



# Asymmetric Linear Filters for Seasonal Adjustment Applications to the COVID-19

ALAIN QUARTIER-LA-TENTE (LEMNA, INSEE)

Ongoing research under supervision of:

Dominique Ladiray (Independent) and Olivier Darné (Lemna)

With the help of Jean Palate (NBB)

# Introduction (1/2)

---

Ongoing research real-time detection of turning points with linear filters

- ➔ Under supervision of Dominique Ladiray and Olivier Darné
- ➔ Help of Jean Palate

# Introduction (1/2)

---

Ongoing research real-time detection of turning points with linear filters

➔ Under supervision of Dominique Ladiray and Olivier Darné

➔ Help of Jean Palate

In this presentation:

- First results comparing 4 methods for trend-cycle extraction:
  - Current X-13ARIMA algorithm (Henderson filter)
  - Local Polynomial filters (Proietti and Luati (2008))
  - Fidelity-Smoothness-Timeliness approach (Grun-Rehonne, Guggemos, and Ladiray (2018))
  - RKHS filters (Dagum and Bianconcini (2008))

# Introduction (1/2)


---

Ongoing research real-time detection of turning points with linear filters

➔ Under supervision of Dominique Ladiray and Olivier Darné

➔ Help of Jean Palate

In this presentation:

- First results comparing 4 methods for trend-cycle extraction:
  - Current X-13ARIMA algorithm (Henderson filter)
  - Local Polynomial filters (Proietti and Luati (2008))
  - Fidelity-Smoothness-Timeliness approach (Grun-Rehonne, Guggemos, and Ladiray (2018))
  - RKHS filters (Dagum and Bianconcini (2008))
-  package `rjdfilters` (<https://github.com/palatej/rjdfilters>, development version: <https://github.com/AQLT/rjdfilters>)

## Introduction (2/2)

---

A raw time series can be decompose as (additive decomposition):

$$X_t = \underbrace{TC_t}_{\text{trend-cycle}} + \underbrace{S_t}_{\text{seasonality}} + \underbrace{I_t}_{\text{irregular}}$$

And  $TC_t$  generally estimated on a series *without* seasonality

## Introduction (2/2)

---

A raw time series can be decompose as (additive decomposition):

$$X_t = \underbrace{TC_t}_{\text{trend-cycle}} + \underbrace{S_t}_{\text{seasonality}} + \underbrace{I_t}_{\text{irregular}}$$

And  $TC_t$  generally estimated on a series *without* seasonality

*Moving averages* (or *linear filters*) are ubiquitous in trend-cycle extraction and seasonal adjustment (e.g.: X-13-ARIMA):

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

## Introduction (2/2)

A raw time series can be decompose as (additive decomposition):

$$X_t = \underbrace{TC_t}_{\text{trend-cycle}} + \underbrace{S_t}_{\text{seasonality}} + \underbrace{I_t}_{\text{irregular}}$$

And  $TC_t$  generally estimated on a series *without* seasonality

*Moving averages* (or *linear filters*) are ubiquitous in trend-cycle extraction and seasonal adjustment (e.g.: X-13-ARIMA):

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

➡ In general, *symmetric* moving averages ( $p = f$  et  $\theta_{-i} = \theta_i$ )

# Introduction (2/2)

A raw time series can be decompose as (additive decomposition):

$$X_t = \underbrace{TC_t}_{\text{trend-cycle}} + \underbrace{S_t}_{\text{seasonality}} + \underbrace{I_t}_{\text{irregular}}$$

And  $TC_t$  generally estimated on a series *without* seasonality

*Moving averages* (or *linear filters*) are ubiquitous in trend-cycle extraction and seasonal adjustment (e.g.: X-13-ARIMA):

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

- ➡ In general, *symmetric* moving averages ( $p = f$  et  $\theta_{-i} = \theta_i$ )
- ➡ For *real-time estimates*, we must rely on *asymmetric* filters ( $p > f$ ): revisions and delay in turning points detection (*phase-shift*): this is the case of the COVID-19



## Introduction (2/2)

A raw time series can be decompose as (additive decomposition):

$$X_t = \underbrace{TC_t}_{\text{trend-cycle}} + \underbrace{S_t}_{\text{seasonality}} + \underbrace{I_t}_{\text{irregular}}$$

And  $TC_t$  generally estimated on a series *without* seasonality

*Moving averages* (or *linear filters*) are ubiquitous in trend-cycle extraction and seasonal adjustment (e.g.: X-13-ARIMA):

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

- ➡ In general, *symmetric* moving averages ( $p = f$  et  $\theta_{-i} = \theta_i$ )
- ➡ For *real-time estimates*, we must rely on *asymmetric* filters ( $p > f$ ): revisions and delay in turning points detection (*phase-shift*): this is the case of the COVID-19
- ➡ Comparison of 3 methods that could be included in X-13-ARIMA

