft, which is an antonym of fore.

[2] A requirement for the Gaussian likelihood to factor, and hence be computable, is that at least d contiguous values are present, where d is the order of differencing polynomial for the process.

log Gaussian likelihood, with constants removed) and residuals. One can then apply the Wiener-Kolmogorov (WK) filter [2] or *ad hoc* filters to the extended data, and obtain the signal extraction mean square error (MSE) at each time point (it is known to be higher at the sample boundaries). This application requires forecasts, aftcasts, and midcasts – henceforth, casts – with associated error covariances, although if a practitioner is pressed for speed and there are no missing values, an underestimate of the signal extraction MSE can be obtained that ignores the contribution of prediction error covariances. A related issue arises when data have a ragged edge, because the start and end dates for the component time series are not aligned. More generally, the patterns of missing values may vary series by series – this situation is awkward to handle in multivariate SSF, but is addressed through the new algorithms of this paper.

All results of this paper have been produced through the *Ecce Signum* package of R routines, which are available from the author's github[3]. Material on stochastic processes and filtering are provided in Section 2 below, while Section 3 describes the algorithms: aftcasting with time reversal, the general recursive casting algorithm, and signal extraction mean squared error. Section 4 provides an illustration, a simulation study, and an application to daily retail data. Proofs are in Appendix A, a discussion of related algorithms is given in Appendix B, and WK filter computations are presented in Appendix C.

2. Framework

2.1. *Process and likelihood factorization*

The context of this paper involves N-dimensional vector time series $\{x_t\}$ that can be described as an aggregation of latent processes. These can be non-linear and/or non-Gaussian processes, although we consider optimal *linear* estimators in the space of finite variance random variables. We suppose that there exists a matrix polynomial $\delta(z)$ of order d, written $\delta(z) = \sum_{j=0}^{d} \delta_j \, z^j$, such that $\det \delta(z) = 0$ implies $|z| = 1$, and such that

$$\underline{x}_t = \delta(B) x_t \tag{2.1}$$

is a weakly stationary time series with autocovariance function (acvf) $\gamma(h) = \mathrm{Cov}[\underline{x}_{t+h}, \underline{x}_t]$. (Although weak stationarity could be tested, as in [18], it is unclear how to extend such procedures when the data has ragged edge missing values.) Here B is the backshift operator – see [26] for background. Although the leading coefficient δ_0 need not be the identity matrix I_N, we always assume that δ_0 and δ_d are invertible matrices. A special case is given where $\delta(z)$ is a scalar polynomial times I_N, so that all component series have the same form of non-stationarity; the current version of *Ecce Signum* presumes this special case.

The spectral density matrix is defined as the Fourier transform of the acvf sequence, and is Hermitian and non-negative definite: $f_x(\lambda) = \sum_{h \in \mathbb{Z}} \gamma(h) \, e^{-ih\lambda}$.

[3] https://github.com/tuckermcelroy/Casting

While the process can have a nonzero mean in practice, our discussion here presumes the expectation of x_t has already been removed; this will simplify our recursions, and involves no loss of generality. This is because in a typical application various regressors would be stipulated to capture the mean (see discussion in [10]) – these are functions of time t that must not be in the null space of $\delta(B)$, in order to avoid identification problems. For any type of casting we modify de-meaned data by formulas involving covariances, and then add fixed effects back; *Ecce Signum* uses these routines with arbitrary sets of regressors (which can be different for each component series).

The second cumulants of $\{x_t\}$ are not fully specified by f_x; we also need to make assumptions about d initial values [3]. We have some freedom to choose the initial values, but it is crucial that there be d contiguous values in the sample; the univariate case is discussed in [24], where it is shown that without this assumption it is not possible to factorize the Gaussian likelihood in such a way that its gradient only depends upon the parameters governing f_x. For example, suppose $d = 7$ and $T = 20$, and the time points $t \in \{5, 6, 7, 8, 9, 10, 18, 19\}$ are missing. Then the only set of d contiguous values are $t \in \{11, \ldots, 17\}$.

Because of the presence of missing values, it is not clear how to apply (2.1) when any of the data happen to be missing. We proceed to describe how the data can be differenced in general, and how the Gaussian divergence (i.e., -2 times the log Gaussian likelihood) is computed. Let $\{1, 2, \ldots, T\}$ denote the indices of the full sample $[x_1, x_2, \ldots, x_T]'$, which just means that we let $t = 1$ denote the index of the first observed value and $t = T$ is the index of the last observed value; in practice, one must have an accounting of which values are missing. (T is not the number of available observations, but the length the sample would be if all missing values were supplied.) Let $\in_t$ for each $1 \leq t \leq T$ denote the components of x_t that are observed: this is some subset of $\{1, 2, \ldots, N\}$, and equals $\emptyset$ if the observation is completely missing. Further, let $\in_{1:t}$ denote a list object with t elements such that for $1 \leq s \leq t$ the sth element is $\in_s$. This encodes all the observed values up to time t. For any $1 \leq t \leq T$, let $X_t = [x_1', x_2', \ldots, x_t']'$ (whether or not the values are actually observed) so that the full (potential) sample is X_T.

Let $t_\star$ be defined as one less than the index of the smallest value of one of the batches of d contiguous values (there can be no missing values in any of the components of these contiguous values). Without loss of generality, consider $t_\star$ to be the smallest possible, so that if there are multiple batches of d contiguous values, we take the batch that lies earliest in the time series. (So in the above example, $t_\star = 10$.) Clearly, $\in_s = \{1, 2, \ldots, N\}$ for $t_\star + 1 \leq s \leq t_\star + d$, and $0 \leq t_\star \leq T - d$. In mathematical notation, $t_\star = \min T_\star$, where

$$T_\star = \{t : 0 \leq t \leq T - d,\ \in_s = \{1, \ldots, N\}\, \forall\, t + 1 \leq s \leq t + d\}. \tag{2.2}$$

Using this definition we can then partition the vector X_T as

$$\begin{aligned} X_T' &= [X_\flat', X_\star', X_\sharp'] \\ X_\star' &= [x_{t_\star+1}', \ldots, x_{t_\star+d}'] \end{aligned}$$

$$X_\flat' = [x_1', \ldots, x_{t_\star}']$$
$$X_\sharp' = [x_{t_\star+d+1}', \ldots, x_T'],$$

where $X_\flat$ and $X_\sharp$ are, respectively, empty vectors if $t_\star = 0$ or $t_\star = T - d$. Unpacking (2.1), we have forward and aftward differencing defined via

$$\underline{x}_t = \sum_{k=0}^{d} \delta_k \, x_{t-k} \quad t_\star + d + 1 \leq t \leq T \tag{2.3}$$

$$\underline{x}_{t+d} = \sum_{k=0}^{d} \delta_k \, x_{t+d-k} \quad 1 \leq t \leq t_\star. \tag{2.4}$$

(Again, set these to be respectively an empty vector if $t_\star = 0$ or $t_\star = T - d$.) Taking (2.3) and (2.4) together, it is clear we can compute $\underline{x}_t$ for $d + 1 \leq t \leq T$ from X_T, assuming this were available. In particular, $\underline{X}_T = [\underline{x}_{d+1}', \ldots, \underline{x}_T']' = [\underline{X}_\flat', \underline{X}_\sharp']'$, where $\underline{X}_\flat$ and $\underline{X}_\sharp$ are defined via $\underline{X}_\flat' = [\underline{x}_{d+1}', \ldots, \underline{x}_{d+t_\star}']$ and $\underline{X}_\sharp' = [\underline{x}_{d+t_\star+1}', \ldots, \underline{x}_T']$. Moreover, if we have the initial values $X_\star$ together with $\underline{X}_T$, then we can recover the full sample X_T, as the following result demonstrates.

Proposition 2.1. *There exists an invertible linear transformation of X_T to $X_\star$ and $\underline{X}_T$, given by the block matrix $\widetilde{\Delta}$ defined via*

$$\begin{bmatrix} \delta_d & \delta_{d-1} & \ldots & \delta_0 & 0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 \\ 0 & \delta_d & \delta_{d-1} & \ldots & \delta_0 & 0 & \ldots & \ldots & \ldots & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & \delta_d & \delta_{d-1} & \ldots & \delta_0 & 0 & \ldots & \ldots & 0 \\ 0 & \ldots & 0 & 0 & I_N & 0 & \ldots & 0 & \ldots & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & \ldots & \ldots & \ldots & 0 & I_N & 0 & \ldots & \ldots & 0 \\ 0 & \ldots & \ldots & 0 & \delta_d & \delta_{d-1} & \ldots & \delta_0 & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & \ldots & 0 & \delta_d & \delta_{d-1} & \ldots & \delta_0 & 0 & \ldots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 & \delta_d & \delta_{d-1} & \ldots & \delta_0 \end{bmatrix}. \tag{2.5}$$

This matrix has $N \times N$ blocks, with the first $t_\star$ and last $T - d - t_\star$ block rows consisting of the elements of $\delta(z)$, with the middle d block rows consisting of I_N matrices.

The definition of $\widetilde{\Delta}$ in (2.5) means that the matrix has structure

$$\widetilde{\Delta} = \left[\begin{array}{ccc} \Delta_\flat & & 0 \\ 0 & I_{dN} & 0 \\ 0 & & \Delta_\sharp \end{array} \right],$$

where the two Δ matrices have jkth block entry given by δ_{j+d-k} (here, we set $\delta_\ell = 0$ if $\ell < 0$ or $\ell > d$); $\Delta_\flat$ has $t_\star$ block rows and $t_\star + d$ block columns, and $\Delta_\sharp$ has $T - d - t_\star$ block rows and $T - t_\star$ block columns.

As a repercussion of Proposition 2.1, X_T can always be expressed as a linear combination of initial values $X_\star$ together with the differenced data $\underline{X}_\flat$ and $\underline{X}_\sharp$; the proof of Proposition 2.1 shows that these linear combinations involve square matrices $C_\flat$ and $C_\sharp$, defined as follows. Let $\delta(z)^{-1} = \sum_{\ell \geq 0} \tau_\ell z^\ell$ be the matrix power series of the inverse of $\delta(z)$, which converges for all $|z| < r$ for some $r < 1$; likewise let $\widetilde{\delta}(z)^{-1} = \sum_{\ell \geq 0} \upsilon_\ell z^\ell$ be the matrix power series of the inverse of $\widetilde{\delta}(z) = \sum_{j=0}^{d} \delta_{d-j} z^j$, the reverse polynomial of $\delta(z)$. Then $C_\sharp$ has $T - t_\star - d$ blocks, with jkth block equal to τ_{j-k} ($\tau_\ell = 0$ for $\ell < 0$), and $C_\flat$ has $t_\star$ blocks with jkth block equal to υ_{k-j} ($\upsilon_\ell = 0$ for $\ell < 0$).

It is common in the literature to assume that the initial values are uncorrelated with $\{\underline{x}_t\}$; here the so-called initial values $X_\star$ need not pertain to the beginning of the sample, but yet are used to generate the stochastic process, and hence are initial in that sense. Here we extend *Assumption A* of [2] (also see [3], [20], [24], [27]) as follows:

Assumption A: $\{x_{t_\star+1}, \ldots, x_{t_\star+d}\}$ are uncorrelated with $\{\underline{x}_t\}$.

This assumption can also be strengthened when latent processes are present. As shown below, Assumption A is sufficient to guarantee that the log Gaussian likelihood factorizes into two portions, only one of which depends upon f_x; this is the justification for optimizing a likelihood purely on the basis of the differenced sample. (While this is well-known [3] for complete observations, our result is an extension to the multivariate case of ragged edge data.)

To proceed, let J be a selection matrix defined by deleting any rows from I_{TN} that correspond to missing observations. Specifically, retain row $(s-1)N + k$ for $1 \leq s \leq T$ and $1 \leq k \leq N$ if k is an element of $\in_s$. Because d contiguous values are observed, J has the structure

$$J = \begin{bmatrix} J_\flat & 0 & 0 \\ 0 & I_{dN} & 0 \\ 0 & 0 & J_\sharp \end{bmatrix}, \qquad (2.6)$$

where the blocks $J_\flat$ and $J_\sharp$ just consist of ones and zeroes. So JX_T is our observed sample, with all the missing values excised and the vector written compactly; we call this a "ragged edge sample." If we can re-express this random vector such that the initial values and the differenced data are featured in separate blocks, then the Gaussian divergence will factor and model evaluation on the basis solely of f_x will be possible. Below, let $V[z]$ denote the variance matrix of a random vector z.

Theorem 2.1. *Let $\{x_t\}$ be a difference-stationary process with degree d matrix polynomial $\delta(z)$, such that δ_0 and δ_d are invertible. Let JX_T be the ragged edge sample from $\{x_t\}$ defined via J given in (2.6), where $t_\star = \min T_\star$ is given by (2.2). If Assumption A holds, $V[X_\star]$ is invertible, and $f_x(\lambda)$ is non-singular for all $\lambda \in [-\pi, \pi]$, then the quadratic form in the Gaussian divergence for the ragged edge sample factors as*

$$(JX_T)' \, V[JX_T]^{-1} \, (J \, X_T)$$

$$= X_\star' \, V[X_\star]^{-1} \, X_\star + [\Delta_J \underline{X}_T]' \left(\Delta_J \, \Gamma_{T-d} \, \Delta_J{}' \right)^{-1} [\Delta_J \underline{X}_T],$$

where Γ_{T-d} *is the block Toeplitz covariance matrix of the differenced data vector* $\underline{X}_T$ *and*

$$\Delta_J = \left[\begin{array}{cc} J_\flat \, C_\flat & 0 \\ 0 & J_\sharp \, C_\sharp \end{array} \right].$$

Also,

$$\log \det V \left(J X_T \right) = \log \det V[X_\star] + \log \det \Delta_J \, \Gamma_{T-d} \, \Delta_J{}'.$$

So long as $V[X_\star]$ shares no parameters with the model for f_x, it follows from Theorem 2.1 that the gradient of the Gaussian divergence equals

$$\nabla \left([\Delta_J \underline{X}_T]' \left(\Delta_J \, \Gamma_{T-d} \, \Delta_J{}' \right)^{-1} [\Delta_J \underline{X}_T] + \log \det \Delta_J \, \Gamma_{T-d} \, \Delta_J{}' \right),$$

and we can compute maximum likelihood estimates with only knowing the parametric form of f_x.

