Improving real-time trend estimates using local parametrisation of local polynomial regression filters

Abstract

This paper studies and compares real-time estimates of the trend cycle component with moving averages based on local polynomial regression. In particular, this theory allows to reproduce Henderson's symmetric and Musgrave's asymmetric filters. This paper describes how they can be extended to include a timeliness criterion to minimise the phase shift. While asymmetric filters are generally parameterised globally, which can be suboptimal around turning points, this paper proposes a procedure for parameterising them locally. An empirical comparison, based on simulated and real data, shows that modelling polynomial trends that are too complex introduces more revisions without reducing the phase shift, and that local parameterisation reduces the delay in detecting turning points and reduces revisions. The paper also shows how these results can be easily reproduced using the R package `rjd3filters` dedicated to the manipulation of moving averages.

Keywords: time series, trend-cycle, seasonal adjustment, turning points, R statistical software.

JEL Classification: E32, E37.

1 Introduction

Analysis of the economic cycle, and in particular the early detection of turning points in a series, is a major topic in the analysis of economic outlook. To this end, economic indicators are generally seasonally adjusted. However, in order to improve their interpretability, it may be necessary to perform additional smoothing in order to reduce noise, and thus analyse the trend-cycle component. By construction, trend-cycle extraction methods are closely related to seasonal adjustment methods, since they are generally applied to seasonally adjusted series.

A moving average, or linear filter, is a statistical method that consists in applying a rolling weighted mean to a times series: for each date t it computes a weighted mean of p past points and q future points where $p, q \geq 0$ depends on the moving average. Moving averages are ubiquitous in business cycle extraction and seasonal adjustment methods. For example, the X-13ARIMA seasonal adjustment method uses several moving averages to estimate the main components of a time series. Symmetric filters are applied to the center of the series, but when it comes to estimate the most recent points, all of these methods must rely on asymmetric filters. For trend-cycle extraction, the most popular symmetric filter is the Henderson (1916) moving average, which is used in the X-13ARIMA seasonal adjustment algorithm.

However, for real-time estimates, due to the lack of future observations, all these methods must rely on asymmetric filters to estimate the most recent points. For example, for trend-cycle extraction, X-13ARIMA uses Henderson's symmetric filter and Musgrave (1964)'s asymmetric filters on an extended series using an ARIMA model. Since the predicted values are linear combinations of past values, this amounts to applying asymmetric averages at the end of the series.

If the classic asymmetric moving averages have good properties regarding the future revisions induced by the process (see for example Pierce (1980)), they create, by construction, phase shifts that impact the real-time estimation of turning points, introducing time delay in the detection.

Using local polynomial regression, Proietti and Luati (2008) developed general class of symmetric and asymmetric moving averages. In particular, it makes possible to reproduce Henderson's symmetric filter and Musgrave's asymmetric filters. However, these methods have two drawbacks. Firstly, the phase shift (i.e. the delay in detecting turning points) is not directly controlled. Secondly, asymmetric filters depend on a parameter that is generally estimated over the whole series using assumptions that may be wrong locally. The aim of this study is to propose extensions to the Proietti and Luati (2008) class of filters in order to take these shortcomings into account. The extensions have been implemented in the R `rjd3filters` package (available at https://github.com/rjdemetra/rjd3filters) and are therefore easily reusable.

In Section 2, we describe the general properties of moving averages and the associated quality criteria. This allows us to understand the foundations behind the construction of moving averages from local polynomial regressions, as well as those behind the two extensions proposed in this article (Section 3). In a final section

(Section 4), all these methods are compared empirically on simulated and real series.

2 Some properties on moving averages

Lots of papers describe the definition and the properties of moving averages and linear filters, see for example Ladiray (2018). In this section we summarize some of the main results to understand the next sections.

Let p et f two integers, a moving average $M_{\boldsymbol\theta}$ or M is defined by a set of coefficients $\boldsymbol\theta = \begin{pmatrix}\theta_{-p} & \cdots & \theta_f\end{pmatrix}^T$ such as for all time series X_t:

$$M_{\boldsymbol\theta}(X_t) = \sum_{k=-p}^{+f} \theta_k X_{t+k}.$$

- $p+f+1$ is called the *moving average order*.
- When $p=f$ the moving average is said to be *centered*. If we also have $\forall k:\ \theta_{-k}=\theta_k$, the moving average $M_{\boldsymbol\theta}$ is said to be *symmetric*. In this case, the quantity $h=p=f$ is called the *bandwidth*.

2.1 Gain and phase shift functions

To interpret the notions of gain and phase shift, it is useful to illustrate the effects of moving averages on harmonic series $X_t = \mathrm{e}^{-i\omega t}$ with $\omega\in[0,\pi]$. The moving average $M_{\boldsymbol\theta}$ transforms X_t into:

$$Y_t = M_{\boldsymbol\theta}X_t = \sum_{k=-p}^{+f}\theta_k\,\mathrm{e}^{-i\omega(t+k)} = \left(\sum_{k=-p}^{+f}\theta_k\,\mathrm{e}^{-i\omega k}\right)\cdot X_t.$$

The function $\Gamma_{\boldsymbol\theta}(\omega)=\sum_{k=-p}^{+f}\theta_k e^{-i\omega k}$ is called the *transfer function* or *frequency response function*. The frequency response function can equivalently be defined as $\Gamma_{\boldsymbol\theta}(\omega)=\sum_{k=-p}^{+f}\theta_k e^{i\omega k}$ ou $\Gamma_{\boldsymbol\theta}(\omega)=\sum_{k=-p}^{+f}\theta_k e^{2\pi i\omega k}$. It can be rewritten as:

$$\Gamma_{\boldsymbol\theta}(\omega)=\rho_{\boldsymbol\theta}(\omega)\,\mathrm{e}^{i\varphi_{\boldsymbol\theta}(\omega)},$$

where $\rho_{\boldsymbol\theta}(\omega)=G_{\boldsymbol\theta}(\omega)=|\Gamma_{\boldsymbol\theta}(\omega)|$ is the *gain* or *amplitude* function and $\varphi_{\boldsymbol\theta}(\omega)$ is the *phase shift* or *time shift* function. This function is sometimes represented as $\phi_{\boldsymbol\theta}(\omega)=\frac{\varphi_{\boldsymbol\theta}(\omega)}{\omega}$ to mesure the phase shift in number of periods. For all symmetric moving average we have $\varphi_{\boldsymbol\theta}(\omega)\equiv 0$ (*modulo* π).

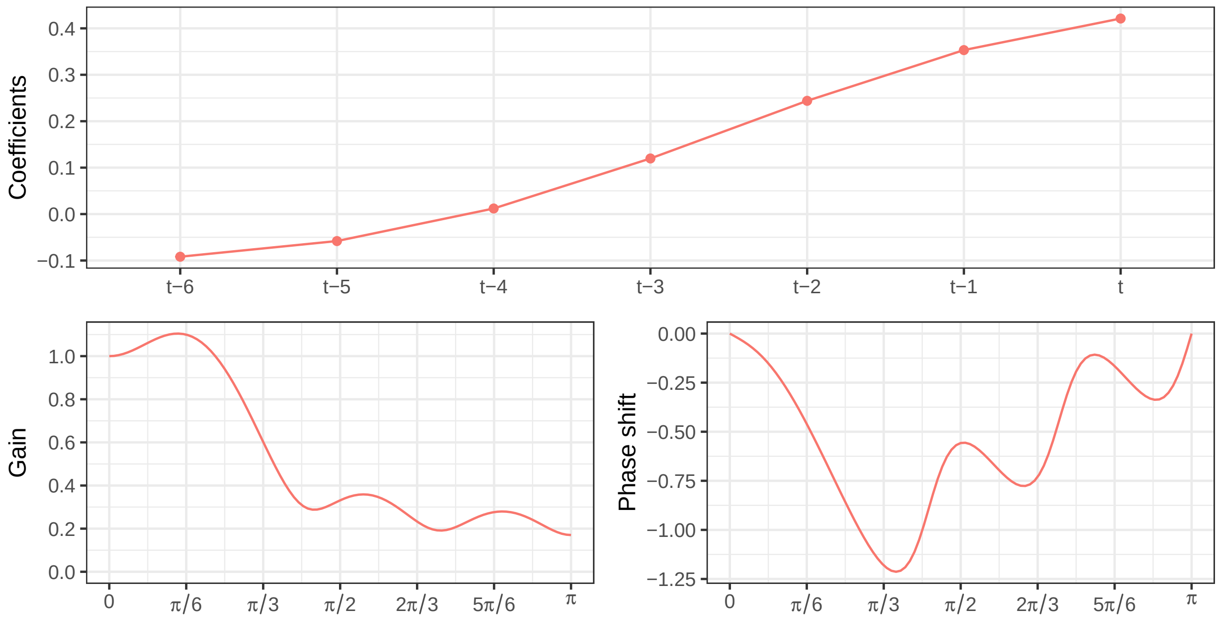
To sum up, applying a moving average to a harmonic times series ($X_t=\mathrm{e}^{-i\omega t}$) affects it in in two different ways:

- by multiplying it by an amplitude coefficient $\rho_{\boldsymbol{\theta}}(\omega)$ (gain);
- by "shifting" it in time by $\varphi_{\boldsymbol{\theta}}(\omega)/\omega$ which directly affects the detection of turning points. When $\varphi_{\boldsymbol{\theta}}(\omega)/\omega < 0$ the time shift is negative: the turning point is detected with delay.

Fourier decomposition allows to analyze any time series as a sum of harmonic series, and each component (trend, cycle, seasonal, irregular) is associated with a set of frequencies. Noting $\omega = 2\pi/p$, the harmonic series of frequency ω represents a series that repeats itself every p periods. For example, for a monthly series (12 observations per year), seasonal movements are those that repeat every year: they are therefore associated with frequencies $2\pi/12$ (annual periodicity), $2\pi/12 \times 2 = 2\pi/6, \dots, 2\pi/12 \times 5$. In this paper, we consider that the trend-cycle is associated with frequencies in the interval $[0, 2\pi/12[$, i.e. movements recurring at least every 12 months. Even if different frequencies are sometimes used (e.g. $[0, 2\pi/36]$ to consider only cycles of at least 36 months), this has no impact on the results of the study. The other frequencies, $[2\pi/12, \pi]$ are associated with the irregular component (undesirable oscillations).

Figure 1 shows the gain and phase shift function for the asymmetric Musgrave filter (see Section 3.2) often used for real-time trend-cycle estimation (i.e. when no point in the future is known). The gain function is greater than 1 on the frequencies associated with the trend-cycle ($[0, 2\pi/12]$): this means that the trend-cycle is well preserved and that short cycles of 1 to 2 years ($[2\pi/24, 2\pi/12]$) are even amplified. However, cycles of 8 to 12 months ($[2\pi/12, 2\pi/8]$), considered undesirable because associated with the irregular, are also amplified: this can lead to the detection of false turning points. At other frequencies, the gain function is less than 1, but always positive: this means that the series smoothed by this moving average will always contain noise, even if it is attenuated. Analysis of the phase shift shows that the shorter the cycles, the greater the phase shift: this means that on series smoothed by this moving average, turning points could be detected at the wrong date.

Figure 1: Coefficients, gain and phase shift function for the Musgrave filter in real time with $I/C = 3.5$.



2.2 Desirable properties of a moving average

To decompose a time series into a seasonal component, a trend-cycle and the irregular, the X-11 decomposition algorithm (used in X-13ARIMA) uses a succession of moving averages, all with specific constraints.

In our case, we assume that our initial series y_t is seasonally adjusted and can be written as the sum of a trend-cycle, TC_t, and an irregular component, I_t:

$$y_t = TC_t + I_t.$$

The aim will be to build moving averages that best preserve the trend-cycle ($M_{\boldsymbol\theta}(TC_t) \simeq TC_t$) and minimise noise ($M_{\boldsymbol\theta}(I_t) \simeq 0$).

2.2.1 Trend preservation

Trend-cycles are generally modelled by local polynomial trends (see Section 3). In order to best preserve the trend-cycles, we try to have moving averages that preserve the polynomial trends. A moving average $M_{\boldsymbol\theta}$ preserves a function of time $f(t)$ if $\forall t: M_{\boldsymbol\theta} f(t) = f(t)$.

We have the following properties for the moving average $M_{\boldsymbol\theta}$:

- To preserve a constant trend $X_t = a$ we need:

$$\forall t : M_{\boldsymbol\theta}(X_t) = \sum_{k=-p}^{+f} \theta_k X_{t+k} = \sum_{k=-p}^{+f} \theta_k a = a \sum_{k=-p}^{+f} \theta_k = a.$$

  The sum of the coefficients of the moving average $\sum_{k=-p}^{+f} \theta_k$ must then be equal to 1.