# Contents

---

## 1. Introduction

## 2. Description of the methods

### 2.1 Current approache

### 2.2 Local Polynomials

### 2.3 Linear Filters and Reproducing Kernel Hilbert Space (RKHS)

### 2.4 Minimization under constraints: FST approach

## 3. Comparison of the methods

## 4. Conclusion

# X-13-ARIMA

---

1. Series extend over 1 year by ARIMA model
2. Trend-Cycle component extracted using symmetric **Henderson** moving average

# X-13-ARIMA

---

1. Series extend over 1 year by ARIMA model
  2. Trend-Cycle component extracted using symmetric **Henderson** moving average
- ➡ Forecasts linear combinations of past values: equivalent to the use of asymmetric filters

## X-13-ARIMA

---

1. Series extend over 1 year by ARIMA model
  2. Trend-Cycle component extracted using symmetric **Henderson** moving average
- ➡ Forecasts linear combinations of past values: equivalent to the use of asymmetric filters
  - ➡ X-11: iteratively decomposes  $X_T$  in  $TC_t$ ,  $S_t$  and  $I_t$  with automatic outlier correction

## X-13-ARIMA

---

1. Series extend over 1 year by ARIMA model
  2. Trend-Cycle component extracted using symmetric **Henderson** moving average
- ➡ Forecasts linear combinations of past values: equivalent to the use of asymmetric filters
  - ➡ X-11: iteratively decomposes  $X_T$  in  $TC_t$ ,  $S_t$  and  $I_t$  with automatic outlier correction
  - ➡ Comparison of 3 alternatives modern approaches that can reproduce Henderson filter

## Local polynomials: rjdfilters::lp\_filter()

Assumption:  $y_t = \mu_t + \varepsilon_t$  with  $\varepsilon_t \stackrel{i.i.d}{\sim} \mathcal{N}(0, \sigma^2)$

$\mu_t$  locally approximated by polynomial of degree  $d$ :

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

## Local polynomials: `R` `rjdfilters::lp_filter()`

Assumption:  $y_t = \mu_t + \varepsilon_t$  with  $\varepsilon_t \stackrel{i.i.d}{\sim} \mathcal{N}(0, \sigma^2)$

$\mu_t$  locally approximated by polynomial of degree  $d$ :

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

Estimation using WLS using *kernels*:  $\hat{\beta} = (X' K X)^{-1} X' K y$  and


$$\hat{m}_t = \hat{\beta}_0 = w' y = \sum_{j=-h}^h w_j y_{t-j} \Rightarrow \text{equivalent to symmetric moving average}$$

$\Rightarrow$  Henderson filter using a specific kernel and  $d = 3$ .



## Asymmetric filters: `rjdfilters::lp_filter()`

Several solutions:

1. Same method with less data (DAF)  $\iff$  Minimize revisions under same polynomial constraints (reproduce cubic trend)
-  **no bias** lots of variance

## Asymmetric filters: `rjdfilters::lp_filter()`

Several solutions:

1. Same method with less data (DAF)  $\iff$  Minimize revisions under same polynomial constraints (reproduce cubic trend)

➡ **no bias** lots of variance

2. Minimization of revisions filter under polynomial constraints:

2.1 *Linear-Constant* (LC):  $y_t$  linear and  $v$  reproduce constant trends (*Musgrave* filter)

2.2 *Quadratic-Linear* (QL):  $y_t$  quadratic and  $v$  reproduce linear trends

2.3 *Cubic-Quadratic* (CQ):  $y_t$  cubic and  $v$  reproduce quadratic trends

➡ Asymmetric filters  $v$  depends on a ratio linked to “IC-Ratio”

## Asymmetric filters: `rjdfilters::lp_filter()`

Several solutions:

1. Same method with less data (DAF)  $\iff$  Minimize revisions under same polynomial constraints (reproduce cubic trend)


 **no bias** lots of variance

2. Minimization of revisions filter under polynomial constraints:

2.1 *Linear-Constant* (LC):  $y_t$  linear and  $v$  reproduce constant trends (*Musgrave* filter)

2.2 *Quadratic-Linear* (QL):  $y_t$  quadratic and  $v$  reproduce linear trends


2.3 *Cubic-Quadratic* (CQ):  $y_t$  cubic and  $v$  reproduce quadratic trends

 Asymmetric filters  $v$  depends on a ratio linked to “IC-Ratio”



simple models with easy interpretation



Timeliness not controlled  method extended in `rjdfilters::lp_filter()`

## RKHS filters: rjdfilters::rkhs\_filter()

---

- RKHS theory used to approximate Henderson filter
- With  $K_p$  the **kernel function**, the symmetric filter:

$$\forall j \in \llbracket -h, h \rrbracket : w_j = \frac{K_p(j/b)}{\sum_{i=-h}^h K_p(i/b)}$$