2.2. Filtering

The filtering problem involves computing $y_t = \Psi(B) \, x_t$ for some given filter $\Psi(B) = \sum_{j \in \mathbb{Z}} \psi_j \, B^j$, where each coefficient ψ_j is an $N \times N$ dimensional matrix. A filter can involve infinitely many nonzero coefficients, and hence y_t may not be computable for all $1 \leq t \leq T$, given that it may depend on various x_{t-j} that are not observed. The optimal estimate $\widehat{y}_t$ of y_t given the sample amounts to applying the filter to an extended series $\{\widetilde{x}_t\}$, where $\widetilde{x}_t$ either equals x_t if $\in_t = \{1, 2, \ldots, N\}$ (i.e., it is fully observed) or is a forecast, aftcast, or midcast otherwise. (Typically the filter will be truncated, because $\Psi(B)$ involves infinitely many nonzero coefficients.) For notation, let the minimal MSE linear projection of a random vector z_r onto the linear span of random variables x_s with $1 \leq s \leq t$ such that components $\in_s$ are observed be denoted by $P_{\in_{1:t}}[z_r] = \widehat{z}_{r|\in_{1:t}}$; see [26] for background on linear projections. The MSE of such an estimate is denoted $V_{\in_{1:t}}[z_r]$.

The filter $\Psi(B)$ can be *ad hoc* or can be connected to the dynamics of the process. Although these two cases tend to pertain unto disjoint camps of practitioners, here we provide a cohesive treatment. References on nonparametric filtering include [7], [6], [28], and [8]. The WK approach to filtering begins by stipulating signal and noise processes $\{s_t\}$ and $\{n_t\}$ such that $x_t = s_t + n_t$. One seeks a WK filter $\Psi(B)$ such that the mean squared error (MSE) of estimating s_t via $\Psi(B) x_t$ is minimized. The solution is given in [2], and extended to the multivariate case in [27]: essentially, $\Psi(e^{-i\lambda})$ is given by a ratio of pseudo-spectra [4] for signal to data process. Hence, if a model with latent components has been fitted the spectral densities for signal and noise can be calculated, and the coefficients ψ_j computed.

The WK estimator of s_t is $P_M[s_t]$, where M is the closed linear span of the entire time series $\{x_t\}$; for short we denote this by $\widehat{s}_{t|\infty}$. As mentioned above, this estimator is not computable unless we extend the sample. By the linearity of projections, an estimator based on the sample is identical to projecting $\widehat{s}_{t|\infty}$ on the available sample, which we denote by $\widehat{s}_{t|\in 1:T}$. Hence we have

$$\widehat{s}_{t|\in 1:T} = P_{\in 1:T}[\widehat{s}_{t|\infty}] = \sum_{j \in \mathbb{Z}} \psi_j \, P_{\in 1:T}[x_{t-j}] = \sum_{j \in \mathbb{Z}} \psi_j \, \widetilde{x}_{t-j}, \tag{2.7}$$

where $\widetilde{x}_t$ is short for $P_{\in 1:T}[x_t]$. This calculation shows that such an estimator can be computed by filtering the extended series. The error in such an estimator is determined as follows:

$$s_t - \widehat{s}_{t|\in 1:T} = \left(s_t - \widehat{s}_{t|\infty}\right) + \sum_{j \in \mathbb{Z}} \psi_j \, \left(x_{t-j} - \widetilde{x}_{t-j}\right). \tag{2.8}$$

The error therefore consists of two summands, which are uncorrelated; the optimality of the WK estimator indicates that the signal extraction error is orthogonal to $\{x_t\}$ under the extension of Assumption A [2], and the second term is a linear combination of elements of $\{x_t\}$. Therefore the signal extraction MSE breaks into two portions: the first is the WK MSE $V[s_t - \widehat{s}_{t|\infty}]$, and the second is the so-called casting MSE. Whereas the former term does not depend on t and can be computed entirely from a knowledge of the signal and noise pseudo-spectra, the latter term depends on t and is a function of the filter coefficients together with the casting error covariances, where the casting errors are $x_{t-j} - \widetilde{x}_{t-j}$ for various j.

Another approach to the entire problem is based upon a direct calculation of $\widehat{s}_{t|\in 1:T}$, namely by determining the covariance of s_t with all the observed portions of X_T, and normalizing by the inverse of the covariance matrix. Explicit formulas – in the case that there are no missing values – are available for the estimator and its time-varying MSE [27], but are computationally unwieldy for large T (greater than 500). Whereas these exact matrix formulas are preferable when they are practicable, we must have recourse to another approach such as (2.7) when T is large or missing values are present; see [12] for a related discussion. One challenge is the computation of the extensions (this is handled through recursive multi-step ahead casting), and the other challenge is determining casting error covariances; these issues are addressed through the recursive algorithms discussed in Sections 3.1 through 3.4 below. (These recursive algorithms are an alternative to the state space approach discussed in [13].) Calculation of the WK MSE involves frequency domain computations.

The case of an *ad hoc* filter is only slightly simpler than the WK case. Now the perspective does not involve latent components, but instead views the filter output $y_t = \widehat{s}_{t|\infty}$ as the target quantity. For example, $\Psi(B)$ could be chosen such that its gain function (the modulus of the Fourier transform of the coefficient sequence) is an indicator function over the low frequencies, known as an ideal low-pass filter [1], [5]. Then the filter output $\{y_t\}$ corresponds to the low-frequency components of the data process. In such a case there is only one component to the error decomposition (2.8), namely the second portion corresponding to extending the sample. So the computational challenges are similar to the case of a WK filter.

3. Algorithms

3.1. Predictor recursions

Here we discuss recursions for multi-step ahead forecasting of non-stationary processes, extending the algorithms given in [23] for the stationary case. In this subsection we suppose that there are no missing values, so that $t_\star = 0$; in Section 3.3 below we allow for missing values by building upon the recursions described here. Our goal is to recursively compute $\widehat{x}_{t+j|\in_{1:t}}$ for some $j \geq 1$. For the remainder of this subsection we assume that $t \geq d+1$, because $x_1, \ldots, x_d$ are observed and therefore need not be forecasted.

The j-step ahead predictor is a linear combination of the random variables $x_1, \ldots, x_t$, and hence can be written as $\widehat{x}_{t+j|\in_{1:t}} = \ell_{t+1}(j)' \, X_t$ for a $tN \times N$ matrix $\ell_{t+1}(j)$, which we call the prediction filter (even though it depends on only a finite past). The notation for the index of the prediction filter refers to the time index for which a prediction is needed, namely time $t+1$.

If we are computing predictors of the stationary process $\{\underline{x}_t\}$ (from observations of such), we use an underscore: $P_{\underline{X}_t}[x_{t+j}] = \underline{\ell}_{t+1-d}(j)' \, \underline{X}_t$ where $\underline{\ell}_{t+1-d}(j)$ is a $(t+1-d)N \times N$ dimensional. Here we use $P_{\underline{X}_t}$ to mean the minimal MSE linear projection onto the linear span of the random vector $\underline{X}_t$. (Similarly, $V_{\underline{X}_t}$ denotes the MSE of such a projection.) In the stationary case the subscript is related to the size of the matrix, which has $t-d$ blocks of dimension N. The quantity $\underline{\ell}_{t+1-d}(1)$ is actually featured in the Durbin-Levinson algorithm for the Gaussian likelihood, as it furnishes the one-step ahead prediction from a sample of size $t-d$; cf. discussion in [23]. We mention this to highlight that $\underline{\ell}_{t+1-d}(1)$ can be quickly computed in a recursive fashion: let $\underline{u}_{t+1-d}(1)$ be the one-step behind prediction filter, so that $\widehat{\underline{x}}_{d|\in_{(d+1):t}} = \underline{u}_{t+1-d}(1)' \, \underline{X}_t$. Then the formulas for the one-step stationary prediction filters (ahead and behind) are

$$\begin{aligned} \underline{\ell}_{t+1-d}(1)' &= [\gamma(t-d), \ldots, \gamma(1)] \, \Gamma_{t-d}^{-1} \\ \underline{u}_{t+1-d}(1)' &= [\gamma(-1), \ldots, \gamma(-t+d)] \, \Gamma_{t-d}^{-1}. \end{aligned}$$

Then by iterating over values of $d \leq t+1 \leq T$, the prediction filters can be recursively computed:

$$\underline{\ell}_{t+1-d}(1)' = [\xi_{t-d} \, m_{t-d}^{-1}, \, \underline{\ell}_{t-d}(1)' - \xi_{t-d} \, m_{t-d}^{-1} \, \underline{u}_{t-d}(1)'] \tag{3.1}$$

$$\underline{u}_{t+1-d}(1)' = [\underline{u}_{t-d}(1)' - \xi_{t-d}' \, n_{t-d}^{-1} \, \underline{\ell}_{t-d}(1)', \, \xi_{t-d}' \, n_{t-d}^{-1}] \tag{3.2}$$

$$m_{t+1-d} = \gamma(0) - [\gamma(-1), \ldots, \gamma(-t+d)] \, \underline{u}_{t+1-d}(1) \tag{3.3}$$

$$n_{t+1-d} = \gamma(0) - [\gamma(t-d), \ldots, \gamma(1)] \, \underline{\ell}_{t+1-d}(1) \tag{3.4}$$

$$\xi_{t+1-d} = \gamma(t+1-d) - [\gamma(t-d), \ldots, \gamma(1)] \, \underline{u}_{t+1-d}(1). \tag{3.5}$$

Here we note that m_{t+1-d} and n_{t+1-d} are the one-step (behind and ahead, respectively) stationary prediction variances for time $t+1$. Also, ξ_{t-d} equals the lag $t-d$ partial autocorrelation of the process $\{\underline{x}_t\}$.

Combining the recursions for the stationary case with the action of $\delta(z)$ given in (2.1) provides new recursions for the non-stationary prediction filter. Let Δ_{t-d} have the same structure as $\Delta_\sharp$ and $\Delta_\flat$ of Section 2.1, but with $(t-d)N$ rows and tN columns.

Theorem 3.1. *The one-step ahead prediction filter for the difference-stationary process $\{x_t\}$ satisfying (2.1) can be expressed in terms of the stationary prediction filter:*

$$\ell_{t+1}(1)' = -\delta_0^{-1}\,[0,\ldots,0,\delta_d,\ldots,\delta_1] + \delta_0^{-1}\,\underline{\ell}_{t+1-d}(1)'\,\Delta_{t-d}, \tag{3.6}$$

where the indicated zeroes in (3.6) are $N \times N$ dimensional matrices of zeroes. Letting $b'_{t+1-d} = \underline{\ell}_{t+1-d}(1)'\,\Delta_{t-d}$ and $a'_{t+1-d} = \underline{u}_{t+1-d}(1)'\,\Delta_{t-d}$, the one-step ahead prediction filter can be recursively computed via

$$\begin{aligned} b'_{t+1-d} &= \xi_{t-d}\, m_{t-d}^{-1}\,[\delta_d,\ldots,\delta_0,0,\ldots,0] + [0,\, b'_{t-d} - \xi_{t-d}\, m_{t-d}^{-1}\, a'_{t-d}] \\ a'_{t+1-d} &= \xi'_{t-d}\, n_{t-d}^{-1}\,[0,\ldots,0,\delta_d,\ldots,\delta_0] + [a'_{t-d} - \xi'_{t-d}\, n_{t-d}^{-1}\, b'_{t-d}, 0], \end{aligned}$$

together with the stationary recursions (3.3), (3.4), and (3.5). The prediction error variance for the one-step ahead predictor is

$$V_{X_t}[x_{t+1}] = \delta_0^{-1}\, n_{t+1-d}\, \delta_0^{-1\prime}. \tag{3.7}$$

Next, we derive recursions for j-step ahead prediction filters by expressing them in terms of lower lead predictors.

Corollary 3.1. *The j-step ahead prediction filter for the difference-stationary process $\{x_t\}$ satisfying (2.1) can be expressed recursively in forecast lead:*

$$\ell_{t+1}(j)' = -\sum_{k=1}^{d} \delta_0^{-1}\,\delta_k\,\ell_{t+1}(j-k)' + \delta_0^{-1}\,\underline{\ell}_{t+1-d}(j)', \tag{3.8}$$

where $\ell_{t+1}(j)$ for $j \leq 0$ is extended to mean a $tN \times N$-dimensional matrix that is zero except for an identity matrix I_N in the $(1-j)$th block row. This prediction filter depends on the stationary j-step ahead predictor, which is recursively computed via

$$\underline{\ell}_{t+1-d}(j) = [I_{(t-d)N},\, \underline{\ell}_{t+1-d}(1),\, \ldots\, \underline{\ell}_{t+1-d}(j-1)]\; \underline{\ell}_{t+j-d}(1). \tag{3.9}$$

As a result of Corollary 3.1, we can run the Durbin-Levinson algorithm, i.e., (3.1) through (3.5), to obtain $\underline{\ell}_{t+j-d}(1)$ for j so large as needed to compute all multi-step ahead prediction filters via (3.9).

Remark 1. Because the partial autocorrelations of stationary processes commonly decay swiftly to zero, it is possible to render (3.9) even more efficient. Observe that the partial autocorrelations ξ_t can become negligible for large t (and identically zero when $t > p$ for a VAR(p) process). If this occurs, there is no update to the one-step ahead predictor, although the block vector gets pre-pended with a zero matrix. Hence, the situation can arise whereby there exists t_0 such that $\underline{\ell}_{t+1-d}(1)' = [0, \ldots, 0, \underline{\ell}_{t_0+1}(1)']$ for all $t \geq t_0 + d$. Here, the first $t - t_0 - d$ block rows of $\underline{\ell}_{t+1-d}(1)$ are zero; the same is true of $\underline{\ell}_{t+1-d}(j)$, which is proved by induction and use of (3.9). As a consequence, we only need to store the lower right $t_0 N \times t_0 N$-dimensional submatrix of $[I_{t-d} \otimes I_N,\ \underline{\ell}_{t+1-d}(1), \ldots \underline{\ell}_{t+1-d}(j-1)]$, which can yield substantial memory savings. This has been implemented in *Ecce Signum*, whereby if the partial autocorrelations are less than a given threshold at lag t_0, then subsequent partial autocorrelations are assumed to be zero.