- To preserve linear trends $X_t = at + b$ we need:

$$\forall t :\ M_{\boldsymbol\theta}(X_t) = \sum_{k=-p}^{+f} \theta_k X_{t+k} = \sum_{k=-p}^{+f} \theta_k [a(t+k)+b] = a \sum_{k=-p}^{+f} k\theta_k + (at+b) \sum_{k=-p}^{+f} \theta_k = at+b.$$

  Which is equivalent to:

$$\sum_{k=-p}^{+f} \theta_k = 1 \quad \text{et} \quad \sum_{k=-p}^{+f} k\theta_k = 0.$$

- In general, $M_{\boldsymbol\theta}$ preserves polynomials of degree d if and only if:

$$\sum_{k=-p}^{+f} \theta_k = 1 \text{ and } \forall j \in [\![1, d]\!] : \sum_{k=-p}^{+f} k^j \theta_k = 0.$$

- If $M_{\boldsymbol\theta}$ is symmetric ($p = f$ and $\theta_{-k} = \theta_k$) and preserves polynomials of degree $2d$ then it also preserves polynomials of degree $2d + 1$.

2.2.2 Noise reduction

All time series are affected by noise, which can blur the extraction of trends and cycles. This is why we try to reduce this noise (by reducing its variance) while preserving the relevant trends. The sum of the squares of the coefficients $\sum_{k=-p}^{+f} \theta_k^2$ is the *reduction of variance* ratio.

Indeed, let $\{\varepsilon_t\}$ a sequence of independent random variables with $\mathbb{E}[\varepsilon_t] = 0$, $\mathbb{V}[\varepsilon_t] = \sigma^2$:

$$\mathbb{V}[M_{\boldsymbol\theta}\varepsilon_t] = \mathbb{V}\left[\sum_{k=-p}^{+f} \theta_k \varepsilon_{t+k}\right] = \sum_{k=-p}^{+f} \theta_k^2 \mathbb{V}[\varepsilon_{t+k}] = \sigma^2 \sum_{k=-p}^{+f} \theta_k^2.$$

2.3 Real-time estimation and asymmetric moving average

2.3.1 Asymmetric moving averages and forecasts

For symmetric filters, the phase shift function is equal to zero (modulo π). So there is no delay in detecting turning points. However, they cannot be used in the beginning and in the end of the time series because no past or future value can be used. Thus, for real-time estimation, it is needed to build asymmetric moving average that approximate the symmetric moving average.

Several approaches can be used for real-time estimation:

1. Use asymmetric moving averages to take account of the lack of available data;

2. Apply symmetrical filters to extended forecast series. This method seems to date back to De Forest (1877), which also suggests modelling a polynomial trend of degree three or less at the end of the period: "*As the first m and last m terms of the series cannot be reached directly by the formula, the series should be graphically extended by m terms at both ends, first plotting the observations on paper as ordinates, and then extending the curve along what seems to be its probable course, and measuring the ordinates of the extended portions. It is not necessary that this extension should coincide with what would be the true course of the curve in those parts. The important point is that the m terms thus added, taken together with the $m+1$ adjacent given terms, should follow a curve whose form is approximately algebraic and of a degree not higher than the third.*"

   This is also the approach used in the X-13ARIMA seasonal adjustment method, which extends which extends the series by one year with an ARIMA model.

Ultimately, the second method is equivalent to using asymmetric moving averages, since forecasts are linear combinations of the past.

Conversely, the implicit forecasts of an asymmetric moving average can be deduced from a reference symmetric moving average. This allows to judge the quality of real-time estimates of the trend-cycle and to anticipate future revisions when forecasts are far from expected evolutions.

Let us denote $v = (v_{-h}, \dots, v_h)$ the reference symmetrical moving average and $w^0, \dots w^{h-1}$ a sequence of asymmetrical moving averages, of order $h+1$ to $2h$ used to estimate the last h points with, for convention, $w_t^q = 0$ for $t > q$. This means that $w^0 = (w_{-h}^0, \dots, w_0^0)$ is used for estimation in real time (when no points are known in the future), $w^1 = (w^1_{-h}, \dots, w^1_1)$ for estimation of the penultimate point (when only one point is known in the future), and so on. Let also note $y_{-h}, \dots, y_0$ the observed series studied and $y^*_1, \dots y^*_h$ the implicit forecast induced by $w^0, \dots w^{h-1}$. This means that for all q we have:

$$\forall q, \quad \underbrace{\sum_{i=-h}^{0} v_i y_i + \sum_{i=1}^{h} v_i y^*_i}_{\text{smoothing by } v \text{ of the extended series}} = \underbrace{\sum_{i=-h}^{0} w^q_i y_i + \sum_{i=1}^{h} w^q_i y^*_i}_{\text{smoothing by } w^q \text{ of the extended series}}.$$

This is equivalent to:

$$\forall q, \quad \sum_{i=1}^{h} (v_i - w^q_i) y^*_i = \sum_{i=-h}^{0} (w^q_i - v_i) y_i.$$

To sum up, in matrix form, this is equivalent to solving:

$$\begin{pmatrix} v_1 & v_2 & \cdots & v_h \\ v_1 - w^1_1 & v_2 & \cdots & v_h \\ \vdots & \vdots & \cdots & \vdots \\ v_1 - w^{h-1}_1 & v_2 - w^{h-1}_2 & \cdots & v_h \end{pmatrix} \begin{pmatrix} y^*_1 \\ \vdots \\ y^*_h \end{pmatrix} = \begin{pmatrix} w^0_{-h} - v_{-h} & w^0_{-(h-1)} - v_{-(h-1)} & \cdots & w^0_0 - v_0 \\ w^1_{-h} - v_{-h} & w^1_{-(h-1)} - v_{-(h-1)} & \cdots & w^1_0 - v_0 \\ \vdots & \vdots & \cdots & \vdots \\ w^{h-1}_{-h} - v_{-h} & w^{h-1}_{-(h-1)} - v_{-(h-1)} & \cdots & w^{h-1}_0 - v_0 \end{pmatrix} \begin{pmatrix} y_{-h} \\ \vdots \\ y_0 \end{pmatrix}.$$

This is implemented in the `rjd3filters::implicit_forecast()` function.

As highlighted by Wildi and Schips (2004), extending the series through forecasting with an ARIMA model is equivalent to calculating asymmetric filters whose coefficients are optimised in relation to the one-step ahead forecast. In other words, the aim is to minimise the revisions between the first and last estimates (with the symmetric filter). However, the phase shift induced by the asymmetric filters is not controlled: we might prefer to have faster detection of turning points and a larger revision rather than just minimising the revisions between the first and last estimates. Furthermore, since the coefficients of the symmetric filter (and therefore the weight associated with distant forecasts) decrease slowly, we should also be interested in the performance of multi-step ahead forecasting. This is why it may be necessary to define alternative criteria for judging the quality of asymmetric moving averages.

2.3.2 Quality indicators for asymmetric moving averages

For asymmetric filters, most of the criteria come from those defined by Grun-Rehomme et al. (2018) and Wildi and McElroy (2019) to build asymmetric filters. They are summarised in Table 1 and can be calculated

Table 1: Quality criteria for asymmetric filters ($q = 0, 1, 2$) computed by local polynomials using the Henderson kernel with $h = 6$ and $R = 3, 5$.

[!h]

Code	Description	Formula
b_c	Constant bias	$\sum_{k=-p}^{+f} \theta_k - 1$
b_l	Linear bias	$\sum_{k=-p}^{+f} k\theta_k$
b_q	Quadratic bias	$\sum_{k=-p}^{+f} k^2\theta_k$
F_g	Variance reduction / Fidelity (Guggemos)	$\sum_{k=-p}^{+f} \theta_k^2$
S_g	Smoothness (Guggemos)	$\sum_j (\nabla^3\theta_j)^2$
T_g	Timeliness (Guggemos)	$\int_0^{\omega_1} \rho_{\boldsymbol\theta}(\omega)\sin(\varphi_{\boldsymbol\theta}(\omega))^2\,\mathrm{d}\omega$
A_w	Accuracy (Wildi)	$2\int_0^{\omega_1} (\rho_s(\omega) - \rho_{\boldsymbol\theta}(\omega))^2\, h(\omega)\,\mathrm{d}\omega$
T_w	Timeliness (Wildi)	$8\int_0^{\omega_1} \rho_s(\omega)\rho_{\boldsymbol\theta}(\omega)\sin^2\left(\frac{\varphi_s(\omega)-\varphi_{\boldsymbol\theta}(\omega)}{2}\right) h(\omega)\,\mathrm{d}\omega$
S_w	Smoothness (Wildi)	$2\int_{\omega_1}^{\pi} (\rho_s(\omega) - \rho_{\boldsymbol\theta}(\omega))^2\, h(\omega)\,\mathrm{d}\omega$
R_w	Residual (Wildi)	$8\int_{\omega_1}^{\pi} \rho_s(\omega)\rho_{\boldsymbol\theta}(\omega)\sin^2\left(\frac{\varphi_s(\omega)-\varphi_{\boldsymbol\theta}(\omega)}{2}\right) h(\omega)\,\mathrm{d}\omega$

Note: X_g criteria from Grun-Rehomme et al (2018) and X_w criteria from Wildi and McElroy (2019). ρ_s and φ_s represent the gain and phase shift function of the symmetric Henderson filter. h is the spectral density of the input series, considered to be that of white noise ($h_{WN}(x) = 1$) or a random walk ($h_{RW}(x) = \frac{1}{2(1-\cos(x))}$).

using the `rjd3filters::diagnostic_matrix()` function.

Grun-Rehomme et al. (2018) propose a general approach to derive linear filters, based on an optimisation problem involving three criteria: *Fidelity* (F_g), *Smoothness* (S_g) and *Timeliness* (T_g). Fidelity can be directly linked to the reduction in variance created by the filter, and timeliness to the notion of phase shift, which again we want to be low. This method can be used inR with the function `rjd3filters::fst_filter()`.

Wildi and McElroy (2019) propose an approach based on the decomposition of the mean squared error between the symmetric and the asymmetric filter in four quantities: *Accuracy* (A_w), *Timeliness* (T_w), *Smoothness* (S_w) and *Residual* (R_w). A simplified version of this method can be used in R with the function `rjd3filters::dfa_filter()`.

3 Non-parametric regression and local polynomial regression

Many trend-cycle extraction methods are based on non-parametric regressions, which are particularly flexible because they do not assume any predetermined dependency in the predictors. In practice, local regressions can be used. More specifically, consider a set of points $(x_i, y_i)_{1\leq i\leq n}$. Non-parametric regression involves assuming that there exists a function μ, to be estimated, such that $y_i = \mu(x_i) + \varepsilon_i$ with ε_i an error term.

According to Taylor's theorem, for any point x_0, if μ is differentiable d times, then:

$$\forall x \,:\, \mu(x) = \mu(x_0) + \mu'(x_0)(x - x_0) + \dots + \frac{\mu^{(d)}(x_0)}{d!}(x - x_0)^d + R_d(x), \tag{1}$$

where R_d is a negligible residual term in the neighbourhood of x_0. In a neighbourhood $h(x_0)$ around x_0, μ can be approximated by a polynomial of degree d. The quantity $h(x_0)$ is called the *bandwidth*. If ε_i is white noise, then we can estimate by least squares $\mu(x_0)$ using the observations which are in $[x_0 - h(x_0), x_0 + h(x_0)]$.

In practice, this means assuming that the trend is locally polynomial. Various estimation methods can be used to derive symmetrical and asymmetrical moving averages.

Gray and Thomson (1996) propose a complete statistical framework which makes it possible, in particular, to model the error in approximating the trend by local polynomials. However, as the specification of this error is generally complex, simpler models may be preferred, such as that of Proietti and Luati (2008).

Finally, Dagum and Bianconcini (2008) propose a similar modelling of the trend-cycle but using the theory of Hilbert spaces with reproducing kernels for estimation, which has the particular advantage of facilitating the calculation of different moving averages at different time frequencies. This method can be used inR with the function `rjd3filters::rkhs_filter()`.