## RKHS filters: rjdfilters::rkhs\_filter()

- RKHS theory used to approximate Henderson filter
- With  $K_p$  the **kernel function**, the symmetric filter:

$$\forall j \in \llbracket -h, h \rrbracket : w_j = \frac{K_p(j/b)}{\sum_{i=-h}^h K_p(i/b)}$$

- For asymmetric filters:

$$\forall j \in \llbracket -h, q \rrbracket : w_{a,j} = \frac{K_p(j/b)}{\sum_{i=-h}^q K_p(i/b)}$$

## RKHS filters: rjdfilters::rkhs\_filter()

- RKHS theory used to approximate Henderson filter
- With  $K_p$  the **kernel function**, the symmetric filter:

$$\forall j \in \llbracket -h, h \rrbracket : w_j = \frac{K_p(j/b)}{\sum_{i=-h}^h K_p(i/b)}$$

➡ with  $b = h + 1$  and a specific  $K_p$  you have the Henderson filter

- For asymmetric filters:

$$\forall j \in \llbracket -h, q \rrbracket : w_{a,j} = \frac{K_p(j/b)}{\sum_{i=-h}^q K_p(i/b)}$$

## RKHS filters: `rjdfilters::rkhs_filter()`

- RKHS theory used to approximate Henderson filter
- With  $K_p$  the **kernel function**, the symmetric filter:

$$\forall j \in \llbracket -h, h \rrbracket : w_j = \frac{K_p(j/b)}{\sum_{i=-h}^h K_p(i/b)}$$

➡ with  $b = h + 1$  and a specific  $K_p$  you have the Henderson filter

- For asymmetric filters:

$$\forall j \in \llbracket -h, q \rrbracket : w_{a,j} = \frac{K_p(j/b)}{\sum_{i=-h}^q K_p(i/b)}$$

➡  $b$  chosen by optimization, e.g. minimizing revisions linked to phase-shift:

$$b_{q,\varphi} = \min_{b_q} \int_0^{2\pi/12} \rho_s(\lambda) \rho_\theta(\lambda) \sin^2 \left( \frac{\varphi_\theta(\omega)}{2} \right) d\omega$$

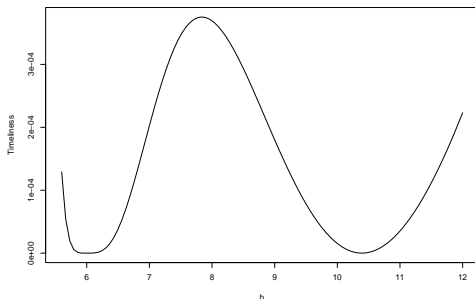
# Filtres asymétriques



several local extremum

```
fun <- rkhs_optimization_fun(horizon = 6,
                             leads = 5, degree = 3,
                             asymmetricCriterion = "Timeliness")
plot(fun, 5.6, 12, xlab = "b",
      ylab = "Timeliness", main = "6X5 filter")
```

6X5 filter



```
rkhs_optimal_bw()
```

```
##      q=0      q=1      q=2      q=3      q=4      q=5
## 6.0000 6.0000 6.3875 8.1500 9.3500 6.0000
```



Generalizable to  
create filters that could  
be applied to irregular  
frequency series



## FST approach: `rjdfilters::fst_filter()`

---

Minimization of a weighted sum of 3 criteria under polynomial constraints:

$$\begin{cases} \min_{\theta} & J(\theta) = \alpha F_g(\theta) + \beta S_g(\theta) + \gamma T_g(\theta) \\ s.c. & C\theta = a \end{cases}$$




$F_g$ : Fidelity (variance reduction ratio),  $S_g$ : Smoothness (Henderson criterion),  $T_g$  timeliness (phase-shift)

## FST approach: rjdfilters::fst\_filter()

Minimization of a weighted sum of 3 criteria under polynomial constraints:

$$\begin{cases} \min_{\theta} & J(\theta) = \alpha F_g(\theta) + \beta S_g(\theta) + \gamma T_g(\theta) \\ \text{s.c.} & C\theta = a \end{cases}$$

$F_g$ : Fidelity (variance reduction ratio),  $S_g$ : Smoothness (Henderson criterion),  $T_g$  timeliness (phase-shift)

-  Unique solution
-  Asymmetric filters independent of data and symmetric filter
-  Non-normalized weights

# Contents

---

## 1. Introduction

## 2. Description of the methods

## 3. Comparison of the methods

### 3.1 Methodology

### 3.2 Time delay

### 3.3 Revision

### 3.4 Some examples

## 4. Conclusion

# Methodology

---

2 404 calendar adjusted series (`sts_inpr_m`, industrial production indices of EU):