3.2. Time reversal

Intuitively, we should be able to aftcast by time-reversing the sample and applying forecasting. However, this presumes a time-reversible structure in the data process, which need not be true. Set $y_t = x_{-t}$ for all $t \in \mathbb{Z}$, so that $\{y_t\}$ is the time-reversed data process. Define $\widetilde{\delta}(z) = \sum_{k=0}^{d} \delta_{d-k} z^k$ to be the reverse order of the polynomial $\delta(z)$, so that

$$\widetilde{\delta}(B) y_t = \sum_{k=0}^{d} \delta_{d-k}\, y_{t-k} = \sum_{j=0}^{d} \delta_j\, x_{d-t-j} = \delta(B)\, x_{d-t} = \underline{x}_{d-t}.$$

We call this difference-stationary time series $\{\underline{y}_t\}$: its acvf is $\mathrm{Cov}[\underline{y}_{t+h}, \underline{y}_t] = \mathrm{Cov}[\underline{x}_{d-t-h}, \underline{x}_{d-t}] = \gamma(-h) = \gamma(h)'$. It follows that the spectral density of the differenced time-reversed process is f', which equals the conjugate $\overline{f}$. Applying this to aftcasting, we suppose that for some $j \geq 1$ it is desired to compute $P_{Y_{t+1}}[x_{t+1-j}]$, where $Y_{t+1} = [x_T', \ldots, x_{t+2}', x_{t+1}']'$. We begin with the difference-stationary case, and use the notation $P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}]$ for j-step behind prediction based on the sample $\underline{Y}_{T-t-d}' = [\underline{x}_T', \ldots, \underline{x}_{t+d+1}']$. Note that the index of $\underline{Y}_{T-t-d}$ denotes the number of random variables included, like $\underline{X}_t$, but the variables are ordered from future to past. The prediction can be explicitly computed via

$$\begin{aligned} P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}] &= \mathrm{Cov}[\underline{x}_{t+d+1-j}, \underline{Y}_{T-t-d}]\, V[\underline{Y}_{T-t-d}]^{-1}\, \underline{Y}_{T-t-d} \\ &= [\gamma(t+d+1-j-T) \ldots \gamma(-j)]\, \Gamma_{T-t-d}^{-1}\, \underline{Y}_{T-t-d} \\ &= [\gamma(-j) \ldots \gamma(t+d+1-j-T)]\, \Gamma_{T-t-d}^{-1}\, \Pi\, \underline{Y}_{T-t-d} \end{aligned}$$

where Π is a block permutation that reverses the time sequence of a block vector. In the case $j = 1$ the last expression equals the transpose of the behind prediction filter $\underline{u}_{T-t-d+1}(1)$ applied to $\Pi\, \underline{Y}_{T-t-d}$, according to the discussion in the previous subsection. Extending this definition to $j \geq 1$, we let $\underline{u}_{T-t-d+1}(j)$ denote the j-step behind prediction filter, given by $\underline{u}_{T-t-d+1}(j)'\, \Pi\, \underline{Y}_{T-t-d} = P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}]$.

The above calculations show that $\underline{u}_{T-t-d+1}(j)$ equals the j-step ahead prediction filter $\underline{\ell}_{T-t-d+1}(j)$ where the autocovariances are computed from f'. (In the case that $f' = f$, it holds that $\underline{u}_{T-t-d}(j)'\, \Pi = \underline{\ell}_{T-t-d}(j)'$.) Hence

$P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}]$ equals a linear combination of the time reversed differenced-sample $\underline{Y}_{T-t-d}$; analogously, we define $P_{Y_{t+1}}[x_{t+1-j}]$ in terms of the time reversed sample Y_{t+1} via $u_{t+1-j}(j)' \,\Pi\, Y_{t+1}$. Here the subscript on $u_{t+1-j}(j)$ denotes the time index for which the aftcast is generated. The one-step behind prediction filters $\underline{u}_{T-t-d+1}(1)$ are already computed in the Durbin-Levinson recursions (3.1) through (3.5), and the following result shows how the nonstationary one-step behind prediction filters $u_t(1)$ are obtained.

Theorem 3.2. *The one-step behind prediction filter for the difference-stationary process $\{x_t\}$ satisfying (2.1) can be expressed in terms of the stationary prediction filter:*

$$u_t(1)' = -\delta_d^{-1}\,[\delta_{d-1}, \ldots, \delta_0, 0, \ldots, 0] + \delta_d^{-1}\,\underline{u}_{T-t-d+1}(1)'\,\Delta_{T-t-d}, \tag{3.10}$$

where the indicated zeroes in (3.10) are $N \times N$ dimensional matrices of zeroes. Letting $b'_{T-t-d+1} = \underline{u}_{T-t-d+1}(1)'\,\Delta_{T-t-d}$ and $a'_{T-t-d+1} = \underline{\ell}_{T-t-d+1}(1)'\,\Delta_{T-t-d}$, the one-step behind prediction filter can be recursively computed via

$$\begin{aligned} b'_{T-t-d+1} &= \xi_{T-t-d}\, m^{-1}_{T-t-d}\,[0, \ldots, 0, \delta_d, \ldots, \delta_0] \\ &\quad + [b'_{T-t-d} - \xi_{T-t-d}\, m^{-1}_{T-t-d}\, a'_{T-t-d}, 0] \\ a'_{T-t-d+1} &= \xi'_{T-t-d}\, n^{-1}_{T-t-d}\,[\delta_d, \ldots, \delta_0, 0, \ldots, 0] \\ &\quad + [0, a'_{T-t-d} - \xi'_{T-t-d}\, n^{-1}_{T-t-d}\, b'_{T-t-d}], \end{aligned}$$

where all quantities m_{T-t-d}, n_{T-t-d}, and ξ_{T-t-d} are recursively computed from transposed autocovariances, corresponding to f', in (3.3), (3.4), and (3.5). The prediction error variance for the one-step behind predictor is

$$V_{\Pi Y_{t+1}}[x_t] = \delta_d^{-1}\, n_{T-t-d+1}\, \delta_d^{-1\prime}. \tag{3.11}$$

Similarly, the j-step behind prediction filters can be expressed in terms of lower lead prediction filters.

Corollary 3.2. *The j-step behind prediction filter for the difference-stationary process $\{x_t\}$ satisfying (2.1) can be expressed recursively in aftcast lead:*

$$u_{t+1-j}(j)' = -\sum_{k=1}^{d} \delta_d^{-1}\, \delta_{d-k}\, u_{t+1-j+k}(j-k)' + \delta_d^{-1}\, \underline{u}_{T-t-d+1}(j)', \tag{3.12}$$

where $u_{t+1-j}(j)$ for $j \leq 0$ is extended to mean a $tN \times N$-dimensional matrix that is zero except for an identity matrix I_N in the $(1-j)$th block row. This prediction filter depends on the stationary j-step behind predictor, which is recursively computed (letting $s_t = \Pi'\,\underline{u}_t$) via

$$s_{T-t-d+1}(j) = [I_{(t-d)N},\, s_{T-t-d+1}(1),\, \ldots\, s_{T-t-d+1}(j-1)]\, s_{T-t-d+j}(1). \tag{3.13}$$

3.3. General casting algorithm

If one only needs forecasts and aftcasts, then the preceding methods are sufficient and efficient. If one needs midcasts, then necessarily covariances of prediction errors must be computed and techniques such as (3.9) will no longer suffice. We discuss herein a novel recursive algorithm that computes all casts and their error covariances in a recursive fashion. Referring to the definition of $t_\star$ in Section 2, we begin with a forward sweep whenever $t_\star < T - d$, which is followed by a backward sweep unless $t_\star = 0$, i.e., in the case that the first d values of the sample are contiguous, only a forward sweep is needed. The basic idea is to proceed through the sample and at each time point make updates to casts via utilizing the new information, beginning at time $t_\star + d + 1$ and proceeding to time T and making forecasts. Then a backward sweep produces aftcasts for times $t_\star$ down to time 1. (Note that no casts are needed for times $t_\star + 1 \leq t \leq t_\star + d$, as these are fully observed.)

3.3.1. The forward sweep

In the forward sweep we consider only times $t_\star + d + 1 \leq t \leq T$ by ignoring any portion of the ragged sample that corresponds to times $1, \ldots, t_\star$. Recall that X_t consists of all variables between times 1 and t; hence, by ignoring times $1, \ldots, t_\star$ we see that X_t consists of all variables (whether observed or not) in the range $t_\star + 1$ through t. We can visualize this as concatenating the large vectors $X_\star$ and $X_\sharp$, and retaining only those components up through index t.

For this subsection only we set $t_\star = 0$ in various notations introduced earlier. Then $P_{\in_{1:t}}[X_t]$ denotes the projection of X_t onto the set of variables with observed indices, and $V_{\in_{1:t}}[X_t]$ is the corresponding prediction error variance. When midcasting, many of these quantities are trivial: if a particular element x_s of X_t is fully observed (so that $\in_s = \{1, \ldots, N\}$), then the projection equals x_s itself (and the prediction error variance is zero), so we do not need to store anything. Our goal is to compute $P_{\in_{1:t}}[X_t]$ and $V_{\in_{1:t}}[X_t]$ in terms of $P_{\in_{1:t-1}}[X_{t-1}]$ and $V_{\in_{1:t-1}}[X_{t-1}]$. The result below describes the efficient computation of such quantities.

For each $d + 1 \leq t \leq T$ define $x_t^{\in}$ to be the subvector of x_t corresponding to components that are observed. (Note that $x_t^{\in}$ is the empty set when $\in_t = \emptyset$ and the vector is entirely missing.) Mathematically, $x_t^{\in} = I_N[\in_t,]\, x_t$, where $I_N[\in_t,]$ indicates taking only those rows of I_N corresponding to the indices in $\in_t$. Likewise, defining $\emptyset_t = \{1, \ldots, N\} \setminus \in_t$, let $x_t^{\emptyset} = I_N[\emptyset_t,] x_t$ consist of those random variables that are missing; this will be only a target quantity, as it is never observed. Note that there exists a row permutation of $I_N[\in_t,]$ and $I_N[\emptyset_t,]$ such that their stacking equals I_N. The recursive algorithm for updating projections, given below, is analogous to the skipping approach [16] for state space systems, but here is applied to general difference-stationary time series.

Theorem 3.3. *Let $\{x_t\}$ be a difference-stationary process with degree d matrix polynomial $\delta(z)$, such that δ_0 and δ_d are invertible. Let JX_T be the ragged edge sample from $\{x_t\}$ defined via J given in (2.6), where $t_\star = \min T_\star$ is given by (2.2). Assume that $t_\star = 0$ and Assumption A holds. For $d+1 \leq t \leq T$, the one-step ahead forecasts and prediction variances can be computed from previously computed quantities ($P_{\in_{1:t-1}}[X_{t-1}]$ and $V_{\in_{1:t-1}}[X_{t-1}]$) via*

$$P_{\in_{1:t-1}}[x_t] = \ell_t(1)' \, P_{\in_{1:t-1}}[X_{t-1}] \tag{3.14}$$

$$V_{\in_{1:t-1}}[x_t] = V_{X_{t-1}}[x_t] + \ell_t(1)' \, V_{\in_{1:t-1}}[X_{t-1}] \, \ell_t(1), \tag{3.15}$$

where $\ell_t(1)$ and $V_{X_{t-1}}[x_t]$ are computed via (3.6) and (3.7). Letting $W_{t-1} = I_N[\in_t,]' \left(I_N[\in_t,] \, V_{\in_{1:t-1}}[x_t] \, I_N[\in_t,]' \right)^{-1}$, we can update the information via

$$\begin{aligned} P_{\in_{1:t}}[X_t] &= P_{\in_{1:t-1}}[X_t] \\ &\quad + \left[\begin{array}{c} V_{\in_{1:t-1}}[X_{t-1}] \, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right] W_{t-1} \, \left(x_t^{\in} - I_N[\in_t,] \, P_{\in_{1:t-1}}[x_t] \right) \end{aligned} \tag{3.16}$$

$$\begin{aligned} V_{\in_{1:t}}[X_t] &= \left[\begin{array}{cc} V_{\in_{1:t-1}}[X_{t-1}] & V_{\in_{1:t-1}}[X_{t-1}] \, \ell_t(1) \\ \ell_t(1)' \, V_{\in_{1:t-1}}[X_{t-1}] & V_{\in_{1:t-1}}[x_t] \end{array} \right] \\ &\quad - \left[\begin{array}{c} V_{\in_{1:t-1}}[X_t] \, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right] W_{t-1} \, I_N[\in_t,] \left[\begin{array}{c} V_{\in_{1:t-1}}[X_t] \, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right]'. \end{aligned} \tag{3.17}$$

Remark 2. Theorem 3.3 shows how to recursively update estimates and variances. Note that formula (3.17) nicely illustrates the impact of additional information at time t, and how it can lower the prediction error variance. These derivations show that the updates to past midcasts require a knowledge of prediction error covariances, and it is not sufficient to know just the midcast predictors – this is different from the case of forecasting and aftcasting, as seen in Corollaries 3.1 and 3.2. However, the number of elements requiring storage in an implementation need not be onerous: prior casts are a subset of $P_{\in_{1:t-1}}[X_{t-1}]$ corresponding to unobserved elements of X_{t-1}, and only the covariances of these element's prediction errors need be stored, as all other covariances will be zero.

Another consequence of Theorem 3.3 is that we can easily compute the factored divergence of Theorem 2.1. With $X_t^{\in}$ denoting the ragged sample portion of X_t, observe that $X_t^{\in\prime} \, V[X_t^{\in}]^{-1} \, X_t^{\in}$ equals

$$X_{t-1}^{\in}{}' \, V[X_{t-1}^{\in}]^{-1} \, X_{t-1}^{\in} + \left(x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}] \right)' V_{\in_{1:t-1}}[x_t^{\in}]^{-1} \, \left(x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}] \right),$$

which suggests defining the time series residual as

$$\epsilon_t = V_{\in_{1:t-1}}[x_t^{\in}]^{-1/2} \, \left(x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}] \right). \tag{3.18}$$

(When an observation is completely missing, the corresponding residual cannot be computed.) Then

$$X_T^{\in\prime} \, V[X_T^{\in}]^{-1} \, X_T^{\in} = \sum_t \epsilon_t' \, \epsilon_t, \tag{3.19}$$

where the sum is over all $d+1 \leq t \leq T$ such that x_t is at least partially observed. These residual calculations are automatically computed during the forward sweep.

3.3.2. The backward sweep

We now describe the backward sweep, for completeness; it is useful to have these results from the standpoint of implementation, since thinking in reverse time requires more care. Supposing the forward sweep is completed and $t_\star > 0$ (if $t_\star = 0$, no backwards sweep is needed); we now take $1 \leq t \leq t_\star$ and set $\in_{t:T}$ to denote observed indices in the set $\{t, t+1, \ldots, T\}$. (These indices are nested as t increases, which is the reverse of the forward sweep case, where as t increased we obtained weak supersets.) Recall that Y_t consists of all variables in this range (but written in reverse time, starting with x_T at the top and finishing with x_t at the bottom of the vector), whether observed or not; so $\Pi\, Y_t$ denotes a portion of the concatenation of $X_\flat$ with $X_\star$ and $X_\sharp$, with the first $t-1$ elements omitted.

Note that at the end of the forward sweep, we have calculated $P_{\in_{t_\star+1:T}}[\Pi Y_{t_\star+1}]$ and $V_{\in_{t_\star+1:T}}[\Pi Y_{t_\star+1}]$; this is so, because even though $t_\star + d + 1 \leq t \leq T$ in the forward sweep, each X_t consists of variables at times $t_\star + 1, \ldots, t_\star + d, \ldots, t$. (In the previous subsection, we used $t_\star = 0$ in the notation, but we generalize this now.) We orient the vectors with Π, as indicated by the definition of the aftcast filters discussed above. Our goal is to generate updates by using aftcasts: we want to compute $P_{\in_{t:T}}[\Pi Y_t]$ and $V_{\in_{t:T}}[\Pi Y_t]$ in terms of $P_{\in_{t+1:T}}[\Pi Y_{t+1}]$ and $V_{\in_{t+1:T}}[\Pi Y_{t+1}]$. The result below describes the efficient computation of such quantities, using the same definitions of $x_t^{\in}$ used above.