3.1 Symmetric moving averages and local polynomial regression

Using Proietti and Luati (2008)'s notation, we assume that our time series y_t can be decomposed into

$$y_t = \mu_t + \varepsilon_t,$$

where μ_t is the trend and $\varepsilon_t \overset{i.i.d}{\sim} \mathcal{N}(0, \sigma^2)$ is the noise (the series is therefore seasonally adjusted). Within a neighbourhood h of t, the local trend μ_t is approximated by a polynomial of degree d, such that $\mu_t \simeq m_t$ with:

$$\forall j \in \llbracket -h, h \rrbracket :\ y_{t+j} = m_{t+j} + \varepsilon_{t+j}, \quad m_{t+j} = \sum_{i=0}^{d} \beta_i j^i.$$

The problem of trend extraction is equivalent to estimating $m_t = \beta_0$ (the constant in the previous formula).

In matrix notation:

$$\underbrace{\begin{pmatrix} y_{t-h} \\ y_{t-(h-1)} \\ \vdots \\ y_t \\ \vdots \\ y_{t+(h-1)} \\ y_{t+h} \end{pmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{pmatrix} 1 & -h & h^2 & \cdots & (-h)^d \\ 1 & -(h-1) & (h-1)^2 & \cdots & (-(h-1))^d \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & h-1 & (h-1)^2 & \cdots & (h-1)^d \\ 1 & h & h^2 & \cdots & h^d \end{pmatrix}}_{\boldsymbol{X}} \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \beta_d \end{pmatrix}}_{\boldsymbol{\beta}} + \underbrace{\begin{pmatrix} \varepsilon_{t-h} \\ \varepsilon_{t-(h-1)} \\ \vdots \\ \varepsilon_t \\ \vdots \\ \varepsilon_{t+(h-1)} \\ \varepsilon_{t+h} \end{pmatrix}}_{\boldsymbol{\varepsilon}}.$$

To estimate $\boldsymbol{\beta}$ we need $2h+1 \geq d+1$ and the estimation is made by weighted least squares (WLS), which is equivalent to minimising the following objective function:

$$S(\hat{\beta}_0, \dots, \hat{\beta}_d) = \sum_{j=-h}^{h} \kappa_j (y_{t+j} - \hat{\beta}_0 - \hat{\beta}_1 j - \dots - \hat{\beta}_d j^d)^2.$$

where κ_j is a set of weights called *kernel*. We have $\kappa_j \geq 0 : \kappa_{-j} = \kappa_j$, and with $\boldsymbol{K} = diag(\kappa_{-h}, \dots, \kappa_h)$, the estimate of $\boldsymbol{\beta}$ can be written as $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{K}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{K}\boldsymbol{y}$. With $\boldsymbol{e}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^T$, the estimate of the trend is:

$$\hat{m}_t = \boldsymbol{e}_1 \hat{\boldsymbol{\beta}} = \boldsymbol{\theta}'\boldsymbol{y} = \sum_{j=-h}^{h} \theta_j y_{t-j} \text{ with } \boldsymbol{\theta} = \boldsymbol{K}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{K}\boldsymbol{X})^{-1}\boldsymbol{e}_1. \tag{2}$$

To conclude, the estimate of the trend $\hat{m}_t$ can be obtained applying the symmetric filter $\boldsymbol{\theta}$ to y_t ($\boldsymbol{\theta}$ is symmetric due to the symmetry of the kernel weights κ_j). Moreover, $\boldsymbol{X}'\boldsymbol{\theta} = \boldsymbol{e}_1$ so:

$$\sum_{j=-h}^{h} \theta_j = 1, \quad \forall r \in [\![1, d]\!] : \sum_{j=-h}^{h} j^r \theta_j = 0.$$

Hence, the filter $\boldsymbol{w}$ preserve deterministic polynomial of order d.

Regarding parameter selection, the general consensus is that the choice between different kernels is not crucial. See, for example, Cleveland and Loader (1996) or Loader (1999). The only desired constraints on the kernel are that it assigns greater weight to the central estimation (κ_0) and decreases towards 0 as it moves away from the central estimation. The uniform kernel should therefore be avoided. It is then preferable to focus on two other parameters:

- the degree of the polynomial, denoted by d: if it is too small, there is a risk of biased estimates of the trend-cycle, and if it is too large, there is a risk of excessive variance in the estimates (due to over-adjustment);
- the number of neighbors $2h + 1$ (or the window h): if it is too small, then there will be too little data for the estimates (resulting in high variance in the estimates), and if it is too large, the polynomial approximation will probably be wrong, leading to biased estimates.

In this paper we will use the Henderson kernel:

$$\kappa_j = \left[1 - \frac{j^2}{(h+1)^2}\right]\left[1 - \frac{j^2}{(h+2)^2}\right]\left[1 - \frac{j^2}{(h+3)^2}\right].$$

However, several other kernels are available in `rjd3filters`.

3.2 Asymmetric moving averages and local polynomial regression

As mentioned in Section 2.3.1, for real-time estimation, several approaches can be used:

1. Apply symmetric filters to the series extended by forecasting $\hat{y}_{n+l|n}, l \in [\![1, h]\!]$.
2. Build an asymmetric filter by local polynomial approximation on the available observations (y_t for $t \in [\![n-h, n]\!]$).
3. Build asymmetric filters that minimise the mean squared error of revision under polynomial trend reproduction constraints.

Proietti and Luati (2008) show that the first two approaches are equivalent when forecasts are made by polynomial extrapolation of degree d. They are also equivalent to the third approach under the same constraints as those of the symmetric filter. The third method is called *direct asymmetric filter* (DAF). This method is used for real-time estimation in the STL seasonal adjustment method (*Seasonal-Trend decomposition based on Loess*, see Cleveland et al. (1990)). Although DAF filter estimates are unbiased, this is at the cost of greater variance in the estimates.

To solve the problem of the variance of real-time filter estimates, Proietti and Luati (2008) propose a general method for constructing asymmetric filters that allows a bias-variance trade-off. This is a generalisation of Musgrave (1964)'s asymmetric filters (used in the X-13ARIMA seasonal adjustment algorithm).

The initial series is modelled here as:

$$\boldsymbol{y} = \boldsymbol{U}\boldsymbol{\gamma} + \boldsymbol{Z}\boldsymbol{\delta} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \boldsymbol{D}). \tag{3}$$

where $[\boldsymbol{U}, \boldsymbol{Z}]$ is of full rank and forms a subset of the columns of X. The objective is to find a filter $\boldsymbol{v}$ that minimises the mean square error of revision (to the symmetric filter $\boldsymbol{\theta}$) under certain constraints. These constraints are represented by the matrix $\boldsymbol{U} = \begin{pmatrix} \boldsymbol{U}_p{}^T & \boldsymbol{U}_f{}^T \end{pmatrix}^T$: $\boldsymbol{U}'_p\boldsymbol{v} = \boldsymbol{U}'\boldsymbol{\theta}$ (with $\boldsymbol{U}_p$ the matrix $(h+q+1)\times(d+1)$ which contains the observations of the matrix U known during estimation by the asymmetric filter). The problem is equivalent to finding v which minimises:

$$\varphi(\boldsymbol{v}) = \underbrace{\underbrace{(\boldsymbol{v}-\boldsymbol{\theta}_p)'\boldsymbol{D}_p(\boldsymbol{v}-\boldsymbol{\theta}_p) + \boldsymbol{\theta}'_f\boldsymbol{D}_f\boldsymbol{\theta}_f}_{\text{revision error variance}} + \underbrace{[\boldsymbol{\delta}'(\boldsymbol{Z}'_p\boldsymbol{v} - \boldsymbol{Z}'\boldsymbol{\theta})]^2}_{bias^2}}_{\text{mean square revision error}} + \underbrace{2\boldsymbol{l}'(\boldsymbol{U}'_p\boldsymbol{v} - \boldsymbol{U}'\boldsymbol{\theta})}_{\text{constraints}}. \tag{4}$$

where $\boldsymbol{l}$ is a vector of Lagrange multipliers.

When $\boldsymbol{U} = \boldsymbol{X}$, the constraint is equivalent to preserving polynomials of degree d: we find asymmetric direct filters (DAF) when $\boldsymbol{D} = \boldsymbol{K}^{-1}$.

When $\boldsymbol{U} = \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}^T$, $\boldsymbol{Z} = \begin{pmatrix} -h & \cdots & +h \end{pmatrix}^T$, $\boldsymbol{\delta} = \delta_1$, $\boldsymbol{D} = \sigma^2\boldsymbol{I}$ and when the symmetric filter is the Henderson filter, we find the asymmetric Musgrave filters (using the Henderson kernel). This filter assumes that, for real-time estimation, the data is generated by a linear process and that the asymmetric filters preserve the constants ($\sum v_i = \sum \theta_i = 1$). These asymmetric filters depend on the ratio $|\delta_1/\sigma|$, which, assuming that the trend is linear and the bias constant, can be linked to the I-C ratio $R = \frac{\bar{I}}{\bar{C}} = \frac{\sum |I_t - I_{t-1}|}{\sum |C_t - C_{t-1}|}$: $\delta_1/\sigma = 2/(R\sqrt{\pi})$. This ratio is used in particular in X-13ARIMA to determine the length of the Henderson filter. For monthly data:

- If the ratio is large ($3.5 < R$) then a 23 term filter is used (to remove more noise).
- If the ratio is small ($R < 1$) a 9 term filter is used.

- Otherwise (most of the cases) a 13-term filter is used.

When $\boldsymbol{U}$ corresponds to the first d^*+1 columns of $\boldsymbol{X}$, $d^* < d$, the constraint is to reproduce polynomial trends of degree d^*. This introduces bias but reduces the variance. Thus, Proietti and Luati (2008) proposes three classes of asymmetric filters:

1. *Linear-Constant* (LC): y_t linear ($d=1$) and $\boldsymbol{v}$ preserves constant signals ($d^*=0$). We obtain Musgrave filters when the Henderson kernel is used and the Henderson filter is used as the symmetric filter.

2. *Quadratic-Linear* (QL): y_t quadratic ($d=2$) and $\boldsymbol{v}$ preserves linear signals ($d^*=1$).

3. *Cubic-Quadratic* (CQ): y_t cubic ($d=3$) and $\boldsymbol{v}$ preserves quadratic signals ($d^*=2$).

Appendix A shows the coefficients, gain and phase shift functions of the four asymmetric filters.

Table 2 compares the quality criteria of the different methods using the Henderson filter and $h=6$ (a symmetric filter with 13 terms). For real-time filters ($q=0$), a more complex asymmetrical filter (in terms of polynomial preservation) results in lower timeliness and higher fidelity/smoothness: reduction in phase shift comes at the expense of an increase in variance. This result varies as q increases: for $q=2$, the QL filter has greater timeliness than the LC filter. This surprising result highlights the fact that the phase shift is not controlled by Proietti and Luati (2008)'s approach.

In terms of revision ($A_w + S_w + T_w + R_w$), the LC and QL filters consistently outperform the CQ and DAF filters.

These theoretical properties are consistent with the empirical results observed in Section 4. Revisions are more important for the CQ and DAF filters, which leads to greater variability in estimations and a longer delay in detecting turning points. The globally parameterised QL filter (with R fixed for all asymmetric filters) may result in later detection of turning points than the LC filter. Indeed, to identify a turning point at date t, it is necessary to have knowledge of at least two points after the given date to confirm the reversal in trend.

3.3 Extension with the timeliness criterion

One drawback of the previous method is the lack of control over the phase shift. However, it is possible to improve the modelling by incorporating in Equation 4 the timeliness criterion defined by Grun-Rehomme

Table 2: Quality criteria for asymmetric filters ($q = 0, 1, 2$) computed by local polynomials using the Henderson kernel with $h = 6$ and $R = 3, 5$.

[!h]

Method	b_c	b_l	b_q	F_g	S_g	$T_g \times 10^{-3}$	A_w	S_w	T_w	R_w	EQM_w
$q = 0$											
LC	0	-0.41	-2.16	0.39	1.27	30.34	0.10	0.49	0.41	0.55	1.54
QL	0	0.00	-0.47	0.71	5.15	0.05	0.07	1.89	0.00	0.11	2.07
CQ	0	0.00	0.00	0.91	11.94	0.01	0.02	2.23	0.00	0.10	2.35
DAF	0	0.00	0.00	0.94	14.20	0.00	0.01	2.18	0.00	0.10	2.29
$q = 1$											
LC	0	-0.12	-0.52	0.27	0.43	4.80	0.01	0.12	0.06	0.11	0.30
QL	0	0.00	-0.06	0.29	0.71	0.69	0.00	0.19	0.01	0.04	0.25
CQ	0	0.00	0.00	0.37	0.57	0.16	0.02	0.58	0.00	0.06	0.66
DAF	0	0.00	0.00	0.41	0.37	0.06	0.02	0.76	0.00	0.06	0.84
$q = 2$											
LC	0	0.00	1.08	0.20	0.08	0.35	0.01	0.01	0.00	0.01	0.04
QL	0	0.00	0.03	0.22	0.05	2.08	0.00	0.01	0.02	0.07	0.10
CQ	0	0.00	0.00	0.37	0.66	0.13	0.02	0.56	0.00	0.06	0.64
DAF	0	0.00	0.00	0.40	0.77	0.02	0.02	0.68	0.00	0.05	0.74

Note: With $EQM_w = A_w + S_w + T_w + R_w$.

et al. (2018). This was proposed by Jean Palate, then coded in Java and integrated into `rjd3filters`.