1. Seasonal adjustment with X-13ARIMA (RJDemetra::x13) for each date to extract: linearized component, length of trend and seasonal filters, decomposition mode and I-C ratio

# Methodology

---

2 404 calendar adjusted series (`sts_inpr_m`, industrial production indices of EU):

1. Seasonal adjustment with X-13ARIMA (`RJDemetra::x13`) for each date to extract: linearized component, length of trend and seasonal filters, decomposition mode and I-C ratio
2. Seasonal adjustment using *fixed* linearized component (est. overall period) and same length/decomposition/I-C ratio with X-11 using custom trend-cycle filters (`rjdfilters::x11()`).

# Methodology

---

2 404 calendar adjusted series (`sts_inpr_m`, industrial production indices of EU):

1. Seasonal adjustment with X-13ARIMA (`RJDemetra::x13`) for each date to extract: linearized component, length of trend and seasonal filters, decomposition mode and I-C ratio
2. Seasonal adjustment using *fixed* linearized component (est. overall period) and same length/decomposition/I-C ratio with X-11 using custom trend-cycle filters (`rjdfilters::x11()`).
3. For each estimate, downturns and upturns:
  - upturn:  $y_{t-3} \geq \dots \geq y_{t-1} < y_t \leq y_{t+1}$
  - downturn:  $y_{t-3} \leq \dots \leq y_{t-1} > y_t \geq y_{t+1}$

# Methodology

---

2 404 calendar adjusted series (`sts_inpr_m`, industrial production indices of EU):

1. Seasonal adjustment with X-13ARIMA (`RJDemetra::x13`) for each date to extract: linearized component, length of trend and seasonal filters, decomposition mode and I-C ratio
2. Seasonal adjustment using *fixed* linearized component (est. overall period) and same length/decomposition/I-C ratio with X-11 using custom trend-cycle filters (`rjdfilters::x11()`).
3. For each estimate, downturns and upturns:
  - upturn:  $y_{t-3} \geq \dots \geq y_{t-1} < y_t \leq y_{t+1}$
  - downturn:  $y_{t-3} \leq \dots \leq y_{t-1} > y_t \geq y_{t+1}$

Time-delay: time to detect the correct turning point without further revisions

Methods compared: RKHS minimizing phase-shift and local polynomial filters

## Time delay to detect turning points in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13 (900 series)

	X-13-ARIMA	RKHS	LC	QL	CQ	DAF
Min	2.0	2.00	2.00	2.00	2.00	2.00
D1	2.0	5.00	3.00	2.00	2.00	2.00
D2	3.0	5.00	3.00	3.00	2.00	2.00
D3	3.0	5.00	4.00	3.00	3.00	3.00
D4	3.0	5.00	4.00	3.00	5.00	3.00
Median	4.0	5.00	4.00	4.00	6.00	5.00
D6	4.0	7.00	4.00	5.00	6.00	6.00
D7	4.0	7.00	5.00	7.00	7.00	7.00
D8	5.0	7.00	5.00	7.00	7.00	7.00
D9	9.0	9.00	8.00	9.00	9.00	9.00
Max	14.0	14.00	14.00	14.00	14.00	14.00
Mean	4.4	6.29	4.69	4.97	5.32	5.09



# Time delay to detect turning points in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13 (900 series)

	X-13-ARIMA	RKHS	LC	QL	CQ	DAF
Min	2.0	2.00	2.00	2.00	2.00	2.00
D1	2.0	5.00	3.00	2.00	2.00	2.00
D2	3.0	5.00	3.00	3.00	2.00	2.00
D3	3.0	5.00	4.00	3.00	3.00	3.00
D4	3.0	5.00	4.00	3.00	5.00	3.00
Median	4.0	5.00	4.00	4.00	6.00	5.00
D6	4.0	7.00	4.00	5.00	6.00	6.00
D7	4.0	7.00	5.00	7.00	7.00	7.00
D8	5.0	7.00	5.00	7.00	7.00	7.00
D9	9.0	9.00	8.00	9.00	9.00	9.00
Max	14.0	14.00	14.00	14.00	14.00	14.00
Mean	4.4	6.29	4.69	4.97	5.32	5.09

# Time delay to detect turning points in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13 (900 series)

	X-13-ARIMA	RKHS	LC	QL	CQ	DAF
Min	2.0	2.00	2.00	2.00	2.00	2.00
D1	2.0	5.00	3.00	2.00	2.00	2.00
D2	3.0	5.00	3.00	3.00	2.00	2.00
D3	3.0	5.00	4.00	3.00	3.00	3.00
D4	3.0	5.00	4.00	3.00	5.00	3.00
Median	4.0	5.00	4.00	4.00	6.00	5.00
D6	4.0	7.00	4.00	5.00	6.00	6.00
D7	4.0	7.00	5.00	7.00	7.00	7.00
D8	5.0	7.00	5.00	7.00	7.00	7.00
D9	9.0	9.00	8.00	9.00	9.00	9.00
Max	14.0	14.00	14.00	14.00	14.00	14.00
Mean	4.4	6.29	4.69	4.97	5.32	5.09