Theorem 3.4. *Let $\{x_t\}$ be a difference-stationary process with degree d matrix polynomial $\delta(z)$, such that δ_0 and δ_d are invertible. Let $J X_T$ be the ragged edge sample from $\{x_t\}$ defined via J given in (2.6), where $t_\star = \min T_\star$ is given by (2.2). Assume that $t_\star > 0$ and Assumption A holds. Letting $Q_t = \Pi\, Y_t$, for $1 \leq t \leq t_\star$ the one-step ahead forecasts and prediction variances can be computed from previously computed quantities ($P_{\in_{t+1:T}}[Q_{t+1}]$ and $V_{\in_{t+1:T}}[Q_{t+1}]$) via*

$$P_{\in_{t+1:T}}[x_t] = u_t(1)'\, P_{\in_{t+1:T}}[Q_{t+1}] \tag{3.20}$$

$$V_{\in_{t+1:T}}[x_t] = V_{Q_{t+1}}[x_t] + u_t(1)'\, V_{\in_{t+1:T}}[Q_{t+1}]\, u_t(1), \tag{3.21}$$

where $u_t(1)$ and $V_{Q_{t+1}}[x_t]$ are computed via (3.10) and (3.11). Letting $W_t = I_N[\in_t,]' \left(I_N[\in_t,]\, V_{\in_{t+1:T}}[x_t]\, I_N[\in_t,]'\right)^{-1}$, we can update the information via

$$\begin{aligned} P_{\in_{t:T}}[Q_t] &= P_{\in_{t+1:T}}[Q_t] \\ &\quad + \left[\begin{array}{c} V_{\in_{t+1:T}}[x_t] \\ V_{\in_{t+1:T}}[Q_{t+1}]\, u_t(1) \end{array} \right] W_t\, \left(x_t^{\in} - I_N[\in_t,]\, P_{\in_{t+1:T}}[x_t] \right) \end{aligned} \tag{3.22}$$

$$\begin{aligned} V_{\in_{t:T}}[Q_t] &= \left[\begin{array}{cc} V_{\in_{t+1:T}}[x_t] & u_t(1)'\, V_{\in_{t+1:T}}[Q_{t+1}] \\ V_{\in_{t+1:T}}[Q_{t+1}]\, u_t(1) & V_{\in_{t+1:T}}[Q_{t+1}] \end{array} \right] \\ &\quad - \left[\begin{array}{c} V_{\in_{t+1:T}}[x_t] \\ V_{\in_{t+1:T}}[Q_t]\, u_t(1) \end{array} \right] W_t\, I_N[\in_t,] \left[\begin{array}{c} V_{\in_{t+1:T}}[x_t] \\ V_{\in_{t+1:T}}[Q_t]\, u_t(1) \end{array} \right]'. \end{aligned} \tag{3.23}$$

With these expressions, the backward sweep recursions are complete; once the $t = 1$ iteration is complete, we have finally obtained $P_{\in_{1:T}}[\Pi Y_1]$ and $V_{\in_{1:T}}[\Pi Y_1]$. Note that if we wish to obtain some forecasts and/or aftcasts as well, this can be achieved by a relabelling of the initial ($t = 1$) and final ($t = T$) time points. (*Ecce Signum* has this capability.) The backward sweep residuals extend those so far obtained by the forward sweep (3.18), and are defined via

$$\epsilon_t = V_{\in_{t+1:T}}[x_t^{\in}]^{-1/2} \left(x_t^{\in} - P_{\in_{t+1:T}}[x_t^{\in}] \right). \tag{3.24}$$

These can be pre-pended to the forward sweep residuals, yielding residuals for all values of t except for $t_\star + 1, \ldots, t_\star + d$ (which correspond to the initial values of the process) and those times for which x_t is fully missing. Then summing $\epsilon_t' \, \epsilon_t$ in (3.19) – and adding the sum of log determinants of prediction error variances – yields the portion of the Gaussian divergence corresponding to the stationary data $\Delta_J \, \underline{X}_T$ described in Theorem 2.1. In this manner, the forward and backward sweeps allow for efficient computation of the exact Gaussian likelihood, correctly accounting for initial values – without recourse to *ad hoc* state space methods such as diffuse initalization – as recommended in [3].

3.4. Computation of casting MSE and WK MSE

Having computed forecasts, aftcasts, and midcasts of the sample, we obtain the extended series $\{\widetilde{x}_t\}$. This extended series is filtered, using a truncated filter $\Psi(B)$, and the result is our approximation to $\widehat{s}_{t|\in_{1:T}}$. By taking the truncation order sufficiently large, the approximation can be improved to any desired degree (though in practice there is a tradeoff between accuracy and computational cost, as higher truncation orders require more casts). Truncation of a bi-infinite filter can be problematic if the coefficients decay slowly, and the filter is applied to nonstationary data. Whereas for stationary series the forecasts and aftcasts converge to the mean – which is zero if this effect has been removed – such is not the case for nonstationary processes, indicating that filter coefficients will be weighting potentially large forecasts and aftcasts at large lags removed from the sample boundaries. Hence it is very important that the truncation is applied at sufficiently high indices, where the magnitude of coefficients is of the order of a machine precision zero. Failure to do so can result in erroneous level defects; if the sum of truncated filter's coefficients differs substantially from unity, the filter will distort constants – trend and seasonal adjustment filters would then yield extractions with a vertical displacement distortion.

We now discuss the calculation of signal extraction error covariances, and begin with extended theory for the multivariate WK filter. The general theory for multivariate signal extraction of difference-stationary time series is discussed in [27], wherein explicit formulas for the resulting WK filter's frequency response function (frf) are presented – in the case that each component series has a different scalar differencing polynomial. This is tantamount to considering $\delta(z)$ to be a diagonal matrix polynomial; here we generalize the frf formulas to the general case, allowing for co-integrated signals. The first step is to generalize the framework of Section 2.2: we suppose there are matrix differencing polynomials $\delta^s(z)$ and $\delta^n(z)$ such that $\underline{s}_t = \delta^s(B) s_t$ and $\underline{n}_t = \delta^n(B) n_t$ are difference-stationary processes, and let their acvfs be denoted γ_s and γ_n (and γ_x now denotes the acvf of the differenced process $\{\underline{x}_t\}$). Assume that $\delta^s(z)$ and $\delta^n(z)$ are commutative and are relatively prime, i.e., $\det \delta^s(z)$ and $\det \delta^n(z)$ share no common zeroes on $\mathbb{C}$. Let

$$\delta(z) = \delta^s(z)\, \delta^n(z) = \delta^n(z)\, \delta^s(z). \tag{3.25}$$

Then because the data process equals signal plus noise, i.e., $x_t = s_t + n_t$, it follows from (2.1) that

$$\underline{x}_t = \delta(B) x_t = \delta^n(B)\, \underline{s}_t + \delta^s(B)\, \underline{n}_t. \tag{3.26}$$

Denote the autocovariance generating function (acvg) by $\gamma_x(z) = \sum_{h \in \mathbb{Z}} \gamma_x(h) z^h$ for the data process (and similarly for signal and noise), and recall that plugging in $z = e^{-i\lambda}$ yields the spectral density. Let the $*$ operator on a matrix polynomial in z denote applying the transpose and $z \mapsto z^{-1}$. Then the acvgs for signal and noise are related via

$$\gamma_x(z) = \delta^n(z)\, \gamma_s(z)\, \delta^n(z)^* + \delta^s(z)\, \gamma_n(z)\, \delta^s(z)^*. \tag{3.27}$$

It can happen in applications that $\gamma_s(z)$ is singular for some $z = e^{-i\lambda}$, but in such a scenario we must assume that $\gamma_n(z)$ is still invertible so that invertibility of $\gamma_x(z)$ is assured; one example where this situation arises is with a co-integrated signal [22]. Our next result presents a formula for the WK frf and the acvg of the error process $\{\eta_t\}$ (where $\eta_t = \widehat{s}_{t|\infty} - s_t$), allowing for this scenario. It depends on a generalization of Assumption A:

Assumption $\widetilde{A}$: $\{x_{t_\star+1}, \ldots, x_{t_\star+d}\}$ are uncorrelated with $\{\underline{s}_t\}$ and $\{\underline{n}_t\}$.

Assumption $\widetilde{A}$ generalizes the Assumption A of [2] to the multivariate case. Note that here we suppose there are missing values in the process, so that $t_\star$ is identified in the manner discussed in Section 2; however, the signal extraction result estimates the signal process in terms of the entire data process (i.e., $\widehat{s}_{t|\infty}$), so that missing values are disallowed. This is still useful: recall from (2.7) that the WK coefficients are combined with the casted series if there are missing values present.

Theorem 3.5. *Suppose that $\{x_t\}$ is a difference-stationary process composed of signal and noise, such that $\delta^s(z)$ and $\delta^n(z)$ are commutative and are relatively prime, and (3.25) holds. Suppose that Assumption $\widetilde{A}$ holds, and that $\gamma_s(z)$ is invertible on the unit circle but possibly $\gamma_s(z)$ is singular for some values of $z = e^{-i\lambda}$. Letting $P(z) = \delta^s(z)^* \gamma_s(z) \delta^n(z)^* - \delta^n(z)^* \gamma_n(z) \delta^s(z)^*$ and $M(z) = \delta^s(z)^* \delta^s(z) + \delta^n(z)^* \delta^n(z)$, the WK filter frf can be expressed as*

$$\Psi(z) = M(z)^{-1} \left(\delta^n(z)^* \delta^n(z) + P(z)\, \gamma_x(z)^{-1}\, \delta(z) \right)$$

for $z = e^{-i\lambda}$ and $\lambda \in [-\pi, \pi]$. The error process is covariance stationary with acvg $\gamma_\eta(z)$ given by

$$M(z)^{-1} \left(\delta^n(z)^* \gamma_n(z) \delta^n(z) + \delta^s(z)^* \gamma_s(z) \delta^s(z) - P(z) \gamma_x(z)^{-1} P(z)^* \right) M(z)^{-1}.$$

Remark 3. There is a useful extension of these results when we desire to estimate some linear combination of the signal $\{s_t\}$. Let $\varphi(B) = \sum_{j=0}^p \varphi_j \, B^j$ be a degree p matrix polynomial such that $z_t = \varphi(B) s_t$ is the target of estimation. Then the optimal linear estimator of z_t given full or partial information is given by $\varphi(B)\widehat{s}_t$, and the filter coefficients are given by $\{\sum_{j=0}^p \varphi_j \psi_{k-j}\}$, i.e., $\varphi(B)\psi_k$. Then the error acvg for $\{z_t\}$ is given by $\varphi(z) \, \gamma_\eta(\lambda) \, \varphi(z)^*$.

Appendix C contains further details on the calculation of $\Psi(z)$. Next, it follows from the orthogonal decomposition in (2.8) that the covariance of signal extraction errors is obtained by combining the WK filter with casting error covariances, which are denoted by $\kappa_{h\ell} = \mathrm{Cov}[x_h, x_\ell | \in_{1:T}]$. These quantities can be computed using the recursive algorithms of Section 3.3.

Proposition 3.1. *Let $\{x_t\}$ be a difference-stationary process with degree d matrix polynomial $\delta(z)$, such that δ_0 and δ_d are invertible. Suppose that there is a ragged edge sample from $\{x_t\}$, and the process satisfies Assumption A. Moreover, if the filter $\Psi(B)$ is WK additionally suppose that $\{x_t\}$ is a difference-stationary process composed of signal and noise, such that $\delta^s(z)$ and $\delta^n(z)$ are commutative and are relatively prime, and (3.25) holds. Suppose that Assumption $\widetilde{A}$ holds, and that $\gamma_s(z)$ is invertible on the unit circle but possibly $\gamma_s(z)$ is singular for some values of $z = e^{-i\lambda}$. Then the covariance $Cov[s_t - \widehat{s}_{t|\in_{1:T}}, s_u - \widehat{s}_{u|\in_{1:T}}]$ between signal extraction errors is given by*

$$\gamma_\eta(t-u) + \sum_{j,k \in \mathbb{Z}} \psi_j \, Cov[x_{t-j}, x_{u-k} | \in_{1:T}] \, \psi_k' \tag{3.28}$$

for any $t, u \in \mathbb{Z}$. If the filter $\Psi(B)$ is ad hoc, the formula (3.28) can be adjusted by setting $\gamma_\eta(t-u) = 0$.

The second term in (3.28) involves two infinite summations, and in practice is approximated by truncation. Supposing that the filter is truncated at some maximum lag m, we obtain

$$\sum_{j,k=-m}^{m} \psi_j \, \mathrm{Cov}[x_{t-j}, x_{u-k} | \in_{1:T}] \, \psi_k' = \sum_{j,k=-m}^{m} \psi_j \, \kappa_{t-j,u-k} \, \psi_k'$$
$$= [\psi_m, \ldots, \psi_{-m}] \begin{bmatrix} \kappa_{t-m,u-m} & \cdots & \kappa_{t-m,u+m} \\ \vdots & \cdots & \vdots \\ \kappa_{t+m,u-m} & \cdots & \kappa_{t+m,u+m} \end{bmatrix} \begin{bmatrix} \psi_m' \\ \vdots \\ \psi_{-m}' \end{bmatrix}$$

as an approximation to the second term of (3.28). There are applications for which covariances ($t \neq u$) are required, e.g., determining the extraction MSE for some linear combination of the signal (such as a growth rate or annual growth rate).

In the case that the casting MSE is desired, we set $t = u$ and compute the above block quadratic form for $t = 1, 2, \ldots, T$. This can be expensive, but note that the block matrix of covariances may have much sparsity – the nonzero sections arise from midcasts, covariances of aftcasts (upper left portion), covariances of forecasts (lower right portion), and covariances between aftcasts and forecasts (upper right and lower left portions). These latter blocks are zero if the time gap between aftcasts and forecasts exceeds the maximal lag such that the partial autocorrelation is nonzero. It is a simple matter to obtain casted signal estimates: if we prepend aftcasts or append forecasts to the sample, then this will redefine the starting or ending indices ($t = 1$ and $t = T$); then the extended series $\{\widetilde{x}_t\}$ will be suitably lengthened, and casting error covariances can also be obtained. In particular, we can rewrite the casting MSE as

$$\sum_{h,\ell=t-m}^{t+m} \psi_{t-h}\, \kappa_{h,\ell}\, \psi'_{t-\ell} = \sum_{h,\ell \in R_t} \psi_{t-h}\, \kappa_{h,\ell}\, \psi'_{t-\ell},$$

where $R_t = \{t - m, \ldots, t + m\} \cap R$ and R consists of all time indices for which an aftcast, midcast, or forecast is needed (so that $\kappa_{h,\ell}$ is a non-zero matrix if and only if $h, \ell \in R$).