Using the same notation as in Section 3.2, $\boldsymbol{\theta}$ represents the symmetric filter and $\boldsymbol{v}$ the asymmetric filter. With $\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}_p{}^T & \boldsymbol{\theta}_f{}^T \end{pmatrix}^T$ and $\boldsymbol{\theta}_p$ of the same length of $\boldsymbol{v}$, and $\boldsymbol{g} = \boldsymbol{v} - \boldsymbol{\theta}_p$, the timeliness criterion can be expressed as:

$$T_g(\boldsymbol{v}) = \boldsymbol{v}'\boldsymbol{T}\boldsymbol{v} = \boldsymbol{g}'\boldsymbol{T}\boldsymbol{g} + 2\boldsymbol{\theta}_p'\boldsymbol{T}\boldsymbol{g} + \boldsymbol{\theta}_p'\boldsymbol{T}\boldsymbol{\theta}_p \quad \text{with } \boldsymbol{T} \text{ a symmetric matrix.}$$

Furthermore, the objective function φ of Equation 4 can be rewritten:

$$\begin{aligned}
\varphi(\boldsymbol{v}) &= (\boldsymbol{v} - \boldsymbol{\theta}_p)'\boldsymbol{D}_p(\boldsymbol{v} - \boldsymbol{\theta}_p) + \boldsymbol{\theta}_f'\boldsymbol{D}_f\boldsymbol{\theta}_f + [\boldsymbol{\delta}'(\boldsymbol{Z}_p'\boldsymbol{v} - \boldsymbol{Z}'\boldsymbol{\theta})]^2 + 2\boldsymbol{l}'(\boldsymbol{U}_p'\boldsymbol{v} - \boldsymbol{U}'\boldsymbol{\theta}) \\
&= \boldsymbol{g}'\boldsymbol{Q}\boldsymbol{g} - 2\boldsymbol{P}\boldsymbol{g} + 2\boldsymbol{l}'(\boldsymbol{U}_p'\boldsymbol{v} - \boldsymbol{U}'\boldsymbol{\theta}) + \boldsymbol{c} \quad \text{with } \boldsymbol{v} \begin{cases} \boldsymbol{Q} = \boldsymbol{D}_p + \boldsymbol{Z}_p\boldsymbol{\delta}\boldsymbol{\delta}'\boldsymbol{Z}_p' \\ \boldsymbol{P} = \boldsymbol{\theta}_f\boldsymbol{Z}_f\boldsymbol{\delta}\boldsymbol{\delta}'\boldsymbol{Z}_p' \\ \boldsymbol{c} \text{ a constant independent of } \boldsymbol{v} \end{cases}.
\end{aligned}$$

Adding the timeliness criterion:

$$\widetilde{\varphi}(\boldsymbol{v}) = \boldsymbol{g}'\widetilde{\boldsymbol{Q}}\boldsymbol{g} - 2\widetilde{\boldsymbol{P}}\boldsymbol{g} + 2\boldsymbol{l}'(\boldsymbol{U}_p'\boldsymbol{v} - \boldsymbol{U}'\boldsymbol{\theta}) + \widetilde{\boldsymbol{c}} \quad \text{with} \begin{cases} \widetilde{\boldsymbol{Q}} = \boldsymbol{D}_p + \boldsymbol{Z}_p\boldsymbol{\delta}\boldsymbol{\delta}'\boldsymbol{Z}_p' + \alpha_T\boldsymbol{T} \\ \widetilde{\boldsymbol{P}} = \boldsymbol{\theta}_f\boldsymbol{Z}_f\boldsymbol{\delta}\boldsymbol{\delta}'\boldsymbol{Z}_p' - \alpha_T\boldsymbol{\theta}_p\boldsymbol{T} \\ \widetilde{\boldsymbol{c}} \text{ a constant independent of } \boldsymbol{v} \end{cases},$$

where α_T is the weight associated to the timeliness criterion. With $\alpha_T = 0$ we find $\varphi(\boldsymbol{v})$. This extension therefore makes it possible to find all the symmetric and asymmetric filters presented in the previous section but also generalizes the approach of Gray and Thomson (1996). This extension is implemented in the R function `rjd3filters::lp_filter()`.

One drawback is that $T_g(\boldsymbol{v})$ is not normalized: the weight α_T has no economic sense. It is then difficult to calibrate the value of α_T since the value of $|\delta/\sigma|$ must also be defined. That's why in this paper we will not compare this extension to the other methods and we will focus on the local calibration of $|\delta/\sigma|$. However, in future studies, we could imagine a two step calibration: fix the value of $|\delta/\sigma|$ to find α_T by cross-validation and then use this weight with a local parameterisation of the ratio $|\delta/\sigma|$ (as in Section 3.4).

3.4 Local parameterisation of asymmetric filters

Asymmetric filters are usually parameterised globally: $|\delta/\sigma|$ estimated on all the data using the IC-ratio or a cross-validation criterion. However, we might prefer a local parameterisation: ratio $|\delta/\sigma|$ which varies as a function of time. Indeed, although the overall parameterisation is generally valid, assuming a constant value of the $|\delta/\sigma|$ ratio for all asymmetrical filters does not appear relevant for real-time estimation, especially during periods of economic downturn. For example, with the LC method, global parameterisation means assuming that the slope of the trend is constant, whereas during economic downturns it tends towards 0 up to the turning point.

This is what is proposed in this article, with a local parameterisation of the asymmetric filters by estimating δ and σ^2 separately. Although this does not give an unbiased estimator of the ratio $|\delta/\sigma|$, it does allow the main evolutions to be captured, such as the decay towards 0 before a turning point and the growth after the turning point for the LC method:

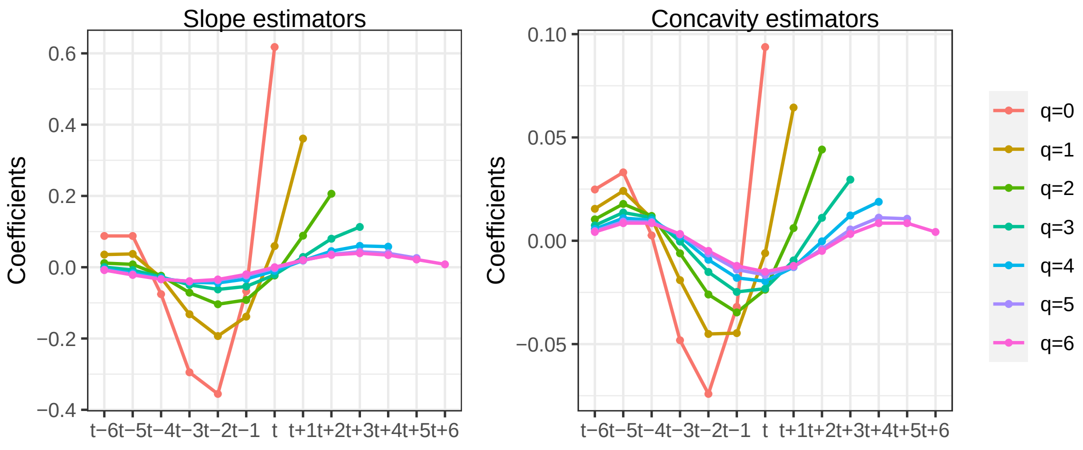
- The variance σ^2 can be estimated using the observed data set and the symmetrical filter $(w_{-p}, \dots, w_p)$:

$$\hat{\sigma}^2 = \frac{1}{n-2h}\sum_{t=h+1}^{n-h}\frac{(y_t - \hat{\mu}_t)^2}{1 - 2w_0^2 + \sum w_i^2}.$$

- The parameter δ can be estimated by moving average from Equation 2. For example, for the LC method we can use the moving average $\boldsymbol{\theta}_2 = \boldsymbol{K}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{K}\boldsymbol{X})^{-1}\boldsymbol{e}_2$ to obtain a local estimate of the slope and for the QL method we can use $\boldsymbol{\theta}_3 = \boldsymbol{K}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{K}\boldsymbol{X})^{-1}\boldsymbol{e}_3$ to obtain a local estimate of the concavity. The DAF method then simply allows us to calculate the associated asymmetric moving averages.

  Although a moving average of different length to that used for estimating the trend could be considered, this seems to degrade the results in terms of phase shift (using the same methodology as in Section 4). Furthermore, for the construction of moving averages, the trend can be modelled as being locally of degree 2 or 3 (this has no impact on the final estimate of concavity). Here, we have chosen to model a trend of degree 2: this reduces the phase shift but slightly increases the revisions linked to the first estimate of the trend-cycle. Figure 2 shows the moving averages used.

Figure 2: Moving averages used for real-time estimation of slope and concavity.



In the empirical applications of Section 4, the final local parameterisation is obtained by estimating δ using all the data (i.e., using the symmetric filters shown in Figure 2), while maintaining a real-time estimate of σ^2. Figure 3 shows an example of the comparison of estimates of the ratio $|\delta_1/\sigma|$ with a global estimator (IC ratio) and two the local parameterisation:

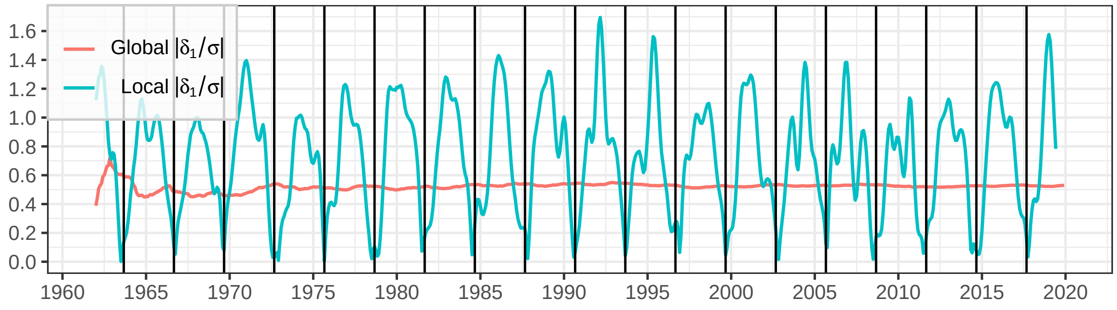
- Figure 3a: using a symmetric filter to estimate δ_1 (final local parameterisation).

- Figure 3b: using the asymmetric filter which need two points in the futur to estimate δ_1. This is closed to the real-time estimates since two points in the future are needed to detect a turning point.

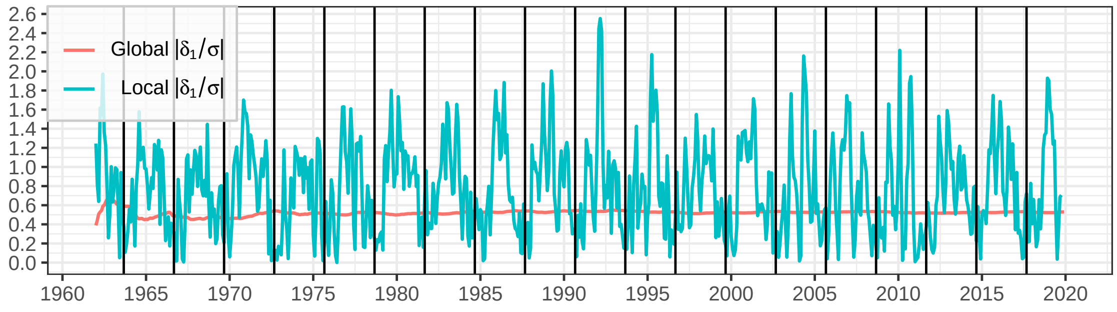
The turning points are clearly detected by the local estimators, but there is much more variance in the estimate of $|\delta_1/\sigma|$ in the real-time estimate. This will lead to another source of revisions in the intermediate estimates.

Figure 3: Comparison of the estimates of $|\delta_1/\sigma|$ with a local estimator and global estimator (with IC ratio) on a simulated series, the vertical lines being the simulated turning points.