# Time delay to detect turning points in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13 (900 series)

	X-13-ARIMA	<b>RKHS</b>	LC	QL	CQ	DAF
Min	2.0	2.00	2.00	2.00	2.00	2.00
D1	2.0	5.00	3.00	2.00	2.00	2.00
D2	3.0	5.00	3.00	3.00	2.00	2.00
D3	3.0	5.00	4.00	3.00	3.00	3.00
D4	3.0	5.00	4.00	3.00	5.00	3.00
Median	4.0	5.00	4.00	4.00	6.00	5.00
D6	4.0	7.00	4.00	5.00	6.00	6.00
D7	4.0	7.00	5.00	7.00	7.00	7.00
D8	5.0	7.00	5.00	7.00	7.00	7.00
D9	9.0	9.00	8.00	9.00	9.00	9.00
Max	14.0	14.00	14.00	14.00	14.00	14.00
Mean	4.4	6.29	4.69	4.97	5.32	5.09

# MAE in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13

$R_t$ : relative revision error between first and last estimates.

Distribution of  $\frac{MAE(R_t)}{MAE(R_t^{X-13})}$

	RKHS	LC	QL	CQ	DAF
Min	0.4	0.5	0.5	0.5	0.5
D1	0.9	0.9	1.1	1.1	1.1
D2	1.0	1.0	1.3	1.4	1.3
D3	1.1	1.1	1.4	1.5	1.5
D4	1.1	1.2	1.6	1.7	1.6
Median	1.2	1.3	1.7	1.8	1.8
D6	1.3	1.4	1.9	2.0	1.9
D7	1.4	1.5	2.0	2.3	2.2
D8	1.5	1.6	2.3	2.6	2.4
D9	1.7	1.8	2.6	3.1	2.9
Max	4.0	4.6	6.5	5.8	5.9
Mean	1.3	1.3	1.8	2.0	1.9

# MAE in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13

$R_t$ : relative revision error between first and last estimates.

Distribution of  $\frac{MAE(R_t)}{MAE(R_t^{X-13})}$

	RKHS	LC	QL	CQ	DAF
Min	0.4	0.5	0.5	0.5	0.5
D1	0.9	0.9	1.1	1.1	1.1
D2	1.0	1.0	1.3	1.4	1.3
D3	1.1	1.1	1.4	1.5	1.5
D4	1.1	1.2	1.6	1.7	1.6
Median	1.2	1.3	1.7	1.8	1.8
D6	1.3	1.4	1.9	2.0	1.9
D7	1.4	1.5	2.0	2.3	2.2
D8	1.5	1.6	2.3	2.6	2.4
D9	1.7	1.8	2.6	3.1	2.9
Max	4.0	4.6	6.5	5.8	5.9
Mean	1.3	1.3	1.8	2.0	1.9

# MAE in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13

$R_t$ : relative revision error between first and last estimates.

Distribution of  $\frac{MAE(R_t)}{MAE(R_t^{X-13})}$

	RKHS	LC	QL	CQ	DAF
Min	0.4	0.5	0.5	0.5	0.5
D1	0.9	0.9	1.1	1.1	1.1
D2	1.0	1.0	1.3	1.4	1.3
D3	1.1	1.1	1.4	1.5	1.5
D4	1.1	1.2	1.6	1.7	1.6
Median	1.2	1.3	1.7	1.8	1.8
D6	1.3	1.4	1.9	2.0	1.9
D7	1.4	1.5	2.0	2.3	2.2
D8	1.5	1.6	2.3	2.6	2.4
D9	1.7	1.8	2.6	3.1	2.9
Max	4.0	4.6	6.5	5.8	5.9
Mean	1.3	1.3	1.8	2.0	1.9

# MAE in 2020

For series for which the optimal trend-cycle symmetric filter is of length 13

$R_t$ : relative revision error between first and last estimates.