This treatment can be generalized if an asymmetric filter is used. Suppose now that the entire length of the filter is p, but r of these coefficients weight future observations (so $0 \leq r \leq p - 1$, because we shall assume there is some weight on the present observation). Then the non-zero filter coefficients are $\{\psi_{-r}, \ldots, \psi_0, \ldots, \psi_{p-1-r}\}$, and the casting MSE is written

$$\sum_{h,\ell=t-p+1+r}^{t+r} \psi_{t-h}\, \kappa_{h,\ell}\, \psi'_{t-\ell} = \sum_{h,\ell \in R_t} \psi_{t-h}\, \kappa_{h,\ell}\, \psi'_{t-\ell},$$

where $R_t = \{t - p + 1 + r, \ldots, t + r\} \cap R$.

4. Illustrations and applications

4.1. An illustration

We provide an illustration of the algorithms in the case of a bivariate VAR(1) process. Let Φ be the 2×2 coefficient matrix with eigenvalues of modulus less than one; then ξ_t is the lag t partial autocorrelation, and equals zero if and only if $t \geq 2$. Moreover, $\xi_1 = \gamma(1) - \gamma(1)\,\gamma(0)^{-1}\,\gamma(1)'$, and hence (3.1) yields $\ell_2(1)' = \gamma(1)\,\gamma(0)^{-1} = \Phi$, whereas (3.2) yields $u_2(1)' = \gamma(-1)\,\gamma(0)^{-1} = \gamma(0)\,\Phi'\,\gamma(0)^{-1}$ for $t = 1$. Turning to $t \geq 2$, we can apply the observation of Remark 1, obtaining $\ell_3(1)' = [0\, \ell_2(1)'] = [0\, \Phi]$. Also the values of m_{t+1} and n_{t+1} stabilize at $m_3 = \gamma(0) - \gamma(-1)\gamma(0)^{-1}\gamma(1)$ and $n_3 = \gamma(0) - \gamma(1)\gamma(0)^{-1}\gamma(-1)$.

Then (3.9) simplifies to $\underline{\ell}_{t+1}(j) = [0 \, \ldots \, 0 \, \Phi^j]$, where there are $t-1$ blocks of 2×2 zero matrices preceding Φ^j. Also (3.13) simplifies to

$$\Pi' \, \underline{u}_{T-t+1}(j) = [0 \, \ldots \, 0 \, \Xi^j],$$

where $\Xi = \gamma(0) \, \Phi' \, \gamma(0)^{-1}$. (This matrix has the same eigenvalues as Φ, and hence its powers are non-explosive as j increases.) When we come to Theorem 3.3, (3.14) is simply $P_{\in_{1:t-1}}[x_t] = \Phi \, P_{\in_{1:t-1}}[x_{t-1}]$, and (3.15) is $V_{\in_{1:t-1}}[x_t] = n_3 + \Phi \, V_{\in_{1:t-1}}[x_{t-1}] \, \Phi'$. Next, in the forward sweep the update equation (3.16) becomes

$$P_{\in_{1:t}}[x_t] = P_{\in_{1:t-1}}[x_t] + V_{\in_{1:t-1}}[x_{t-1}] \, W_{t-1} \, \left(x_t^{\in} - I_N[\in_t,] \, P_{\in_{1:t-1}}[x_t] \right),$$

and (3.17) becomes

$$V_{\in_{1:t}}[x_t] = V_{\in_{1:t-1}}[x_{t-1}] - V_{\in_{1:t-1}}[x_t] \, W_{t-1} \, I_N[\in_t,] \, V_{\in_{1:t-1}}[x_t]'.$$

From this last formula, it is clear that if there are no missing values at time t, then $V_{\in_{1:t}}[x_t] = 0$. These update equations resemble expressions appearing in the Kalman filter of the SSF, but are adapted to handle ragged edge missing values. Expressions for the backward sweep are quite similar, and are omitted.

4.2. Simulation study

We explore the speed and accuracy of the ragged edge casting algorithm through the simulation of a bivariate Gaussian VAR(1) process, where a variable proportion of missing values are randomly allocated for each simulation (at random for both components of the data vector). There are $10,000$ time series simulated, of sample sizes $T = 50, 100, 200, 400$, and 800, with innovation variance matrix equal to the identity and VAR(1) coefficient matrix

$$\Phi = \left[\begin{array}{cc} 1 & .5 \\ -.2 & .3 \end{array} \right].$$

The proportion of missing values is $P = 0, .1, .2, .3, .4$, or $.5$. We report the average MSE by computing the squared difference between midcast and truth, averaged over the number of missing values, and then averaged over all the simulations. Results are summarized in Table 1. Whereas MSE decreases slightly with increasing sample size, it increases more systematically with P, which reflects how an increasing degree of "missingness" degrades our ability to obtain accurate projections. Average speed (in seconds) of the computations needed for a single simulation are also reported; the algorithm is slower with larger P (since more computations are needed), and speed also deteriorates slightly with increasing sample size.

Table 1
Comparison of MSE and Speed (in seconds).

	Measure	$T = 50$	$T = 100$	$T = 200$	$T = 400$	$T = 800$
$P = 0$	MSE	0	0	0	0	0
	Speed	1.090976	1.102666	1.104912	1.107893	1.111348
$P = .1$	MSE	0.6615658	0.65647	0.6558633	0.6528845	0.6544117
	Speed	1.117089	1.168106	1.169392	1.176506	1.183627
$P = .2$	MSE	0.7100986	0.6999037	0.6979248	0.6964243	0.6954443
	Speed	1.18862	1.190496	1.192662	1.197082	1.214639
$P = .3$	MSE	0.7699053	0.7572977	0.7512301	0.7495556	0.7483932
	Speed	1.254713	1.255488	1.258262	1.263912	1.287535
$P = .4$	MSE	0.8364147	0.827912	0.8257946	0.820716	0.8208299
	Speed	1.855538	1.856432	1.864264	1.870044	1.931155
$P = .5$	MSE	0.9435211	0.9282926	0.9231314	0.9213543	0.919364
	Speed	2.203703	2.204623	2.206942	2.213797	2.263211

4.3. Daily retail data

We now study daily retail series 4482 (Shoe Stores) collected by FirstData, and acquired by the U.S. Census Bureau through a purchase agreement[4]. There is a complex seasonal pattern present, and no apparent trend (see Figure 1). Our goal is to model this *shoe* series, apply an ideal low-pass filter to extract the annual seasonality, and quantify the uncertainty in the *ad hoc* filtering. We remark that due to the ragged edge nature of the data, modeling and analysis cannot be done with either a direct approach or with SSF (unless a non-standard, customized encoding is utilized).

In this scenario we have a univariate time series that is filtered with an *ad hoc* filter; the time series is non-stationary, but each of the seven weekly day-of-week time series are stationary. That is, the Shoe series is rendered stationary by embedding: for $N = 7$, we sub-sample the time series into contiguous blocks of length N. Thus, this daily time series is embedded as a weekly time series with the seven components corresponding to each day of the week. Let the resulting embedded time series be denoted $\{x_t\}$. References on embedding include [29] and [11]; [25] shows that the output of a univariate filter can be expressed instead in terms of a multivariate filter operating on the embedded input time series.

We make application of these ideas to the shoe series, noting that regressors for the univariate process must also be modified by embedding. We consider an *ad hoc* seasonal adjustment filter $\psi(B)$ for the original time series, which can be embedded to a weekly 7-variate filter, denoted $\Psi(L)$; here B is the backshift operator for the daily frequency, and L is the backshift (or lag) operator for a weekly frequency.

Given these preliminaries, we can apply the techniques of Sections 2 and 3 to determine the filter output and casting MSE. October 1, 2012 is a Monday, and we wish to embed our series such that the first component is Sunday, followed by Monday, Tuesday, etc. This means that one missing value for the first week is

[4] Series 4482 is a daily time series covering the period from October 1, 2012 through April 12, 2016. The research was conducted under DMS Project P-7506880.

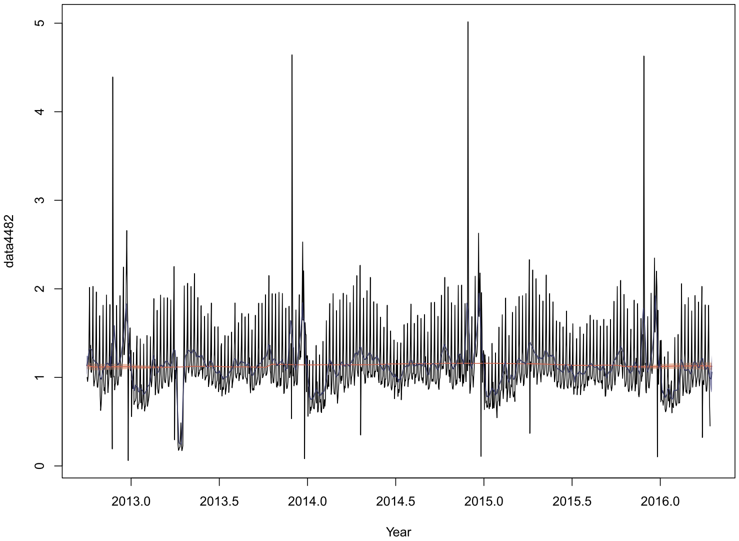

Fig 1. *Time series plot of daily retail series 4482 (Shoe stores), with seasonal adjustment (red) and non-weekly effect (blue); uncertainty bands are shaded.*

included, and there are four missing values for the final week, which corresponds to $T = 185$.

A SVARMA model with regression effects is fitted to the embedded ragged edge time series. The SVARMA has a first order VAR and a first order seasonal VAR, with these specifications based upon exploratory analysis – and confirmed by the whiteness of the resulting residuals (Figure 2). In addition to a mean parameter for each day of week, there are seven holiday regressors: Easter day, pre-Easter (the week antecedent to Easter itself), Cyber Monday, Black Friday, Super Bowl Sunday, Labor Day, and Chinese New Year. When these holiday regressors are embedded, the corresponding parameters are enforced to be the same (i.e., the Easter day parameters for Sunday through Saturday are enforced to be identical, and so on for the other holidays).

The final model fit successfully captures the main dynamic features, and is used to generate midcasts, forecasts, and aftcasts. In order to extract an *ad hoc* trend-irregular, we consider a seasonal adjustment filter $\psi(B) = (1 + B + \ldots + B^{365})(1 + B)B^{-183}/730$, which removes all periodic components of frequency $2\pi j/365$ (for $1 \leq j \leq 182$) while preserving linear trend lines – see [21] for background. This filter is also very close to removing weekly effects, which have frequencies $2\pi/7$, $4\pi/7$, and $6\pi/7$, because with $j = 52, 104, 156$ the annual frequencies are $52/365 \approx 1/7$, $104/365 \approx 2/7$, and $156/365 \approx 3/7$.

Hence, we expect that applying this *ad hoc* filter will remove both annual and weekly seasonality in the series. Although the frequency response function of $\psi(B)$ is real (due to its symmetry), it is not everywhere positive [21] and hence cannot be interpreted as a WK filter; this emphasizes the capabilities of

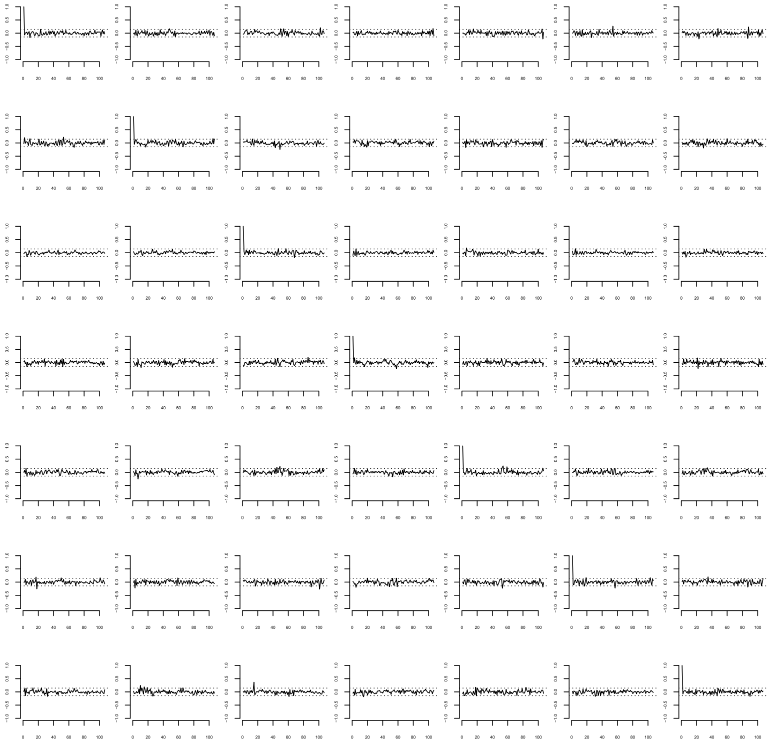

Fig 2. *Sample autocorrelation of residuals for daily retail series 4482 (Shoe stores) embedded as a weekly series. Plots give cross-autocorrelations for Sunday through Saturday (from left to right and top to bottom), with horizontal bands denoting the white noise hypothesis test.*

our framework, which can apply nonparametric filters that can never arise from a model-based framework. Furthermore, note that because $\psi(B)$ is a moving average filter, it has a finite number of non-zero coefficients, and hence no truncation is needed to do forecast and aftcast extension. In particular, 183 forecasts and aftcasts are needed for the scalar filter; the embedded filter $\Psi(L)$ has total length 55, and requires 27 forecasts and aftcasts of the weekly vector time series.

Once the seasonally adjusted component has been obtained, it must be de-embedded, i.e., expressed back in terms of a daily time series. An important aspect of this is to incorporate relevant regression effects (which are all removed prior to filtering). The holiday effects should not be present in the seasonally adjusted component, but the trend level should be preserved. The mean vector of the embedded weekly series contains 7 different level effects, one for each day-of-week time series; these can be viewed as the common mean c plus a deviation of that day-of-week from the common mean. We obtain c by averaging the day-of-week means, and add c to the de-embedded seasonally adjusted component.

The final results are displayed in Figure 1: the data is in black and the seasonally adjusted component is in red (with shaded uncertainty intervals).

If we only suppress the weekly effect, we obtain the blue line. The seasonal adjustment is very close to a horizontal line with some slight movements. This is a satisfactory seasonal adjustment, because the spectral density plot indicates there is no seasonality (neither annual nor weekly) remaining. Additionally, we also consider an *ad hoc* filter $\psi(B)$ that removes only weekly seasonality, defined via $\psi(B) = (1 + B + \ldots + B^6)/7$. We obtain the corresponding $\Psi(L)$ (only one forecast and aftcast is needed), and extract the component, and de-embed – again only adding the overall mean c. The resulting extraction tracks the main monthly movements, unlike the seasonal adjustment, but the weekly effect is not present; interestingly, the weekly effect seems to affect the uncertainty intervals, but not the extraction itself.