(a) Symmetric moving average used for $\hat{\delta}_1$.



(b) Asymmetric moving average using two futur points ($q = 2$) used for $\hat{\delta}_1$.



4 Comparison of different methods

The different methods are compared on simulated and real data.

For all series, a symmetrical 13-term filter is used. These methods are also compared with estimates obtained by extending the series with an ARIMA model, then applying a 13-term symmetrical Henderson filter. The ARIMA model is determined automatically, using no seasonal lag (the series being seasonally adjusted) and no external variables (such as external regressors for correction of atypical points).

The performance of the different methods is judged on criteria relating to revisions (between two consecutive estimates and in relation to the final estimate) and the number of periods required to detect turning points.

4.1 Simulated series

4.1.1 Methodology

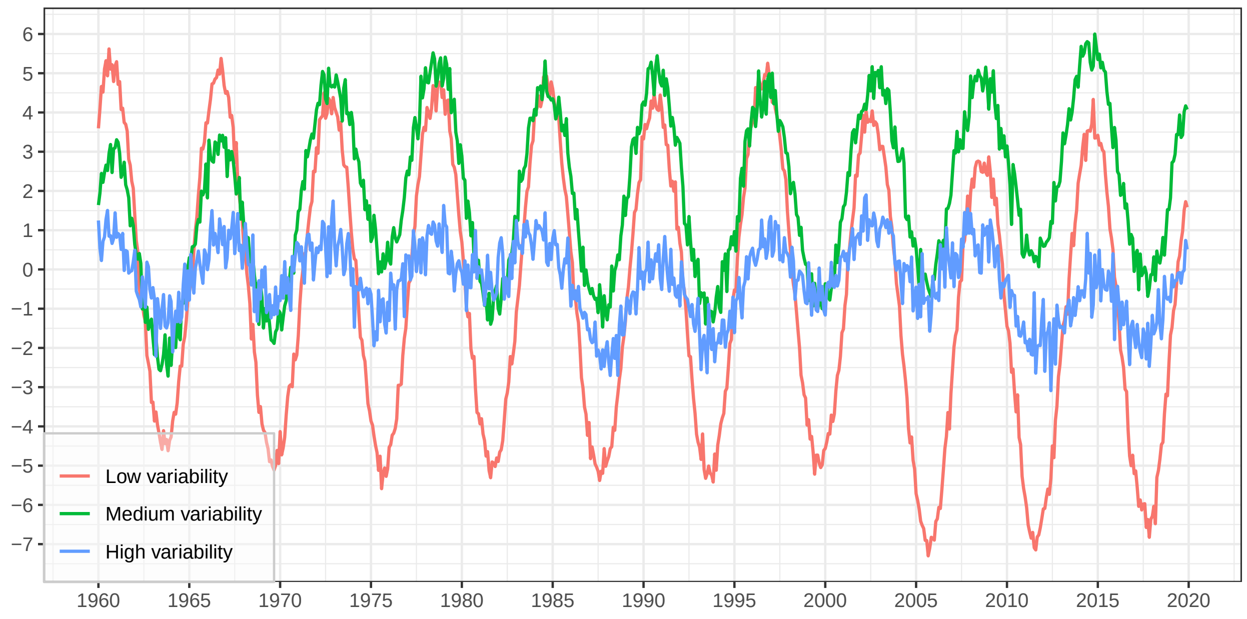
Following a methodology close to that of Darne and Dagum (2009), nine monthly series are simulated between January 1960 and December 2020 with different levels of variability. Each simulated series $y_t = C_t + T_t + I_t$ can be written as the sum of three components:

- the cycle $C_t = \rho[\cos(2\pi t/\lambda) + \sin(2\pi t/\lambda)]$, λ is fixed at 72 (6-year cycles, so there are 19 detectable turning points);
- the trend $T_t = T_{t-1} + \nu_t$ with $\nu_t \sim \mathcal{N}(0, \sigma_\nu^2)$, σ_ν being fixed at 0.08;
- and the irregular $I_t = e_t$ with $e_t \sim \mathcal{N}(0, \sigma_e^2)$.

For the different simulations, we vary the parameters ρ and σ_e^2 in order to obtain series with different signal-to-noise ratios (see Figure 4):

- High signal-to-noise ratio (i.e. low I-C ratio and low variability): $\sigma_e^2 = 0.2$ and $\rho = 3.0, 3.5$ or 4.0 (I-C ratio between 0.9 and 0.7);
- Medium signal-to-noise ratio (i.e. medium I-C ratio and medium variability): $\sigma_e^2 = 0.3$ and $\rho = 1.5$, 2.0 or 3.0 (I-C ratio between 2.3 and 1.4);
- Low signal-to-noise ratio (i.e. high I-C ratio and high variability): $\sigma_e^2 = 0.4$ and $\rho = 0.5$, 0.7 or 1.0 (I-C ratio between 8.9 and 5.2).

Figure 4: Simulated series with low ($\sigma_e^2 = 0.2$ and $\rho = 3,5$), medium ($\sigma_e^2 = 0.3$ and $\rho = 2.0$) and high variability ($\sigma_e^2 = 0.4$ and $\rho = 1.0$).



For each series and each date, the trend-cycle is estimated using the different methods presented in this report. Three quality criteria are also calculated:

1. Computation of the phase shift in the detection of turning points. The definition of Zellner et al. (1991) is used to determine the turning points:

   - An upturn occurs when the economy moves from a phase of recession to a phase of expansion. This is the case at date t when $y_{t-3} \geq y_{t-2} \geq y_{t-1} < y_t \leq y_{t+1}$.

   - A downturn occurs when the economy moves from a phase of expansion to a phase of recession. This is the case at date t when $y_{t-3} \leq y_{t-2} \leq y_{t-1} > y_t \geq y_{t+1}$.

     It thus takes at least 2 months to detect a turning point.

     The phase shift is often defined as the number of months required to detect the right turning point. Here we use a slightly modified criterion: the phase shift is defined as the number of months needed to detect the right turning point without any future revision. Indeed, it can happen that the right turning point is detected by asymmetric filters but is not detected by the final estimate using a symmetric filter (this is the case for 41 reversal points out of the 9 series with asymmetric Musgrave filters) or that there are revisions in successive estimates (this is the case for 7 turning points out of the 9 series with asymmetric Musgrave filters). Finally, relatively few turning points are detected at the right date with the final estimate. With the 13-term Henderson filter, 18 are correctly detected in series with low variability (out of 57 possible), 11 in series with medium variability and 12 in series with high variability.

2. Computation of two revision criteria: the average of the relative deviations between the q^{th} estimate and the last estimate $MAE_{fe}(q)$ and the average of the relative deviations between the q^{th} estimate and the $q+1^{\text{th}}$ estimate $MAE_{qe}(q)$.

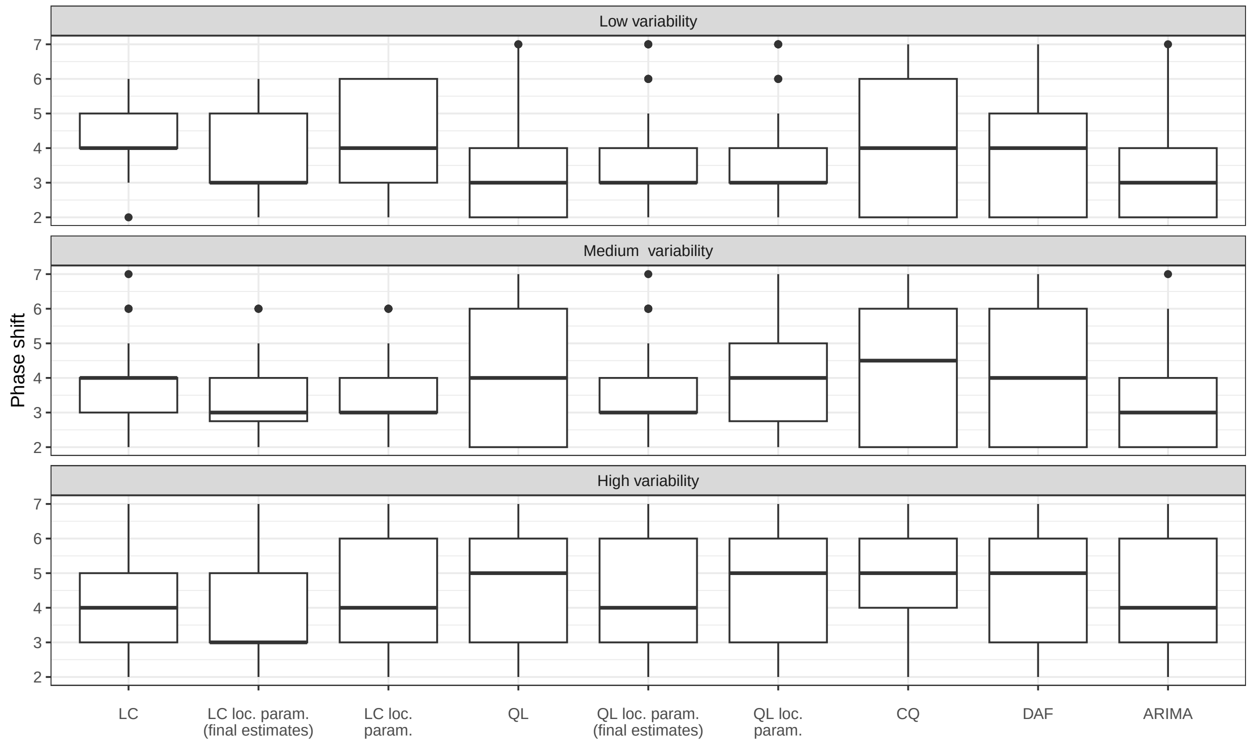
$$MAE_{fe}(q) = \mathbb{E}\left[\left|\frac{y_{t|t+q} - y_{t|last}}{y_{t|last}}\right|\right] \quad \text{and} \quad MAE_{qe}(q) = \mathbb{E}\left[\left|\frac{y_{t|t+q} - y_{t|t+q+1}}{y_{t|t+q+1}}\right|\right].$$

4.2 Comparison

Excluding for the moment the local parameterisations of the polynomial filters, the linear-constant polynomial filter (LC) seems to give the best results in terms of delay in the detection of turning points (Figure 5). Performance is relatively close to that obtained by extending the series using an ARIMA model. However, when variability is low, the LC filter seems to give poorer results and the quadratic-linear polynomial filter (QL) seems to give the best results.

For series with moderate variability, local parameterisation of the LC and QL filters reduces the phase shift. For series with high variability, the phase shift is only reduced by using the final $\hat{\delta}$ parameters: real-time estimates seem to add more variance. For series with low variability, performance seems to be slightly improved only with the LC filter.

Figure 5: Distribution of phase shifts on simulated series.



In terms of revisions, the variability of the series has little effect on the respective performances of the different methods, but it does affect the orders of magnitude, which is why the results are presented only for series with medium variability (Table 3). In general, LC filters always minimise revisions (with relatively small effect from the local parameterisation of the filters) and revisions are greater with cubic-quadratic (CQ) and direct (DAF) polynomial filters.

For the QL filter, there is a large revision between the second and third estimates: this may be due to the fact that for the second estimate (when one point in the future is known), the QL filter assigns a greater weight to the estimate in $t+1$ than to the estimate in t, which creates a discontinuity. This revision is considerably reduced by parameterising the filter locally. For polynomial filters other than the LC filter, the large revisions at the first estimate were to be expected given the coefficient curve: a very large weight is associated with the current observation and there is a strong discontinuity between the moving average used for real-time estimation (when no point in the future is known) and the other moving averages.

Extending the series using an ARIMA model gives revisions with the latest estimates of the same order of magnitude as the LC filter, but slightly larger revisions between consecutive estimates, particularly between

Table 3: Average of the relative deviations of the revisions for the different filters on simulated series with medium variability.