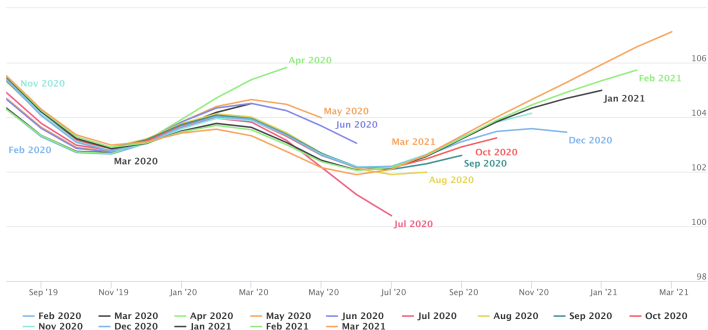
Distribution of  $\frac{MAE(R_t)}{MAE(R_t^{X-13})}$

	RKHS	LC	QL	CQ	DAF
Min	0.4	0.5	0.5	0.5	0.5
D1	0.9	0.9	1.1	1.1	1.1
D2	1.0	1.0	1.3	1.4	1.3
D3	1.1	1.1	1.4	1.5	1.5
D4	1.1	1.2	1.6	1.7	1.6
Median	1.2	1.3	1.7	1.8	1.8
D6	1.3	1.4	1.9	2.0	1.9
D7	1.4	1.5	2.0	2.3	2.2
D8	1.5	1.6	2.3	2.6	2.4
D9	1.7	1.8	2.6	3.1	2.9
Max	4.0	4.6	6.5	5.8	5.9
Mean	1.3	1.3	1.8	2.0	1.9

# IPI in the manufacture of fabricated metal products, except machinery and equipment (C25) in Sweden (turning point in February 2020)

Trend-cycle estimation of the series C25\_SE with the method X12-ARIMA

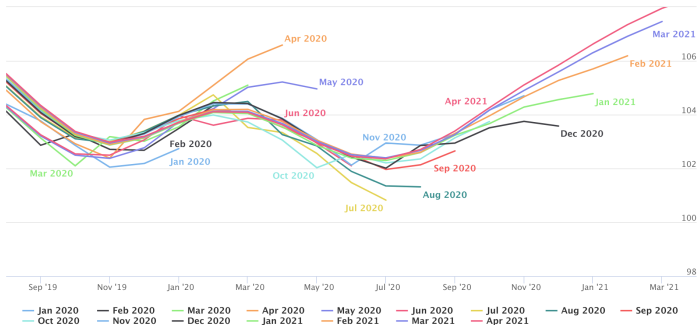
Time-delay = 5 months





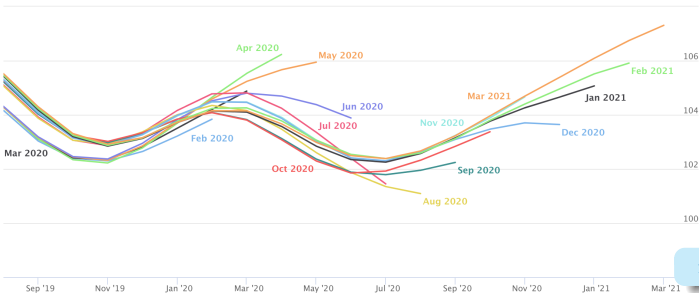
## Trend-cycle estimation of the series C25\_SE with the method RKHS

Time-delay = 13 months



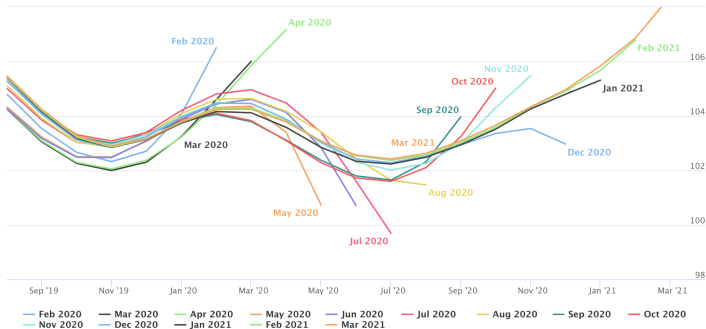
## Trend-cycle estimation of the series C25\_SE with the method LC

Time-delay = 13 months



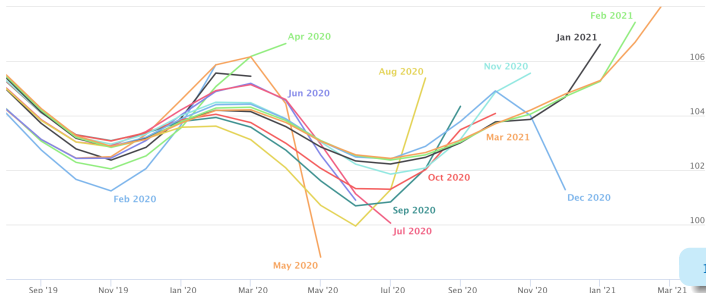
## Trend-cycle estimation of the series C25\_SE with the method QL