Appendix A: Proofs

Proof of Proposition 2.1. Applying the matrices $\Delta_\flat$ and $\Delta_\sharp$ to sub-vectors of X_T, it is apparent that they apply $\delta(B)$ to consecutive elements of the sample, reducing them to stationarity. In particular, applying $\Delta_\flat$ corresponds to (2.4), and applying $\Delta_\sharp$ corresponds to (2.3). Therefore

$$\widetilde{\Delta}\, X_T = \left[\begin{array}{ccc} \Delta_\flat & & 0 \\ 0 & I_{dN} & 0 \\ 0 & & \Delta_\sharp \end{array} \right] \left[\begin{array}{c} X_\flat \\ X_\star \\ X_\sharp \end{array} \right] = \left[\begin{array}{c} \underline{X}_\flat \\ X_\star \\ \underline{X}_\sharp \end{array} \right].$$

This shows that $\widetilde{\Delta}$ is the desired linear transformation. Next, we construct its inverse. Observe that $\widetilde{\Delta}$ can be written as

$$\widetilde{\Delta} = \left[\begin{array}{ccc} C_\flat^{-1} & D_\flat & 0 \\ 0 & I_{dN} & 0 \\ 0 & D_\sharp & C_\sharp^{-1} \end{array} \right],$$

where we have written $\Delta_\flat = [C_\flat^{-1}, D_\flat]$ and $\Delta_\sharp = [D_\sharp, C_\sharp^{-1}]$. This means that $D_\flat$ consists of the right-hand dN columns of $\Delta_\flat$, and $C_\flat$ is the inverse of the left-hand $t_\star N$ columns. This inverse exists, because the matrix in question is block upper triangular with diagonal block entries given by δ_d, which itself is invertible by assumption. It is easy to see that $C_\flat$ is block Toeplitz with jkth block entry given by the $k - j$ matrix in the power series expansion of $\delta^{\text{rev}}(z)^{-1}$, where rev denotes the polynomial with coefficients in reverse order. Similarly, $D_\sharp$ consists of the left-hand dN columns of $\Delta_\sharp$. The remaining right-hand $(T - t_\star - d)N$ columns comprise the inverse of $C_\sharp$, and is block lower triangular with diagonal block entries δ_0. Hence $C_\sharp$ is block Toeplitz with jkth block entry given by the $j - k$ matrix in the power series expansion of $\delta(z)^{-1}$. Now it can be directly checked that the inverse of $\widetilde{\Delta}$ is given by

$$\widetilde{\Delta}^{-1} = \left[\begin{array}{ccc} C_\flat & -C_\flat\, D_\flat & 0 \\ 0 & I_{dN} & 0 \\ 0 & -C_\sharp\, D_\sharp & C_\sharp \end{array} \right]. \qquad \square$$

Proof of Theorem 2.1. Define the matrix $\widehat{\Delta}$ via

$$\widehat{\Delta} = \left[\begin{array}{ccc} I & J_\flat \, C_\flat \, D_\flat & 0 \\ 0 & I_{dN} & 0 \\ 0 & J_\sharp \, C_\sharp \, D_\sharp & I \end{array} \right].$$

Here the upper left and lower right block matrices are identity matrices of dimension equal to N times the row dimension of $J_\flat$ and $J_\sharp$, respectively. It can be shown using permutations that $\widehat{\Delta}$ is invertible with unit determinant. Moreover,

$$\widehat{\Delta} \, J X_T = \left[\begin{array}{c} J_\flat \, C_\flat \, \underline{X}_\flat \\ X_\star \\ J_\sharp \, C_\sharp \, \underline{X}_\sharp \end{array} \right].$$

So if we omit $X_\star$ (by deleting the middle d block rows of $\widehat{\Delta}$), we have a linear transformation of the data vector to a sample $[\underline{X}_\flat', \underline{X}_\sharp']'$ from the stationary, differenced process. In particular, let P be a block permutation matrix that interchanges the first row block $J_\flat \, C_\flat \, \underline{X}_\flat$ of $\widehat{\Delta} \, J X_T$ wth $X_\star$. Then

$$J X_T = \widehat{\Delta}^{-1} \left[\begin{array}{c} J_\flat \, C_\flat \, \underline{X}_\flat \\ X_\star \\ J_\sharp \, C_\sharp \, \underline{X}_\sharp \end{array} \right] = \widehat{\Delta}^{-1} \, P \left[\begin{array}{cc} I_{dN} & 0 \\ 0 & \Delta_J \end{array} \right] \left[\begin{array}{c} X_\star \\ \underline{X}_T \end{array} \right],$$

so that by Assumption A we obtain

$$V \left(J X_T \right) = \widehat{\Delta}^{-1} \, P \left[\begin{array}{cc} V[X_\star] & 0 \\ 0 & \Delta_J \, \Gamma_{T-d} \, {\Delta_J}' \end{array} \right] P' \, \widehat{\Delta}^{-1 \prime}.$$

This matrix is invertible by the assumptions on the spectral density and the initial values, and the formula for the quadratic form immediately follows. Likewise, the formula for the log determinant follows as well, since $\det \widehat{\Delta} = 1$ and $\det P \cdot \det P' = 1$. □

Proof of Theorem 3.1. We begin with some relations that hold for $j \geq 1$. By (2.1) we obtain

$$\begin{aligned} x_{t+j} &= - \sum_{k=1}^{d} \delta_0^{-1} \, \delta_k \, x_{t+j-k} + \delta_0^{-1} \, \underline{x}_{t+j} \\ \widehat{x}_{t+j|\in_{1:t}} &= - \sum_{k=1}^{d} \delta_0^{-1} \, \delta_k \, \widehat{x}_{t+j-k|\in_{1:t}} + \delta_0^{-1} \, \widehat{\underline{x}}_{t+j|\in_{1:t}}. \end{aligned} \tag{A.1}$$

In (A.1) we have linear combinations of "previous" forecasts $\widehat{x}_{t+j-k|\in_{1:t}}$ (because $t+j-k < t+j$), which by the recursive principle we can assume have already been computed and stored. Using Assumption A, $\widehat{\underline{x}}_{t+j|\in_{1:t}}$ can be computed from $\underline{x}_{d+1}, \ldots, \underline{x}_t$ (i.e., $\underline{X}_t$) alone; this is because the vector X_t can be expressed as a linear combination of initial values $X_\star$ and the vector $\underline{X}_t$ by Proposition 2.1.

(One can check that the resulting forecast error $\underline{x}_{t+j} - \widehat{\underline{x}}_{t+j|\in_{1:t}}$ is orthogonal both to $X_\star$ by Assumption A, and to $\underline{X}_t$, showing that the normal equations are satisfied.) Hence $\widehat{\underline{x}}_{t+j|\in_{1:t}} = \underline{\ell}_{t+1-d}(j)' \, \underline{X}_t$. Using this in (A.1) and setting $j = 1$ yields (3.6). The key here is that because $j = 1$, all the quantities $\widehat{x}_{t+j-k|\in_{1:t}}$ are actual observations x_{t+1-k}. The recursions for b_{t+1-d} and a_{t+1-d} now follow from the Toeplitz structure of Δ_{t-d} together with the stationary prediction filter recursions (3.1) and (3.2). Finally, the one-step ahead prediction error is

$$x_{t+1} - \widehat{x}_{t+1|\in_{1:t}} = \delta_0^{-1} \left(\underline{x}_{t+1} - \underline{\ell}_{t+1-d}(j)' \, \underline{X}_t \right),$$

which is just δ_0^{-1} times the prediction error for the difference-stationary process, which has variance n_{t+1-d}. □

Proof of Corollary 3.1. First, (3.8) is immediate from (A.1). Second, to obtain the multi-step ahead prediction filters for the stationary case we use the basic iterative property of linear projections, finding that the j-step ahead predictor is obtained via

$$\begin{aligned}
\underline{\ell}_{t+1-d}(j)' \, \underline{X}_t &= P_{\underline{X}_t}[\underline{x}_{t+j}] = P_{\underline{X}_t}[P_{\underline{X}_{t+j-1}}[\underline{x}_{t+j}]] \\
&= P_{\underline{X}_t}[\underline{\ell}_{t+j-d}(1)' \, \underline{X}_{t+j-1}] = \underline{\ell}_{t+j-d}(1)' \, P_{\underline{X}_t}[\underline{X}_{t+j-1}] \\
&= \underline{\ell}_{t+j-d}(1)' \begin{bmatrix} \underline{X}_t \\ P_{\underline{X}_t}[\underline{x}_{t+1}] \\ \vdots \\ P_{\underline{X}_t}[\underline{x}_{t+j-1}] \end{bmatrix} = \underline{\ell}_{t+j-d}(1)' \begin{bmatrix} I_{(t-d)N} \\ \underline{\ell}_{t+1-d}(1)' \\ \vdots \\ \underline{\ell}_{t+1-d}(j-1)' \end{bmatrix} \underline{X}_t.
\end{aligned}$$

Taking the transpose, we obtain (3.9). □

Proof of Theorem 3.2. Assume $j \geq 1$; again by (2.1) we obtain

$$\begin{aligned}
x_{t+1-j} &= -\sum_{k=1}^{d} \delta_d^{-1} \, \delta_{d-k} \, x_{t+1-j+k} + \delta_d^{-1} \, \underline{x}_{t+d+1-j} \\
P_{Y_{t+1}}[x_{t+1-j}] &= -\sum_{k=1}^{d} \delta_d^{-1} \, \delta_{d-k} \, P_{Y_{t+1}}[x_{t+1-j+k}] + \delta_d^{-1} \, P_{Y_{t+1}}[\underline{x}_{t+d+1-j}]. \quad \text{(A.2)}
\end{aligned}$$

In (A.2) we have linear combinations of "previous" aftcasts (because $d-k > 0$), which by the recursive principle we can assume have already been computed and stored. By the arguments used in the proof of Theorem 3.1, we can utilize Assumption A again to assert that $P_{Y_{t+1}}[\underline{x}_{t+d+1-j}] = P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}]$, which is a j-step behind aftcast of a stationary process, given a sample of size $T-t-d$. Hence $P_{Y_{t+1}}[\underline{x}_{t+d+1-j}] = \underline{u}_{T-t-d+1}(j)' \, \Pi \, \underline{Y}_{T-t-d}$, and applying this with $j = 1$ to (A.2) yields (3.10). The other recursions, and (3.11), follow along the lines given in the proof of Theorem 3.1. □

Proof of Corollary 3.2. Formula (3.12) follows from (A.2). Next, we have

$$\underline{u}_{T-t-d+1}(j)' \, \Pi \, \underline{Y}_{T-t-d} = P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+1-j}]$$

$$
\begin{aligned}
&= P_{\underline{Y}_{T-t-d}}[P_{\underline{Y}_{T-t-d+j-1}}[\underline{x}_{t+d+1-j}]] = P_{\underline{Y}_{T-t-d}}[\underline{u}_{T-t-d+j}(1)' \, \Pi \, \underline{Y}_{T-t-d+j-1}] \\
&= \underline{u}_{T-t-d+j}(1)' \, \Pi \, P_{\underline{Y}_{T-t-d}}[\underline{Y}_{T-t-d+j-1}] \\
&= \underline{u}_{T-t-d+j}(1)' \, \Pi \begin{bmatrix} \underline{Y}_{T-t-d} \\ P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d}] \\ \vdots \\ P_{\underline{Y}_{T-t-d}}[\underline{x}_{t+d+2-j}] \end{bmatrix} \\
&= \underline{u}_{T-t-d+j}(1)' \, \Pi \begin{bmatrix} I_{(T-t-d)N} \\ \underline{u}_{T-t-d+1}(1)' \, \Pi \\ \vdots \\ \underline{u}_{T-t-d+1}(j-1)' \, \Pi \end{bmatrix} \underline{Y}_{T-t-d}.
\end{aligned}
$$

Taking the transpose, we obtain (3.13). □

The proof of Theorem 3.3 utilizes the following lemma.

Lemma A.1. *Under the assumptions of Theorem 3.3, $P_{\in_{1:t}}[x_s]$ is given by*

$$P_{\in_{1:t-1}}[x_s] + Cov[x_s, x_t^{\in} | \in_{1:t-1}] \, V_{\in_{1:t-1}}[x_t^{\in}]^{-1} \, (x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}]) \qquad \text{(A.3)}$$

for $1 \leq s \leq t$. In the case that $\in_t = \emptyset$, the second term of (A.3) is absent.

Proof of Lemma A.1. We note that the quantity that multiplies $x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}]$ on the left corresponds to the Kalman gain in state space algorithms, although the context is different here, as $x_t^{\in}$ represents part of the data vector rather than part of an unobserved state vector. To verify that $P_{\in_{1:t}}[x_s]$ is given by formula (A.3), it suffices to show the error $x_s - P_{\in_{1:t}}[x_s]$ is uncorrelated with variables in $\in_{1:t}$. The error is

$$\left(x_s - P_{\in_{1:t-1}}[x_s]\right) - \text{Cov}[x_s, x_t^{\in} | \in_{1:t-1}] \, V_{\in_{1:t-1}}[x_t^{\in}]^{-1} \, (x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}]),$$

and both summands are clearly uncorrelated with $\in_{1:t-1}$. But $\in_{1:t}$ consists of $\in_{1:t-1}$ together with $x_t^{\in}$. So we can take the covariance of the error with $x_t^{\in}$, or equivalently with $x_t^{\in} - P_{\in_{1:t-1}}[x_t^{\in}]$, and this covariance is clearly zero. □

Proof of Theorem 3.3. Formula (3.14) is proved by nested projections:

$$P_{\in_{1:t-1}}[x_t] = P_{\in_{1:t-1}}[P_{X_{t-1}}[x_t]] = P_{\in_{1:t-1}}[\ell_t(1)' X_{t-1}] = \ell_t(1)' \, P_{\in_{1:t-1}}[X_{t-1}].$$

Next, we can write the projection error as

$$x_t - P_{\in_{1:t-1}}[x_t] = \left(x_t - P_{X_{t-1}}[x_t]\right) + \left(P_{X_{t-1}}[x_t] - P_{\in_{1:t-1}}[x_t]\right).$$

The first term on the right hand side involves the prediction error obtained if there were no previous missing values, and is orthogonal to all linear functions of the data vector X_{t-1}; the second term involves discrepancies between projections on full and partial information sets, and is therefore a function of X_{t-1}, so the first and second summands are orthogonal. In particular,

$$P_{X_{t-1}}[x_t] - P_{\in_{1:t-1}}[x_t] = \ell_t(1)' \, \left(X_{t-1} - P_{\in_{1:t-1}}[x_t]\right)$$

by (3.14). This formula cannot be used for calculations (because the X_{t-1} need not be fully observed), but assists us to compute the prediction error variance. Using the above expression and taking the variance of the projection error now yields (3.15).