[!h]

Method	$q=0$	$q=1$	$q=2$	$q=3$	$q=4$	$q=5$
$MAE_{fe}(q)=\mathbb{E}\left[\left\|(y_{t\|t+q}-y_{t\|last})/y_{t\|last}\right\|\right]$						
LC	0.21	0.10	0.03	0.03	0.03	0.01
LC local param. (final estimates)	0.19	0.09	0.03	0.03	0.03	0.01
LC local param.	0.29	0.10	0.03	0.03	0.03	0.01
QL	0.33	0.10	0.04	0.04	0.03	0.01
QL local param. (final estimates)	0.21	0.10	0.03	0.03	0.03	0.01
QL local param.	0.30	0.10	0.04	0.03	0.03	0.01
CQ	0.45	0.13	0.13	0.09	0.06	0.02
DAF	0.47	0.15	0.15	0.09	0.06	0.02
ARIMA	0.22	0.10	0.03	0.03	0.03	0.01
$MAE_{ce}(q)=\mathbb{E}\left[\left\|(y_{t\|t+q}-y_{t\|t+q+1})/y_{t\|t+q+1}\right\|\right]$						
LC	0.19	0.10	0.02	0.01	0.07	0.01
LC local param. (final estimates)	0.20	0.10	0.03	0.01	0.05	0.01
LC local param.	0.24	0.11	0.03	0.01	0.05	0.01
QL	0.29	3.46	0.00	0.03	0.04	0.01
QL local param. (final estimates)	0.31	0.11	0.02	0.01	0.04	0.01
QL local param.	0.24	0.16	0.00	0.03	0.04	0.01
CQ	0.43	0.02	0.10	0.07	0.05	0.02
DAF	0.66	0.24	0.11	0.14	0.06	0.02
ARIMA	0.21	0.13	0.02	0.02	0.25	0.01

the fourth and fifth estimates (as might be expected, as highlighted in Section 2.3.1).

4.3 Real series

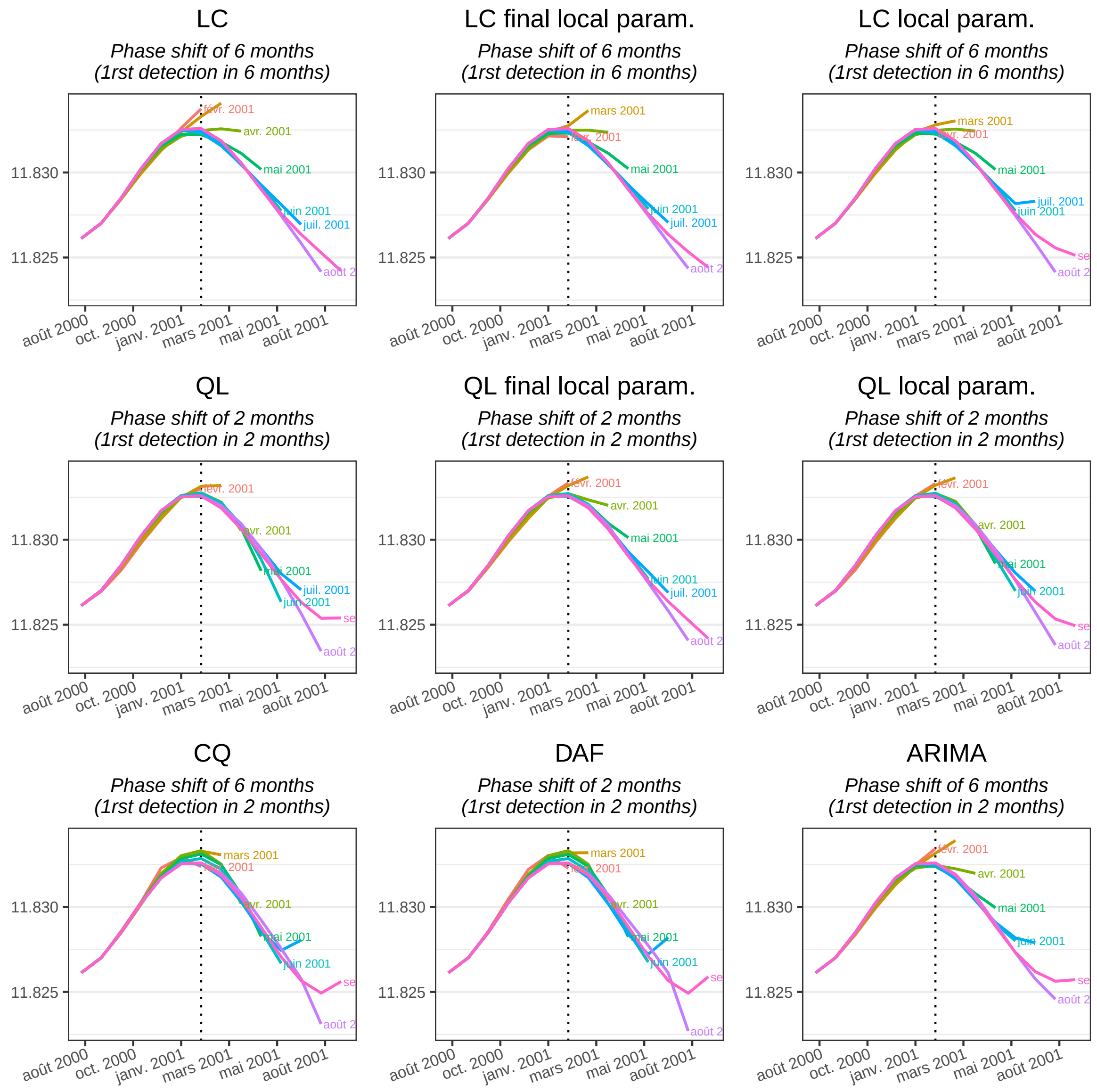
The differences between the methods are also illustrated using an example from the FRED-MD database (McCracken and Ng (2016)) containing economic series on the United States. The series studied correspond to the database published in November 2022. The series studied is the level of employment in the United States (series `CE16OV`, used in logarithm) around the February 2001 turning point, consistent with the monthly dating of the turning points. This point and this series were chosen because the economic turnaround is particularly visible in the raw series (Figure 7), because this series is used for dating economic cycles and because the FRED-MD database facilitates the reproducibility of the results (thanks to the availability of series published on past dates). Studying the series up to January 2020, this series has a medium variability: a symmetrical filter of 13 terms is therefore appropriate. Appendix (**ann-ex-r**) shows how the code to reproduce the different plots.

Figure 6 shows the successive estimates of the trend-cycle using the different methods studied.

For this series, the phase shift is 6 months for the LC and CQ methods and the extension of the series by ARIMA. Except for the LC method (where the turning point is first detected in January 2001), the high phase shift is due to revisions in the turning point detected in the intermediate estimates (even if the correct turning point is detected after 2 months, it is no longer detected after 3 months). It is of two months for the other methods (QL, DAF).

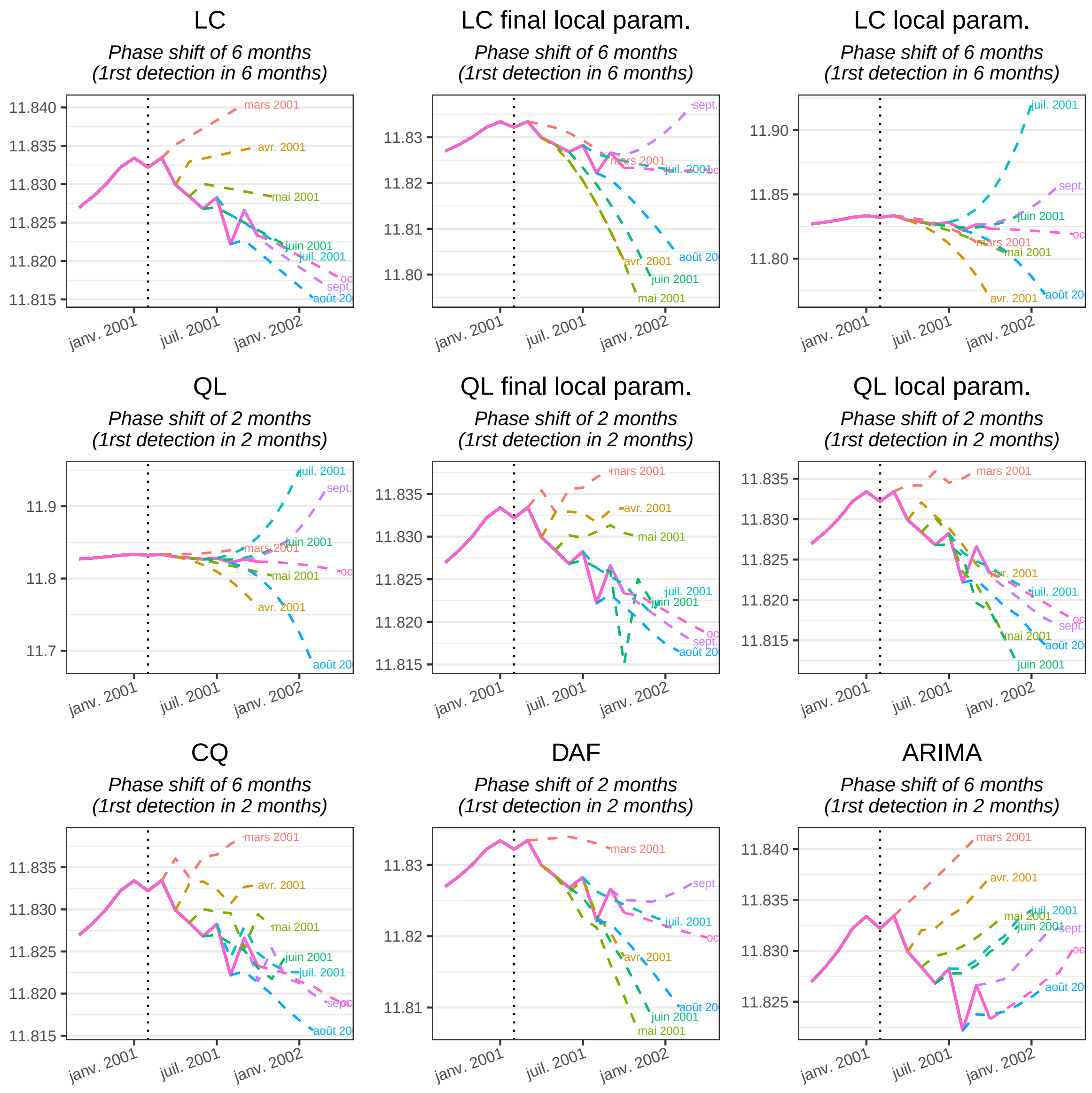
In this example, local parameterisation does not reduce the phase shift, but it does reduce revisions. The QL and DAF polynomials lead to greater variability in the intermediate estimates, especially in February 2001.

Figure 6: Successive estimates of the trend-cycle in US employment (in logarithms).



The quality of the intermediate estimates can also be analysed using the implicit forecasts of the different methods (Figure 7). As a reminder, these are forecasts of the raw series which, by applying the symmetrical Henderson filter to the extended series, give the same estimates as the asymmetrical moving averages. The forecasts of the ARIMA model are naive and do not take the turning point into account, unlike the other methods. Finally, local parameterisation of the QL filter produces much more consistent forecasts.

Figure 7: Implicit forecasts linked to successive estimates of the trend-cycle in US employment (in logarithms) using local polynomial methods.



5 Conclusion

For business cycle analysis, most statisticians use trend-cycle extraction methods, either directly or indirectly. They are used, for example, to reduce the noise of an indicator in order to improve its analysis, and models, such as forecasting models, usually use seasonally adjusted series based on these methods.

This paper presents the R package `rjd3filters`, which implements a several methods to build moving average for real-time trend-cycle estimation. It also offers various functions for the theoretical analysis of moving averages (gain functions, phase shift, quality criteria, etc.) and judging the quality of the latest estimates (for example with implicit forecasts). It also makes it easy to combine moving averages and also to integrate custom moving averages into the X-11 seasonal adjustment algorithm (with the `rjd3filters::x11plus_trend()` function). `rjd3filters` thus facilitates research into the use of moving averages as part of real-time cycle trend estimation. This is the case, for example, with the local parametrisation of Musgrave moving averages and other asymmetric moving averages linked to local polynomial regression, presented in this article.

By comparing the different methods, we can learn a few lessons about the construction of these moving averages.

During economic downturns, asymmetric filters used as an alternative to extending the series using the ARIMA model can reduce revisions to intermediate estimates of the trend-cycle and enable turning points to be detected more quickly.

At the end of the period, modelling polynomial trends of degree greater than three (cubic-quadratic, CQ, and direct, DAF) seems to introduce variance into the estimates (and therefore more revisions) without allowing faster detection of turning points. At the end of the period, for estimating the trend-cycle, we can therefore restrict ourselves to methods modelling polynomial trends of degree two or less (linear-constant, LC, and quadratic-linear, QL). In addition, parameterising polynomial filters locally as suggested in this paper enables turning points to be detected more quickly (especially for the QL filter). Even when the phase shift is not reduced, local parameterisation is recommended because it reduces revisions and produces intermediate estimates that are more consistent with expected future trends. However, with these methods, the length of the filter used must be adapted to the variability of the series: if the filter used is too short (i.e. if the variability of the series is "low"), retaining polynomial trends of degree one or less (LC method) produces poorer results in terms of detecting turning points.