Time-delay = 14 months



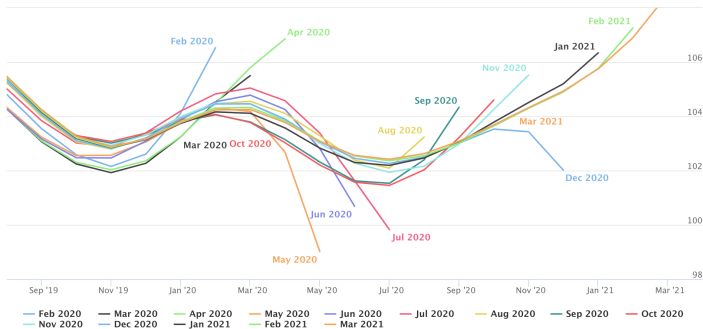
## Trend-cycle estimation of the series C25\_SE with the method CQ

Time-delay = 14 months



Trend-cycle estimation of the series C25\_SE with the method DAF

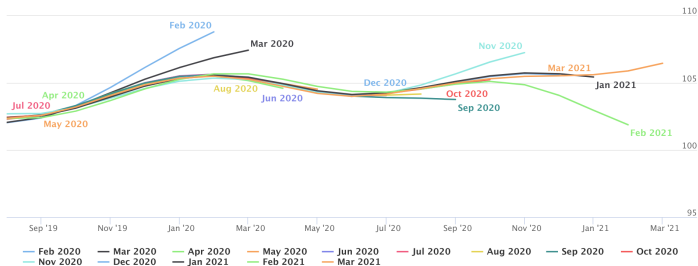
Time-delay = 14 months



# IPI in the manufacture of cement, lime and plaster (C235) in Germany (turning point in February 2020)

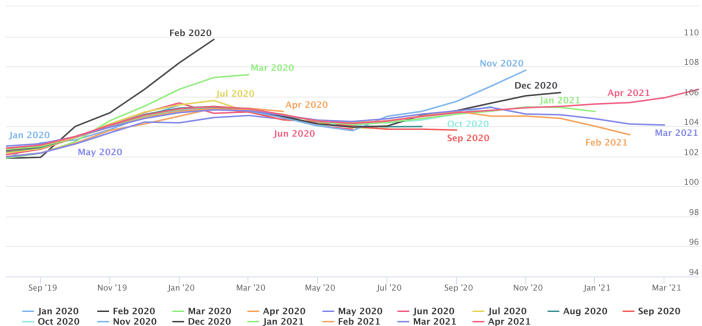
Trend-cycle estimation of the series C235\_DE with the method X12-ARIMA

Time-delay = 13 months



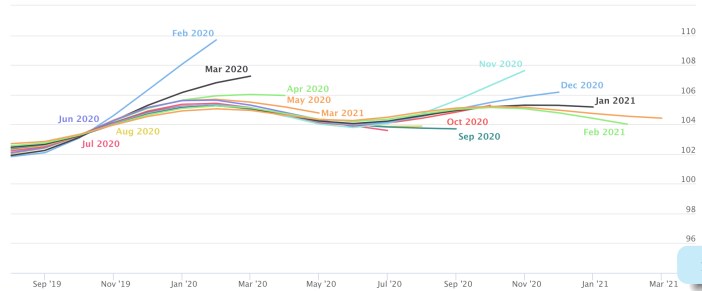
## Trend-cycle estimation of the series C235\_DE with the method RKHS

Time-delay = 5 months



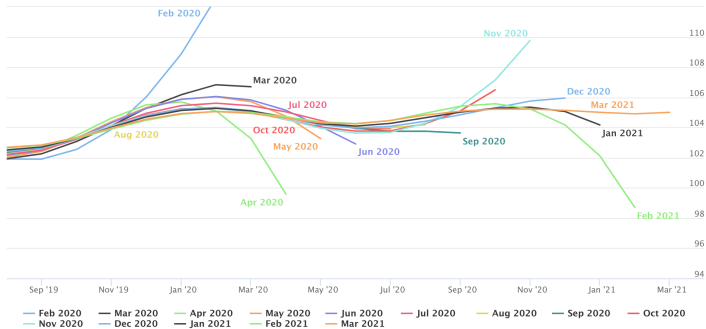
## Trend-cycle estimation of the series C235\_DE with the method LC

Time-delay = 3 months



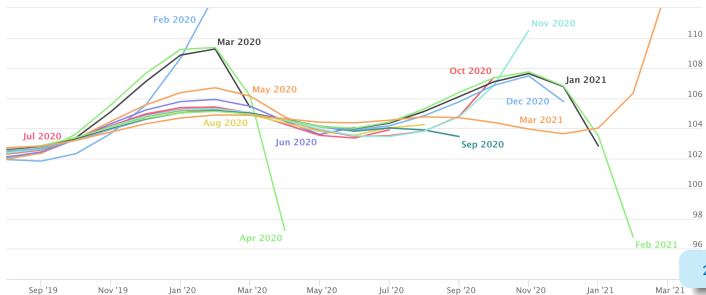
## Trend-cycle estimation of the series C235\_DE with the method QL