For the updates, apply Lemma A.1, so that with $1 \leq s \leq t-1$ in (A.3) we obtain

$$P_{\in_{1:t}}[x_s] = P_{\in_{1:t-1}}[x_s] + E_s\, V_{\in_{1:t-1}}[X_{t-1}]\, \ell_t(1)\, W_{t-1}\, \left(x_t^{\in} - I_N[\in_t,]\, P_{\in_{1:t-1}}[x_t] \right),$$

where E_s is a block row matrix of zeroes except for I_N on block s. The case of $s = t$ yields

$$P_{\in_{1:t}}[x_t] = P_{\in_{1:t-1}}[x_t] + V_{\in_{1:t-1}}[x_t]\, W_{t-1}\, \left(x_t^{\in} - I_N[\in_t,]\, P_{\in_{1:t-1}}[x_t] \right).$$

Putting the cases together yields (3.16). The error $X_t - P_{\in_{1:t}}[X_t]$ is

$$\begin{aligned} &\left[\begin{array}{c} X_{t-1} - P_{\in_{1:t-1}}[X_{t-1}] \\ x_t - P_{\in_{1:t-1}}[x_t] \end{array} \right] \\ &\quad - \left[\begin{array}{c} V_{\in_{1:t-1}}[X_{t-1}]\, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right] W_{t-1}\, \left(x_t^{\in} - I_N[\in_t,]\, P_{\in_{1:t-1}}[x_t] \right). \end{aligned}$$

Using the fact that the second term is in $\in_{1:t}$ and is therefore orthogonal to $X_t - P_{\in_{1:t}}[X_t]$, the projection MSE $V_{\in_{1:t}}[X_t]$ is

$$\begin{aligned} &V \left[\begin{array}{c} X_{t-1} - P_{\in_{1:t-1}}[X_{t-1}] \\ x_t - P_{\in_{1:t-1}}[x_t] \end{array} \right] \\ &\quad - \left[\begin{array}{c} V_{\in_{1:t-1}}[X_{t-1}]\, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right] W_{t-1}\, I_N[\in_t,] \left[\begin{array}{c} V_{\in_{1:t-1}}[X_t]\, \ell_t(1) \\ V_{\in_{1:t-1}}[x_t] \end{array} \right]^{\prime}. \end{aligned}$$

Now the variance of the first term on the right hand side can be expanded in block form, as given in (3.17), since $\mathrm{Cov}[X_{t-1}, x_t - P_{\in_{1:t}}[x_t]]$ equals

$$V_{\in_{1:t-1}}[X_{t-1}]\, \ell_t(1) - V_{\in_{1:t-1}}[X_{t-1}]\, \ell_t(1)\, W_{t-1}\, I_N[\in_t,]\, V_{\in_{1:t-1}}[x_t]. \qquad \square$$

Proof of Theorem 3.4. The proof is the same as that of Theorem 3.3, only taking account of the time-reversed structure. The predictor $u_t(1)$ is oriented such that the upper components weight observations close to the desired target x_t, and hence it must be applied to ΠY_{t+1}. Also, when we partition block vectors, the current observation will be at the top rather than the bottom. □

Proof of Theorem 3.5. The technique of proof follows that of [20]. First, the differenced signal is stationary and can be projected on $\{x_t\}$. By Assumption $\widetilde{A}$, it suffices to project $\underline{s}_t$ on $\{\underline{x}_t\}$, i.e., the initial values are not needed. We claim the frf for the filter $\Theta(B)$ that estimates $\underline{s}_t$ from $\{\underline{x}_t\}$ is $\Theta(z) = \gamma_s(z) \delta^n(z)^* \gamma_x(z)^{-1}$. This is proved by verifying that the error process is orthogonal to $\{\underline{x}_t\}$. For any $h \in \mathbb{Z}$

$$\mathrm{Cov}[\Theta(B) \underline{x}_t, \underline{x}_{t-h}] = \langle z^{-h} \Theta(z) \gamma_x(z) \rangle = \langle z^{-h} \gamma_s(z) \delta^n(z)^* \rangle,$$

where $\langle g(z) \rangle$ is a shorthand for $(2\pi)^{-1} \int_{-\pi}^{\pi} g(e^{-i\lambda})\, d\lambda$. On the other hand,

$$\mathrm{Cov}[\underline{s}_t, \underline{x}_{t-h}] = \mathrm{Cov}[\underline{s}_t, \delta^n(B)\underline{s}_{t-h} + \delta^s(B)\underline{n}_{t-h}] = \langle z^{-h} \gamma_s(z) \delta^n(z)^* \rangle,$$

which uses (3.26) and Assumption $\widetilde{A}$. This shows optimality of $\Theta(B)$. Similarly, we can project the differenced noise onto $\{x_t\}$, and the filter $\Phi(B)$ expressing this estimate has frf $\Phi(z) = \gamma_n(z)\delta^s(z)^* \gamma_x(z)^{-1}$. To summarize,

$$\begin{aligned} \delta^s(B)\, P_{\{x_t\}}[s_t] &= P_{\{x_t\}}[\underline{s}_t] = \Theta(B)\underline{x}_t = \Theta(B)\delta(B)x_t \\ \delta^n(B)\, P_{\{x_t\}}[n_t] &= P_{\{x_t\}}[\underline{n}_t] = \Phi(B)\underline{x}_t = \Phi(B)\delta(B)x_t. \end{aligned}$$

This implies that $\delta^s(B)\Psi(B) = \Theta(B)\delta(B)$ and $\delta^n(B)(1 - \Psi(B)) = \Phi(B)\delta(B)$, or in terms of z

$$\begin{aligned} \delta^s(z)^*\delta^s(z)\Psi(z) &= \delta^s(z)^*\Theta(z)\delta(z) \\ \delta^n(z)^*\delta^n(z)\Psi(z) &= \delta^n(z)^*\delta^n(z) - \delta^n(z)^*\Phi(z)\delta(z). \end{aligned}$$

Clearly, we should add the two equations and apply $M(z)^{-1}$ to solve for $\Psi(z)$; this gives the stated formula in the theorem, provided $M(z)$ is invertible on the unit circle. For a z of unit magnitude such that $M(z)$ has reduced rank, there exists a non-zero v such that $M(z)v = 0$. Hence $0 = v'Mv$ is the sum of two non-negative definite quadratic forms, both of which must therefore be zero. This implies that $\delta^s(z)v = 0$ and $\delta^n(z)v = 0$. Hence both $\delta^s(z)$ and $\delta^n(z)$ are singular, indicating that z is a common zero of $\det \delta^s(\cdot)$ and $\det \delta^n(\cdot)$. This contradicts the assumption that these matrix polynomials are relatively prime, and hence no such z exists, and $M(z)$ is on invertible on the unit circle.

A second proof of the optimality of the given filter can be provided by directly showing that the error process is uncorrelated with $\{\underline{x}_t\}$, using Assumption $\widetilde{A}$. We omit this, but derive the error process acvg. It follows that $\Psi(z) = M(z)^{-1}(\delta^n(z)^*\delta^n(z) + P(z)\gamma_x(z)^{-1}\delta(z))$. Then

$$\begin{aligned} \eta_t &= \Psi(B)n_t - (1 - \Psi(B))s_t \\ &= M(B)^{-1}(\delta^n(B)^*\underline{n}_t + P(B)\gamma_x(B)^{-1}\delta^s(B)\underline{n}_t) \\ &\quad - M(B)^{-1}(\delta^s(B)^*\underline{s}_t - P(B)\gamma_x(B)^{-1}\delta^n(B)\underline{s}_t) \\ &= M(B)^{-1}\Big([\delta^n(B)^* + P(B)\gamma_x(B)^{-1}\delta^s(B)]\underline{n}_t \\ &\qquad -[\delta^s(B)^* + P(B)\gamma_x(B)^{-1}\delta^n(B)]\underline{s}_t\Big), \end{aligned}$$

where the application of $*$ to a matrix polynomial in B indicates that the transpose should be applied, followed by replacing B by B^{-1}, the forward shift operator. Then using (3.27), the acvg $\gamma_\eta(z)$ for $\{\eta_t\}$ is

$$\begin{aligned} &M(z)^{-1}\Big([\delta^n(z)^* + P(z)\gamma_x(z)^{-1}\delta^s(z)]\gamma_n(z)[\delta^n(z)^* + P(z)\gamma_x(z)^{-1}\delta^s(z)]^* \\ &\quad [\delta^s(z)^* - P(z)\gamma_x(z)^{-1}\delta^n(z)]\gamma_s(z)[\delta^s(z)^* - P(z)\gamma_x(z)^{-1}\delta^n(z)]^*\Big)\, M(z)^{-1}, \end{aligned}$$

which simplifies to the stated formula. □

Proof of Proposition 3.1. Formula (3.28) follows directly from (2.7), using the orthogonality of the two summands, and noting that $\eta_t = s_t - \widehat{s}_{t|\infty}$ and $x_{t-j} - \widetilde{x}_{t-j}$ is the casting error. However, the first term of (2.7) is absent if the filter is not WK, and we are considering an *ad hoc* filter. □

Appendix B: Discussion of signal extraction algorithms

Some further discussion of the motivating computational issues of signal extraction is provided here. When the sample size T is small, say 500 or less (though cross-sectional dimension N also plays a role here), non-recursive (e.g., brute force) methods based upon explicit matrix formulas [27] can be employed to obtain signal extraction estimates. Essentially, the determination of predictors and error covariances revolves upon the calculation of partial covariances, wherein a matrix inversion is required; straight Cholesky approaches are viable when the overall matrix dimension is small – as advocated in [20]. However, with high frequency data or high-dimensional time series the direct approaches tend to fail – either outright through memory allocation violations and/or numerical stability issues associated with ill-conditioned matrices (this can happen when the process' spectral density is non-invertible, yielding at least one eigenvalue that is approximately zero when the Toeplitz autocovariance matrix is large), or through impracticable computation times, e.g., a minute for a single likelihood evaluation indicates that days or weeks may be necessary to run numerical optimization. Moreover, the matrix approach is not easily extended to handle missing values.

In order to circumvent the calculation of exact matrix formulas, the author proceeded to determine the WK filter through frequency domain calculations and apply a suitably truncated sequence to the extended data; once the forecasts and aftcasts were computed and appended, convolution methods rendered the filtering extremely speedy. However, the signal extraction error available from the WK error (the error variance arising from the ideal case of a bi-infinite sample) only represents part of the total extraction error – one must also account for the casting errors, and their covariances (aggregated appropriately by the truncated filter) offer a substantial contribution at the sample boundary. Given these motivations, the algorithms of this paper render feasible the efficient computation of this second contribution to signal extraction error. For an *ad hoc* filter, there is no first portion (because signal is essentially defined differently) and its error cannot be described at all without this second portion arising from error covariances.

While the new algorithms here require a substantial exertion to implement, the framework is both more flexible and less strenuous than SSF. Table 2 provides a comparison of the new recursive algorithms of this paper to direct matrix approaches (discussed in [27] and [24]) and SSF, summarizing the introductory discussion given above. Regarding the ragged edge problem, it seems that with some custom coding the SSF can be adapted to handle such situations, although the basic implementation [14] treats values as missing (at some particular time)

TABLE 2

Comparison of Algorithmic Methods. T is sample size, and N is series dimension.

Capability	Recursive	Matrix	SSF
Large T, moderate N	yes	no	yes
Ragged Edge	yes	partial	partial
Growth Rate Signals	yes	yes	partial
Ad hoc Filters	yes	yes	partial
Correct Initialization	yes	yes	no
Models	any acf	any acf	markov

across all variables. Similarly, linear combinations of signals (such as growth rates) can be generated from SSF with little difficulty, but the uncertainty is not obtainable without a customized Kalman filter that iterates prediction error covariances across time lags. The problem of applying *ad hoc* filters can be viewed as a generalization of computing linear combinations of a signal. As for the problems with the diffuse intialization commonly used in SSF for nonstationary processes, this has been extensively discussed in [3] and [14].

The paper is written to give a comprehensive description of recursive algorithms needed for casting and signal extraction, and therefore contains a mixture of known and novel results. We summarize the main contributions below, indicating where they can be found in the paper:

- Factorization of the Gaussian divergence for difference-stationary processes: extends univariate results of [24] to the multivariate case, allowing for ragged edge missing values and non-scalar differencing operators (Section 2).
- Recursive algorithms for one-step ahead and multi-step ahead predictors for difference-stationary processes: generalizes recursions for stationary predictors given in [23] to difference-stationary and multi-step cases (Section 3.1).
- Predictors for the past via time reversal (Section 3.2).
- Recursive casting algorithm for ragged edge difference-stationary processes: new method with minimal storage, providing both casts and casting error covariances (Section 3.3).
- Algorithms for filter MSE: the algorithm for combining a given filter (which can be *ad hoc*) with casting error covariances is given, and the WK filter formula is extended from the scalar differencing polynomial case of [27] to allow for matrix differencing polynomials (Section 3.4).
- Extensions to growth rates of signals: algorithms to yield estimates and MSE for linear combinations of a desired signal are provided (Section 3.4).

Appendix C: WK computations for structural time series

This appendix describes the calculation of frequency response functions for WK signal extraction filters. Adopting the general framework of [22], we suppose the data process $\{x_t\}$ can be decomposed in terms of $J+2$ latent processes, where J of these processes are difference stationary with scalar differencing operators.

The remaining two components correspond to a cycle $\{\rho_t\}$ and an irregular $\{\iota_t\}$, both of which are stationary. We denote the non-stationary components by $\{\chi_t^{(j)}\}$ for $1 \leq j \leq J$. The components are related via

$$x_t = \sum_{j=1}^{J} \chi_t^{(j)} + \rho_t + \iota_t.$$

Let each scalar differencing operator be denoted $\delta^{(j)}(B)$; these have distinct unit roots by assumption. When differenced to stationarity, a non-stationary component is denoted $\{\underline{\chi}_t^{(j)}\}$, where $\underline{\chi}_t^{(j)} = \delta^{(j)}(B)\chi_t^{(j)}$. In order to reduce the data process $\{x_t\}$ to stationarity – necessary to evaluate the Gaussian likelihood via Durbin-Levinson algorithm – we must apply the differencing operator $\delta(B) = \prod_{j=1}^{J} \delta^{(j)}(B)$, and this is the minimal degree operator with this property. Setting $\delta^{(-j)}(B) = \prod_{k \neq j} \delta^{(k)}(B)$ (if $J = 1$, this is equal to one), we obtain

$$\underline{x}_t = \delta(B)x_t = \sum_{j=1}^{J} \delta^{(-j)}(B)\, \underline{\chi}_t^{(j)} + \delta(B)\rho_t + \delta(B)\iota_t.$$

Each latent process is driven by white noise innovations, with a covariance matrix of possibly reduced rank – though we stipulate that the irregular has full rank. These matrices are denoted by $\Sigma^{(j)}$ for the non-stationary processes, and by Σ^{ρ} and Σ^{ι} for the cycle and irregular:

$$\underline{\chi}_t^{(j)} \sim \mathrm{WN}(0, \Sigma^{(j)}) \qquad \iota_t \sim \mathrm{WN}(0, \Sigma^{\iota}).$$

The Generalized Cholesky Decomposition (GCD) for each white noise covariance matrix produces lower triangular matrices L and diagonal matrices D of column dimension r, where $r \leq N$ is the rank, such that $\Sigma = L\,D\,L'$. (See [22] for more detail.) These matrices will be super-scripted in correspondence with each latent component. Let f_ρ denote the scalar spectral density of the autoregressive portion of the cycle, so that when multiplied by Σ^{ρ} we obtain that process' multivariate spectral density.