This study could be extended in many ways.

One possible extension would be to look at the impact of filter length on the detection of turning points. Asymmetric filters are calibrated using indicators calculated for the estimation of symmetric filters (for example, to automatically determine their length), whereas a local estimate might be preferable. Furthermore, we have only focused on monthly series with a 13-term symmetric filter, but the results may be different if the symmetric filter studied is longer or shorter and if we study series with other frequencies (quarterly or daily, for example).

Another possibility would be to study the impact of atypical points: moving averages, like any linear operator, are very sensitive to the presence of atypical points. To limit their impact, in X-13ARIMA a strong correction for atypical points is performed on the irregular component before applying the filters to extract the trend-cycle. This leads us to study the impact of these points on the estimation of the trend-cycle and turning points, and also to explore new types of asymmetric filters based on robust methods (such as robust local regressions or moving medians).

References

Cleveland, Robert B., William S. Cleveland, Jean E. McRae, and Irma Terpenning. 1990. "STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion)." *Journal of Official Statistics* 6:3–73. https://www.scb.se/contentassets/ca21efb41fee47d293bbee5bf7be7fb3/stl-a-seasonal-trend-decomposition-procedure-based-on-loess.pdf.

Cleveland, William S., and Clive Loader. 1996. "Smoothing by local regression: Principles and methods." In *Statistical theory and computational aspects of smoothing,* 10–49. Springer.

Dagum, Estela Bee, and Silvia Bianconcini. 2008. "The Henderson Smoother in Reproducing Kernel Hilbert Space." *Journal of Business & Economic Statistics* 26:536–545. https://ideas.repec.org/a/bes/jnlbes/v26y2008p536-545.html.

Darne, Olivier, and Estelle Bee Dagum. 2009. "Performance of short-term trend predictors for current economic analysis." *Economics Bulletin* 29 (1): 79–89. http://www.accessecon.com/Pubs/EB/2009/Volume29/EB-09-V29-I1-P7.pdf.

De Forest, Erastus L. 1877. "On adjustment formulas." *The Analyst* 4 (3): 79–86.

Gray, Alistair, and Peter Thomson. 1996. "Design of Moving-Average Trend Filters using Fidelity and Smoothness Criteria." In *Athens Conference on Applied Probability and Time Series Analysis,* edited by P. M. Robinson and Murray Rosenblatt, 205–219. New York, NY: Springer New York. ISBN: 978-1-4612-2412-9. https://www.census.gov/library/working-papers/1996/adrm/rr96-01.html.

Grun-Rehomme, Michel, Fabien Guggemos, and Dominique Ladiray. 2018. "Asymmetric Moving Averages Minimizing Phase Shift." *Handbook on Seasonal Adjustment,* ec.europa.eu/eurostat/web/products-manuals-and-guidelines/-/KS-GQ-18-001.

Henderson, Robert. 1916. "Note on graduation by adjusted average." *Transactions of the actuarial society of America* 17:43–48.

Ladiray, Dominique. 2018. "Moving Average Based Seasonal Adjustment." *Handbook on Seasonal Adjustment,* ec.europa.eu/eurostat/web/products-manuals-and-guidelines/-/KS-GQ-18-001.

Loader, Clive. 1999. *Local regression and likelihood.* 290. New York: Springer-Verlag. ISBN: 0-387-9877.

McCracken, Michael W., and Serena Ng. 2016. "FRED-MD: A Monthly Database for Macroeconomic Research." *Journal of Business & Economic Statistics* 34 (4): 574–589. doi:10.1080/07350015.2015.1086655.

Musgrave, John C. 1964. "A set of end weights to end all end weights." *US Census Bureau [custodian],* https://www.census.gov/library/working-papers/1964/adrm/musgrave-01.html.

Pierce, David A. 1980. "Data revisions with moving average seasonal adjustment procedures." *Journal of Econometrics* 14, no. 1 (September): 95–114. https://ideas.repec.org/a/eee/econom/v14y1980i1p95-114.html.

Proietti, Tommaso, and Alessandra Luati. 2008. "Real time estimation in local polynomial regression, with application to trend-cycle analysis." *Ann. Appl. Stat.* 2, no. 4 (December): 1523–1553. doi:10.1214/08-AOAS195.

Wildi, Marc, and Tucker McElroy. 2019. "The trilemma between accuracy, timeliness and smoothness in real-time signal extraction." *International Journal of Forecasting* 35 (3): 1072–1084. https://EconPapers.repec.org/RePEc:eee:intfor:v:35:y:2019:i:3:p:1072-1084.

Wildi, Marc, and Bernd Schips. 2004. *Signal Extraction: How (In)efficient Are Model-Based Approaches? An Empirical Study Based on TRAMO/SEATS and Census X-12-ARIMA.* KOF Working papers 04-96. KOF Swiss Economic Institute, ETH Zurich, December. doi:10.3929/ethz-a-004957347.

Zellner, Arnold, Chansik Hong, and Chung-ki Min. 1991. "Forecasting turning points in international output growth rates using Bayesian exponentially weighted autoregression, time-varying parameter, and pooling techniques." *Journal of Econometrics* 49 (1-2): 275–304. https://EconPapers.repec.org/RePEc:eee:econom:v:49:y:1991:i:1-2:p:275-304.

A Coefficients, gain and phase shift functions

Figure 8: Coefficients, gain and phase shift functions for the Linear-Constant (LC) filter with $I/C = 3.5$.

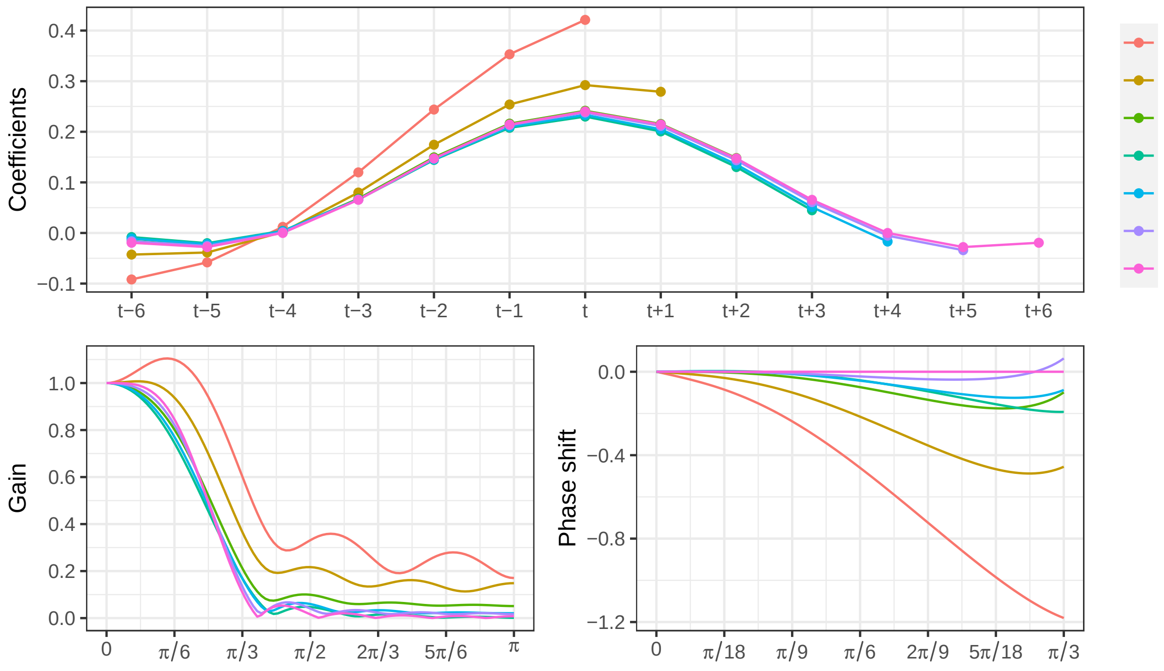


Figure 9: Coefficients, gain and phase shift functions for the Quadratic-Linear (QL) filter with $I/C = 3.5$.

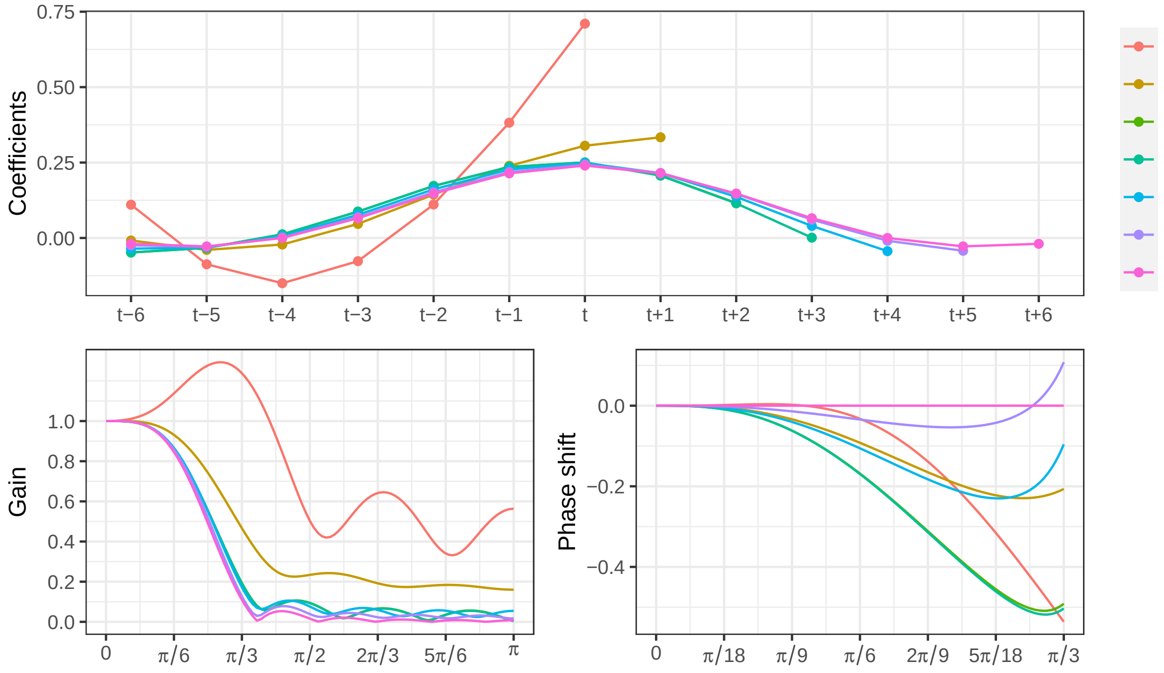


Figure 10: Coefficients, gain and phase shift functions for the Cubic-Quadratic (CQ) filter with $I/C = 3.5$.