Time-delay = 3 months



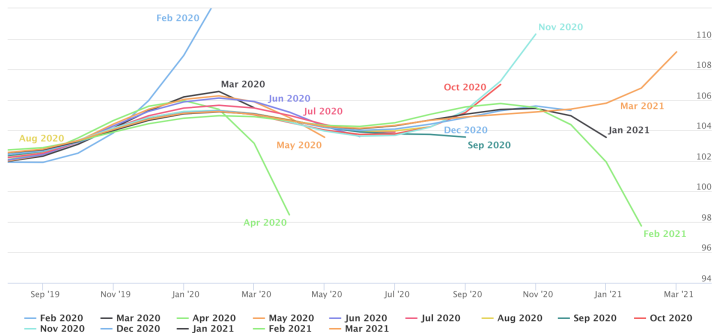
## Trend-cycle estimation of the series C235\_DE with the method CQ

Time-delay = 2 months



Trend-cycle estimation of the series C235\_DE with the method DAF

Time-delay = 3 months



# Conclusion

---

- To build asymmetric filters, we can focus on the ones that reproduces at most linear trend (excluding QL, CQ and DAF filters)



# Conclusion

---

- To build asymmetric filters, we can focus on the ones that reproduces at most linear trend (excluding QL, CQ and DAF filters)
- During the COVID-19, the current X-13-ARIMA algorithm seems to produce on average satisfying results

# Conclusion

---

- To build asymmetric filters, we can focus on the ones that reproduces at most linear trend (excluding QL, CQ and DAF filters)
- During the COVID-19, the current X-13-ARIMA algorithm seems to produce on average satisfying results
- In some cases, we could prefer others trend-cycle filters ➡ `rjdfilters` can help

# What next?

---

- More investigations to understand why and when a method performs better

# What next?

---

- More investigations to understand why and when a method performs better
- Study on other datasets

# What next?

---

- More investigations to understand why and when a method performs better
- Study on other datasets

Other methods:

- FST can lead to filters that performs better in terms of Fidelity, Smoothness, Timeliness than:
  - RKHS filters with same polynomial constraints
  - LC filters with same polynomial constraints
- ➔ Study of those moving averages?

# What next?

---

- More investigations to understand why and when a method performs better
- Study on other datasets

Other methods:

- FST can lead to filters that performs better in terms of Fidelity, Smoothness, Timeliness than:
  - RKHS filters with same polynomial constraints
  - LC filters with same polynomial constraints
- ➔ Study of those moving averages?
- Direct Filter Approach (Wildi and McElroy (2019)), Cascade Linear Filter (Dagum and Luati (2008)), etc.

# What next?

---

- More investigations to understand why and when a method performs better
- Study on other datasets

Other methods:

- FST can lead to filters that performs better in terms of Fidelity, Smoothness, Timeliness than:
  - RKHS filters with same polynomial constraints
  - LC filters with same polynomial constraints
- ➔ Study of those moving averages?
- Direct Filter Approach (Wildi and McElroy (2019)), Cascade Linear Filter (Dagum and Luati (2008)), etc.
- Impact of outliers? Study of robust methods?

# Bibliography

🔗 package: 🔗 palatej/rjdfilters

About me: 🔗 AQLT

## Bibliography:

- Dagum, Estela Bee, and Silvia Bianconcini. 2008. "The Henderson Smoother in Reproducing Kernel Hilbert Space." *Journal of Business & Economic Statistics* 26: 536–45. <https://ideas.repec.org/a/bes/jnlbes/v26y2008p536-545.html>.
- Dagum, Estela Bee, and Alessandra Luati. 2008. "A Cascade Linear Filter to Reduce Revisions and False Turning Points for Real Time Trend-Cycle Estimation." *Econometric Reviews* 28 (1-3): 40–59. <https://doi.org/10.1080/07474930802387837>.
- 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](http://ec.europa.eu/eurostat/web/products-manuals-and-guidelines/-/KS-GQ-18-001).
- Proietti, Tommaso, and Alessandra Luati. 2008. "Real Time Estimation in Local Polynomial Regression, with Application to Trend-Cycle Analysis." *Ann. Appl. Stat.* 2 (4): 1523–53.
- Wildi, Marc, and Tucker McElroy. 2019. "The Trilemma Between Accuracy, Timeliness and Smoothness in Real-Time Signal Extraction." *International Journal of Forecasting* 35 (3): 1072–84. <https://EconPapers.repec.org/RePEc:eee:intfor:v:35:y:2019:i:3:p:1072-1084>.