Given these preliminaries, we can write down formulas for the WK signal extraction frequency response functions, and examine their behavior at so-called co-integrating frequencies. An $\omega \in [-\pi, \pi]$ is a co-integrating frequency for the jth latent process if $e^{-i\omega}$ is a root of $\delta^{(j)}(B)$. The reason for the terminology is the following: latent process j is co-integrated of rank $N - r$ if and only if the space of left co-integrating vectors has dimension $N - r$ (which means that application of a co-integrating vector reduces the order of non-stationarity, up to fixed effects, from $\delta(B)$ to $\delta^{(-j)}(B)$), which is true if and only if $\Sigma^{(j)}$ has rank r. The spectral density of $\{\underline{x}_t\}$ at a co-integrating frequency is equal to $\Sigma^{(j)}$, and hence has rank r. Hence, there are co-integrating vectors computable from the GCD of $\Sigma^{(j)}$, such that their application to the data process removes non-stationarity associated with frequency ω.

These claims are apparent once we compute the spectral density of the differenced data process:

$$f_{\underline{x}}(\lambda) = \sum_{j=1}^{J} \left|\delta^{(-j)}(e^{-i\lambda})\right|^2 \Sigma^{(j)} + \left|\delta(e^{-i\lambda})\right|^2 f_\rho(\lambda)\, \Sigma^\rho + \left|\delta(e^{-i\lambda})\right|^2 \Sigma^\iota.$$

Then if $\lambda^{(j)}$ is the co-integrating frequency for the jth latent process, we have $\delta^{(j)}(e^{-i\lambda^{(j)}}) = 0$, but $\delta^{(k)}(e^{-i\lambda^{(j)}}) \neq 0$ for $k \neq j$. Therefore $f_{\underline{x}}(\lambda^{(j)})$ is given by $\left|\delta^{(-j)}(e^{-i\lambda^{(j)}})\right|^2 \Sigma^{(j)}$, which has rank $r_j \leq N$. Note that if the roots are close to one another, it is possible for $\delta^{(-j)}(e^{-i\lambda^{(j)}})$ to be close to zero.

The formulas for the WK frf in each case are obtained by application of Theorem 3.5; we denote the various filter frfs with a superscript corresponding to each signal, i.e., $\Psi^{(j)}$, Ψ^ρ, Ψ^ι. Away from co-integrating frequencies, they are explicitly given by

$$\begin{aligned}
\Psi^{(j)}(e^{-i\lambda}) &= \left|\delta^{(-j)}(e^{-i\lambda})\right|^2 \Sigma^{(j)}\, f_{\underline{x}}(\lambda)^{-1} \\
\Psi^{\rho}(e^{-i\lambda}) &= \left|\delta(e^{-i\lambda})\right|^2 f_\rho(\lambda)\, \Sigma^\rho\, f_{\underline{x}}(\lambda)^{-1} \\
\Psi^{\iota}(e^{-i\lambda}) &= \left|\delta(e^{-i\lambda})\right|^2 \Sigma^\iota\, f_{\underline{x}}(\lambda)^{-1}.
\end{aligned}$$

Each $\Psi^{(j)}(B)$ is a type of signal-pass filter for extracting the jth non-stationary signal, whereas Ψ^ρ is the band-pass filter, and Ψ^ι is the high-pass filter. To express the jth signal extraction frf at signal frequency $\lambda^{(j)}$, let

$$G^{(j)}(\lambda) = \sum_{\ell \neq j}^{J} \left|\delta^{(-j,-\ell)}(e^{-i\lambda})\right|^2 \Sigma^{(\ell)} + \left|\delta^{(-j)}(e^{-i\lambda})\right|^2 f_\rho(\lambda)\, \Sigma^\rho + \left|\delta^{(-j)}(e^{-i\lambda})\right|^2 \Sigma^\iota$$

for $1 \leq j \leq J$, where $\delta^{(-j,-k)}(B) = \prod_{\ell \neq j,k} \delta^{(\ell)}(B)$; if $J = 1$, this polynomial is interpreted as zero, and equals one in the case $J = 2$. Then it follows that

$$f_{\underline{x}}(\lambda) = \left|\delta^{(-j)}(e^{-i\lambda})\right|^2 \Sigma^{(j)} + \left|\delta^{(j)}(e^{-i\lambda})\right|^2 G^{(j)}(\lambda).$$

Because $\delta^{(-j)}(e^{-i\lambda^{(j)}}) \neq 0$ and Σ^ι has full rank, $G^{(j)}(\lambda^{(j)})$ has full rank, and is invertible. Using the GCD for $\Sigma^{(j)}$ for λ in a neighborhood of $\lambda^{(j)}$ the following matrix is well-defined:

$$H^{(j)}(\lambda) = I_N - L^{(j)} \left(\frac{\left|\delta^{(j)}(e^{-i\lambda})\right|^2}{\left|\delta^{(-j)}(e^{-i\lambda})\right|^2} I_N + L^{(j)\prime} G^{(j)}(\lambda)^{-1} L^{(j)} \right)^{-1} L^{(j)\prime}\, G^{(j)}(\lambda)^{-1}.$$

Note that $L^{(j)\prime}\, G^{(j)}(\lambda)^{-1}\, L^{(j)}$ is well-defined for λ in a neighborhood of $\lambda^{(j)}$, and this matrix moreover is invertible (it has rank r_j, which is full). Also,

$$\lim_{\lambda \to \lambda^{(j)}} H^{(j)}(\lambda) = \left\{ I_N - L^{(j)} \left(L^{(j)\prime}\, G^{(j)}(\lambda^{(j)})^{-1}\, L^{(j)} \right)^{-1} L^{(j)\prime}\, G^{(j)}(\lambda^{(j)})^{-1} \right\},$$

which is taken as the definition of $H^{(j)}(\lambda^{(j)})$. Then

$$\Psi^{(j)}(e^{-i\lambda}) = \begin{cases} {|\delta^{(-j,-k)}(e^{-i\lambda})|}^2 \, \Sigma^{(j)} \, {G^{(j)}(\lambda)}^{-1} \, H^{(j)}(\lambda) \text{ if } \lambda = \lambda^{(k)}, k \neq j \\ I_N - H^{(j)}(\lambda) \text{ if } \lambda = \lambda^{(j)}, \end{cases}$$

which is proved in the first case by the Sherman-Woodbury identity

$$f_{\underline{x}}(\lambda)^{-1} = {|\delta^{(j)}(e^{-i\lambda})|}^{-2} \, {G^{(j)}(\lambda)}^{-1} \, H^{(j)}(\lambda).$$

The second case follows from

$$\Psi^{(j)}(e^{-i\lambda}) = I_N - {|\delta^{(j)}(e^{-i\lambda})|}^2 \, G^{(j)}(\lambda) \, f_{\underline{x}}(\lambda)^{-1}.$$

As for the band-pass and high-pass filters, it follows that at $\lambda = \lambda^{(j)}$ they can be expressed as

$$\begin{aligned} \Psi^{\rho}(e^{-i\lambda}) &= {|\delta^{(-j)}(e^{-i\lambda})|}^2 \, f_{\rho}(\lambda) \, \Sigma^{\rho} \, {G^{(j)}(\lambda)}^{-1} \, H^{(j)}(\lambda) \\ \Psi^{\iota}(e^{-i\lambda}) &= {|\delta^{(-j)}(e^{-i\lambda})|}^2 \, \Sigma^{\iota} \, {G^{(j)}(\lambda)}^{-1} \, H^{(j)}(\lambda). \end{aligned}$$

The error spectral densities $f_{\eta}^{(j)}(\lambda)$ for each signal can be computed in terms of the same quantities:

$$\begin{cases} \left(\Psi^{(j)}(e^{-i\lambda}) \, G^{(j)}(\lambda) + [I_N - \Psi^{(j)}(e^{-i\lambda})] \, \Sigma^{(j)} \right) / {|\delta^{(-j)}(e^{-i\lambda})|}^2 \text{ if } \lambda = \lambda^{(j)} \\ \left(G^{(j)}(\lambda) \, {\Psi^{(j)}(e^{-i\lambda})}' + [I_N - \Psi^{(j)}(e^{-i\lambda})] \, \Sigma^{(j)} \right) / {|\delta^{(j)}(e^{-i\lambda})|}^2 \text{ if } \lambda = \lambda^{(k)}, k \neq j \\ \Sigma^{(j)} \, f_{\underline{x}}(\lambda)^{-1} \, G^{(j)}(\lambda) \quad \text{else.} \end{cases}$$

For the band-pass and high-pass filters similar expressions can be computed.

Disclaimer

This paper is released to inform interested parties of research and to encourage discussion. The views expressed on statistical issues are those of the author and not those of the U.S. Census Bureau. The statistics in this paper were approved for dissemination by the Disclosure Review Board (release number: CBDRB-FY21-CED004-001).

References

[1] Baxter, M. and King, R. (1999) Measuring business cycles: approximate bandpass filters for economic time series. *Review of Economics and Statistics* **81**, 575–593.

[2] Bell, W. (1984) Signal extraction for nonstationary time series. *The Annals of Statistics* **12**, 646–664. MR0740918

[3] Bell, W. and Hillmer, S. (1991) Initializing the Kalman filter for nonstationary time series models. *Journal of Time Series Analysis* **12**(4), 283–300. MR1131002

[4] Bujosa, M., Bujosa, A. and Garcı, A. (2015) Mathematical framework for pseudo-spectra of linear stochastic difference equations. *IEEE Transactions on Signal Processing* **63**(24), 6498–6509. MR3429334

[5] Christiano, L.J. and Fitzgerald, T.J. (2003) The band pass filter. *International economic review* **44**(2), 435–465.

[6] Dagum, E.B. and Bianconcini, S. (2008) The Henderson smoother in reproducing kernel Hilbert space. *Journal of Business & Economic Statistics* **26**(4), 536–545. MR2459350

[7] Dagum, E.B. and Luati, A. (2002) Global and local statistical properties of fixed-length nonparametric smoothers. *Statistical Methods and Applications* **11**(3), 313–333.

[8] Dagum, E. and Luati, A. (2012) Asymmetric filters for trend-cycle estimation. In *Economic Time Series: Modeling and Seasonality*, eds. Bell, W., Holan, S., McElroy, T. CRC Press. Boca Raton, FL. MR3076017

[9] Ferrara, L., Guegan, D. and Rakotomarolahy, P. (2010) GDP nowcasting with ragged-edge data: a semi-parametric modeling. *Journal of Forecasting* **29**(1-2), 186–199. MR2752009

[10] Findley, D. F., B. C. Monsell, W. R. Bell, M. C. Otto, and B. C. Chen (1998) New Capabilities of the X-12-ARIMA Seasonal Adjustment Program (with discussion). *Journal of Business and Economic Statistics* **16**, 127–177.

[11] Franses, P.H. (1994) A multivariate approach to modeling univariate seasonal time series. *Journal of Econometrics* **63**, 133–151.

[12] Gómez, V. (1999) Three equivalent methods for filtering finite nonstationary time series. *Journal of Business & Economic Statistics* **17**(1), 109–116. MR1671010

[13] Gómez, V. (2007) Wiener-Kolmogorov Filtering and Smoothing for Multivariate Series With State–Space Structure. *Journal of Time Series Analysis* **28**(3), 361–385. MR2355313

[14] Gómez, V. (2016) *Multivariate time series with linear state space structure.* New York, NY: Springer. MR3468927

[15] Gómez, V. and Maravall, A. (1994) Estimation, prediction, and interpolation for nonstationary series with the Kalman filter. *Journal of the American Statistical Association* **89**(426), 611–624. MR1294087

[16] Gómez, V., Maravall, A. and Peña, D. (1999) Missing observations in ARIMA models: Skipping approach versus additive outlier approach. *Journal of Econometrics* **88**(2), 341–363. MR1666907

[17] Holan, S., McElroy, T., and Wu, G. (2017) The cepstral model for multivariate time series: the vector exponential model. *Statistica Sinica* **27**, 23–42. MR3618159

[18] Jentsch, C. and Subba Rao, S. (2015) A test for second order stationarity of a multivariate time series. *Journal of Econometrics* **185**(1), 124–161. MR3300340

[19] Marcellino, M. and Schumacher, C. (2010) Factor MIDAS for nowcasting

and forecasting with ragged-edge data: A model comparison for German GDP. *Oxford Bulletin of Economics and Statistics* **72**(4), 518–550.

[20] McElroy, T.S. (2008) Matrix formulas for nonstationary ARIMA signal extraction. *Econometric Theory* **24**, 1–22. MR2521631

[21] McElroy, T.S. (2011) A nonparametric method for asymmetrically extending signal extraction filters. *Journal of Forecasting* **30**, 597–621. MR2861629

[22] McElroy, T.S. (2017) Multivariate seasonal adjustment, economic identities, and seasonal taxonomy. *Journal of Business & Economics Statistics* **35**(4), 611–625. MR3716039

[23] McElroy, T.S. (2018) Recursive computation for block nested covariance matrices. *Journal of Time Series Analysis* **39** (3), 299–312. MR3796521

[24] McElroy, T.S., and Monsell, B.C. (2015) Model estimation, prediction, and signal extraction for nonstationary stock and flow time series observed at mixed frequencies. *Journal of the American Statistical Association* **110**, 1284–1303. MR3420702

[25] McElroy, T.S., Pang, O., and Monsell, B. (2019) Seasonal adjustment subject to lower frequency benchmarks. *2019 Proceedings American Statistical Association [CD-ROM]: Alexandria, VA.*

[26] McElroy, T.S. and Politis, D.N. (2020) *Time Series: A First Course with Bootstrap Starter*. New York: Chapman-Hall.

[27] McElroy, T.S. and Trimbur, T.M. (2015) Signal extraction for nonstationary multivariate time Series with illustrations for trend inflation. *Journal of Time Series Analysis* **36**, 209–227. Also in "Finance and Economics Discussion Series," Federal Reserve Board. 2012–45 MR3316467

[28] Proietti, T. and Luati, A. (2008) Real time estimation in local polynomial regression, with application to trend-cycle analysis. *The Annals of Applied Statistics* **2**(4), 1523–1553. MR2655670

[29] Tiao, G.C. and Grupe, M.R. (1980) Hidden periodic autoregressive-moving average models in time series data. *Biometrika* **67**, 365–373. MR0581732