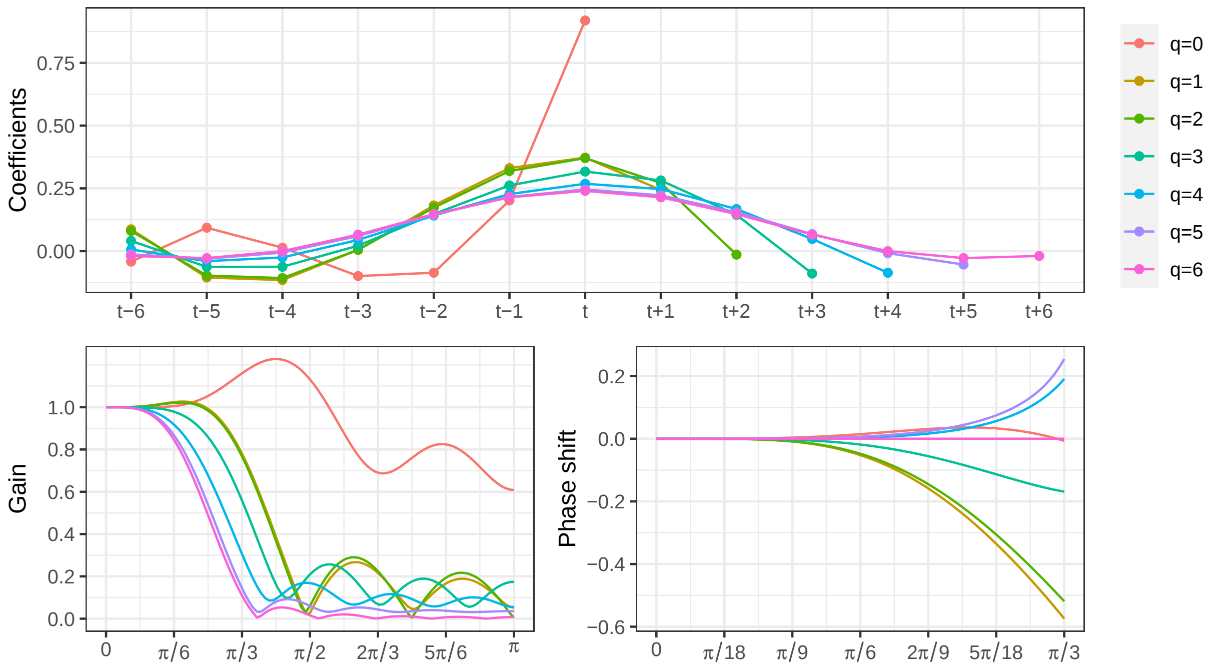
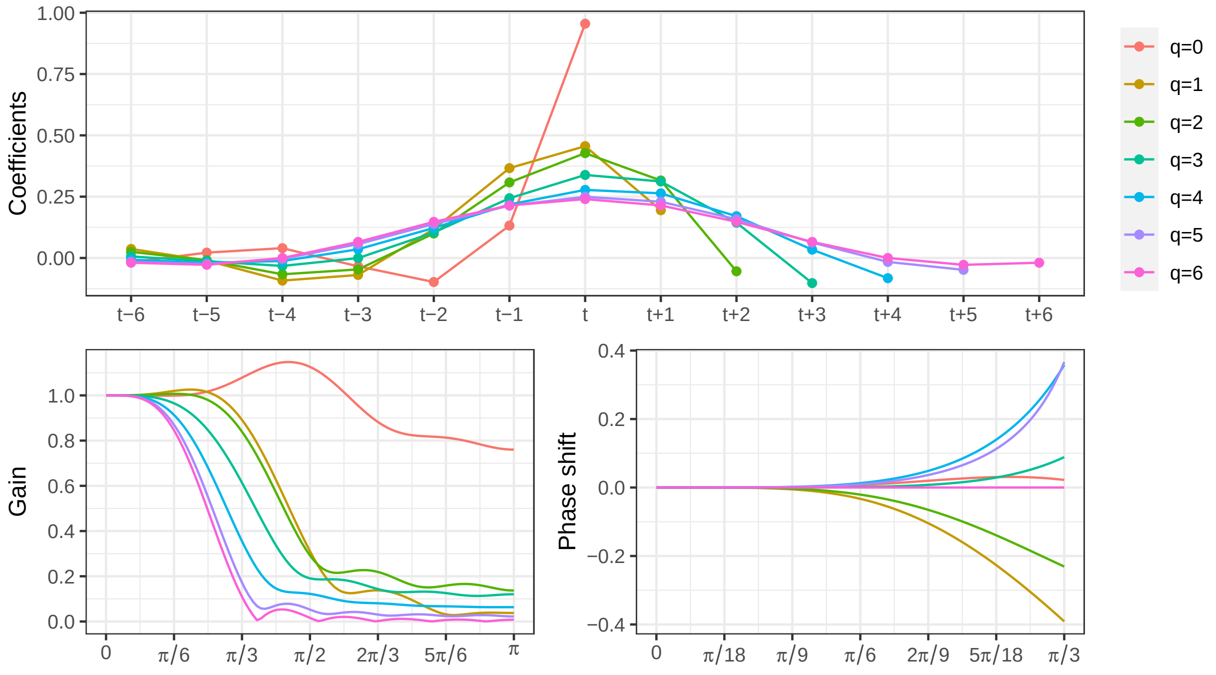


Figure 11: Coefficients, gain and phase shift functions for the direct asymmetric filter (DAF).



B R code example

In this appendix we present the R code that can be used to estimate trend-cycle components for the US Employment with local polynomial filters.

```r
# # To install rjd3filters
# remotes::install_github("rjdemetra/rjd3toolkit")
# remotes::install_github("rjdemetra/rjd3x11plus")
# remotes::install_github("rjdemetra/rjd3filters")
library(rjd3filters)
library(ggplot2)
library(patchwork)
library(zoo)
library(forecast)
file <-"https://files.stlouisfed.org/files/htdocs/fred-md/monthly/2022-11.csv"
data <- read.csv(file)
data <- data[-1,] # First line removed: it contains the transformation codes
y <- ts(log(data[, "CE16OV"]), # We study US employment
         start = 1959, frequency = 12)
y <- window(y, end = c(2001, 9))
last_dates <- c(tail(time(y), 8))
names(last_dates) <- as.character(zoo::as.yearmon(last_dates))
last_est <- lapply(last_dates, function(x) window(y, end = x))


# MA of length 13 is appropriated:
select_trend_filter(y)
# Final estimates:
tc_f <- henderson(y, length = 13, musgrave = FALSE)
plot(y)
lines(tc_f, col = "red")
```

```r
forecast::autoplot(ts.union(y, tc_f))


# IC-ratio computation
icr <- sapply(last_est, function(x) {
  ic_ratio(x, henderson(x, length = 13, musgrave = FALSE))
})
lp_est <- lapply(c("LC", "QL", "CQ", "DAF"), function(method) {
  res <- lapply(seq_along(icr), function(i) {
    lp_coef <- lp_filter(horizon = 6,
                         kernel = "Henderson",
                         endpoints = method,
                         ic = icr[i])
    rjd3filters::filter(last_est[[i]], lp_coef)
  })
  names(res) <- names(last_est)
  res
})
lp_if <- lapply(c("LC", "QL", "CQ", "DAF"), function(method) {
  res <- lapply(seq_along(icr), function(i) {
    lp_coef <- lp_filter(horizon = 6,
                         kernel = "Henderson",
                         endpoints = method,
                         ic = icr[i])
    implicit_forecast(last_est[[i]], lp_coef)
  })
  names(res) <- names(last_est)
  res
})
```

```r
names(lp_est) <- names(lp_if) <-  c("LC", "QL", "CQ", "DAF")


# Local estimates of IC-ratios

# We replicate the direct estimates to have

# estimators of the slope and the concavity

gen_MM <- function(p=6, q=p, d=2){

  X_gen <- function(d = 1, p = 6, q = p){

    sapply(0:d, function(exp) seq(-p, q)^exp)

  }

  k = rjd3filters::get_kernel("Henderson", h = p)

  k = c(rev(k$coef[-1]), k$coef[seq(0,q)+1])

  K = diag(k)

  X = X_gen(d=d, p = p, q = q)

  e1 = e2 = e3 = matrix(0, ncol = 1, nrow = d+1)

  e1[1] = 1

  e2[2] = 1

  e3[3] = 1

  # Estimator of the constant

  M1 = K %*% X %*% solve(t(X) %*% K %*% X, e1)

  # Estimator of the slope

  M2 = K %*% X %*% solve(t(X) %*% K %*% X, e2)

  # Estimor of the concavity

  M3 = K %*% X %*% solve(t(X) %*% K %*% X, e3)

  mm <- list(const = M1, slope = M2, concav = M3)

  lapply(mm, moving_average, lags = -p)

}

all_mm <- lapply(6:0, gen_MM, p = 6, d = 2)

est_slope <- finite_filters(all_mm[[1]]$slope,
                            lapply(all_mm[-1], `[[`, "slope"))
est_concav <- finite_filters(all_mm[[1]]$concav,
                             lapply(all_mm[-1], `[[`, "concav"))


henderson_f <- lp_filter(h=6)@sfilter
lp_filter2 <- function(icr, method = "LC", h = 6, kernel = "Henderson"){
  all_coef = lapply(icr, function(ic){
    lp_filter(horizon = h,
              kernel = kernel,
              endpoints = method,
              ic = ic)
  })
  rfilters = lapply(1:h, function(i){
    q=h -i
    all_coef[[i]][,sprintf("q=%i", q)]
  })
  finite_filters(henderson_f, rfilters = rfilters)
}
loc_lc_est <-
  lapply(last_est, function(x) {
    est_loc_slope <- c(tail(est_slope * x, 6))
    sigma2 <- var_estimator(x, henderson_f)
    icr = 2/(sqrt(pi) * (est_loc_slope / sqrt(sigma2)))
    lp_coef = lp_filter2(ic = icr,
                         method = "LC", h = 6, kernel = "Henderson")
    rjd3filters::filter(x, lp_coef)
  })
loc_lc_if <-
```

```
  lapply(last_est, function(x) {
    est_loc_slope <- c(tail(est_slope * x, 6))
    sigma2 <- var_estimator(x, henderson_f)
    icr = 2/(sqrt(pi) * (est_loc_slope / sqrt(sigma2)))
    lp_coef = lp_filter2(ic = icr,
                         method = "LC", h = 6, kernel = "Henderson")
    implicit_forecast(x, lp_coef)
  })
loc_ql_est <-
  lapply(last_est, function(x) {
    est_loc_concav <- c(tail(est_concav * x, 6))
    sigma2 <- var_estimator(x, henderson_f)
    icr = 2/(sqrt(pi) * (est_loc_concav / sqrt(sigma2)))
    lp_coef = lp_filter2(ic = icr,
                         method = "QL", h = 6, kernel = "Henderson")
    rjd3filters::filter(x, lp_coef)
  })
loc_ql_if <-
  lapply(last_est, function(x) {
    est_loc_concav <- c(tail(est_concav * x, 6))
    sigma2 <- var_estimator(x, henderson_f)
    icr = 2/(sqrt(pi) * (est_loc_concav / sqrt(sigma2)))
    lp_coef = lp_filter2(ic = icr,
                         method = "QL", h = 6, kernel = "Henderson")
    implicit_forecast(x, lp_coef)
  })


## Plots
```

```r
# Plots with all the estimates
plot_est <- function(data, nperiod = 6) {
  joint_data <- do.call(ts.union, data)
  joint_data <-
    window(joint_data,
           start = last_dates[1] - nperiod * deltat(joint_data))

  data_legend <-
    data.frame(x = last_dates,
               y = sapply(data, tail, 1),
               label = colnames(joint_data))

  forecast::autoplot(joint_data) + theme_bw() +
    scale_x_continuous(labels = zoo::as.yearmon) +
    geom_text(aes(x = x, y = y, label = label, colour = label),
              data = data_legend,
              check_overlap = TRUE, hjust = 0, nudge_x = 0.01,
              size = 2, inherit.aes = FALSE) +
    theme(legend.position = "none")  +
    labs(x = NULL, y = NULL)
}

plot_prevs <- function (data, nperiod = 6) {
  joint_data <- do.call(ts.union, lapply(data, function(x) {
    first_date <- time(x)[1] - deltat(x)
    # The last observed data is added for readability
    ts(c(window(y, start = first_date, end = first_date), x),
       start = first_date, frequency = frequency(x))
  }))

  data_legend <-
    data.frame(x = sapply(data, function(x) tail(time(x), 1)),
               y = sapply(data, tail, 1),
               label = colnames(joint_data))
  forecast::autoplot(joint_data, linetype = "dashed") +
    forecast::autolayer(
      window(y, start = last_dates[1] - nperiod * deltat(y)),
      colour = FALSE
    ) +
    theme_bw() +
    scale_x_continuous(labels = zoo::as.yearmon) +
    geom_text(aes(x = x, y = y, label = label, colour = label),
              data = data_legend,
              check_overlap = TRUE, hjust = 0, nudge_x = 0.01,
              size = 2, inherit.aes = FALSE) +
    theme(legend.position = "none") +
    labs(x = NULL, y = NULL)
}

all_est <- c(lp_est, list("LC local param." = loc_lc_est),
             list("QL local param." = loc_ql_est))
all_if <- c(lp_if, list("LC local param." = loc_lc_if),
            list("QL local param." = loc_ql_if))
y_lim <- NULL
all_plots_est <- lapply(
  names(all_est),
```

```r
  function(x) plot_est(all_est[[x]]) +
    ggtitle(latex2exp::TeX(sprintf("Trend-cycle with %s", x))) +
    coord_cartesian(ylim = y_lim)
)
all_plots_prev <- lapply(
  names(all_if),
  function(x) plot_prevs(all_if[[x]]) +
    ggtitle(latex2exp::TeX(sprintf("Implicit forecasts with %s", x)))
)
# Combine all plots:
wrap_plots(all_plots_est, ncol = 2)
wrap_plots(all_plots_prev, ncol = 2)
```