Workshop on TSA and SDCM for Official Statistics

Institut national de la statistique
et des études économiques

Mesurer pour comprendre

R package `tvCoef`, implementing time-varying coefficients models has never been so easy

Alain Quartier-la-Tente
Insee (Joint work with Claire du Campe de Rosamel)
Session 7: New tools for Seasonal Adjustment 2
Friday 15 December 2023

Sommaire

Introduction

Over the long term, institutions, corporate norms and the behavior of economic agents evolve, leading to changes in the dynamics of the economic series studied.

Introduction

Over the long term, institutions, corporate norms and the behavior of economic agents evolve, leading to changes in the dynamics of the economic series studied.

Many models are based on linear regressions (WDA, forecasts, benchmark, etc.), which assume that relationships between variables are fixed over time.

Introduction

Over the long term, institutions, corporate norms and the behavior of economic agents evolve, leading to changes in the dynamics of the economic series studied.

Many models are based on linear regressions (WDA, forecasts, benchmark, etc.), which assume that relationships between variables are fixed over time.

Assumption **true** in the **short term**, but generally **false** in the **long term** or in the presence of structural changes (change of nomenclature, definition, COVID...).

Introduction

Over the long term, institutions, corporate norms and the behavior of economic agents evolve, leading to changes in the dynamics of the economic series studied.

Many models are based on linear regressions (WDA, forecasts, benchmark, etc.), which assume that relationships between variables are fixed over time.

Assumption **true** in the **short term**, but generally **false** in the **long term** or in the presence of structural changes (change of nomenclature, definition, COVID...).

Goal:

- to study methods of relaxing this constraint;
- propose a simple way of implementing and comparing these methods (package R `tvCoef`).

Linear regression model

General idea:

$$
\begin{aligned}
& y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \\
\iff & y_t = \beta X_t + \varepsilon_t
\end{aligned}
$$

β estimated using the OLS

Linear regression model

General idea:

$$\begin{aligned} & y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \\ \iff & y_t = \beta X_t + \varepsilon_t \end{aligned}$$

β estimated using the OLS

Example: forecast of French production growth in other manufacturing using

- IPI overhang
- INSEE business climate
- Balances of opinion published by INSEE and Banque de France

Model estimated with `stats::lm()` or `dynlm::dynlm()`

Linear regression model

General idea:

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2)$$
$$\iff y_t = \beta X_t + \varepsilon_t$$

β estimated using the OLS

Example: forecast of French production growth in other manufacturing using

- IPI overhang
- INSEE business climate
- Balances of opinion published by INSEE and Banque de France

Model estimated with `stats::lm()` or `dynlm::dynlm()`

Even if machine learning models can be used, linear models performs well and are often used as reference models

R code (1)

```r
library(tvCoef)
library(dynlm)
data <- window(manufacturing, start = 1993, end = c(2019, 4))
y <- data[, "prod_c5"]
model_c5 <- dynlm(
    formula = prod_c5 ~ overhang_ipi1_c5 + insee_bc_c5_m3 +
        + diff(insee_tppre_c5_m3, 1) + diff(bdf_tuc_c5_m2, 1),
    data = data
)
summary(model_c5)
```

```
Time series regression with "ts" data:
Start = 1993(2), End = 2019(4)

Call:
dynlm(formula = prod_c5 ~ overhang_ipi1_c5 + insee_bc_c5_m3 +
    +diff(insee_tppre_c5_m3, 1) + diff(bdf_tuc_c5_m2, 1), data = data)

Residuals:
     Min       1Q   Median       3Q      Max
```

R code (2)

```
-2.33630 -0.46798  0.02535  0.47036  1.61058

Coefficients:
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                -5.223513   0.798384  -6.543 2.46e-09 ***
overhang_ipi1_c5            0.100841   0.022195   4.543 1.52e-05 ***
insee_bc_c5_m3              0.050966   0.007969   6.396 4.89e-09 ***
diff(insee_tppre_c5_m3, 1)  0.040771   0.011052   3.689 0.000363 ***
diff(bdf_tuc_c5_m2, 1)      0.410629   0.068227   6.019 2.79e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7165 on 102 degrees of freedom
Multiple R-squared:  0.7151,    Adjusted R-squared:  0.7039
F-statistic:    64 on 4 and 102 DF,  p-value: < 2.2e-16
```

Goal

Study different methods to estimate

$$y_t = \beta_t X_t + \varepsilon_t$$

Goal

Study different methods to estimate

$$y_t = \beta_t X_t + \varepsilon_t$$

Idea: stay close to the case of linear regression so that results remain easily interpretable

Goal

Study different methods to estimate

$$y_t = \beta_t X_t + \varepsilon_t$$

Idea: stay close to the case of linear regression so that results remain easily interpretable

Outline:

1. Statistical tests
2. Piecewise regressions
3. Local regressions
4. State-space models

Sommaire

Statistical tests: Bai and Perron

Most famous test: Bai and Perron, closed to Chow test.
They propose an efficient algorithm for finding break dates (package `strucchange`). Let the model be:

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t$$

Statistical tests: Bai and Perron

Most famous test: Bai and Perron, closed to Chow test.
They propose an efficient algorithm for finding break dates (package `strucchange`). Let the model be:

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t$$

We split it in two, around a date t_1, and obtain two sub-models:

$$\forall t \leq t_1 : \quad y_t = \beta'_0 + \beta'_1 x_{1,t} + \cdots + \beta'_p x_{p,t} + \varepsilon_t$$

$$\forall t > t_1 : \quad y_t = \beta'_0 + \beta'_1 x_{1,t} + \cdots + \beta'_p x_{p,t} + \varepsilon_t$$

The null hypothesis assumes that $\beta'_0 = \beta''_0$, $\beta'_1 = \beta''_1, \ldots \beta'_p = \beta''_p$

R code (1)

```
strucchange::breakdates(strucchange::breakpoints(
    prod_c5 ~ overhang_ipi1_c5 + insee_bc_c5_m3
    + `diff(insee_tppre_c5_m3, 1)` + `diff(bdf_tuc_c5_m2, 1)`,
    data = model_c5$model))
```

```
[1] 2008.5
```

Bai Perron's limitations

- The break may only be on a subset of variables but in `strucchange` only global tests implemented.

Bai Perron's limitations

- The break may only be on a subset of variables but in `strucchange` only global tests implemented.
- Instability in the choice of date and the break is not necessarily abrupt (e.g. slow evolution over time).

Bai Perron's limitations

- The break may only be on a subset of variables but in `strucchange` only global tests implemented.
- Instability in the choice of date and the break is not necessarily abrupt (e.g. slow evolution over time).
- Structural breaks are usually known

Bai Perron's limitations

- The break may only be on a subset of variables but in `strucchange` only global tests implemented.
- Instability in the choice of date and the break is not necessarily abrupt (e.g. slow evolution over time).
- Structural breaks are usually known
- Assume that there is a break date to be determined, we might just want to test whether the coefficients are constant or not

Nyblom and Hansen

$$\begin{cases} (H_0): & \text{constant coefficients} \\ (H_1): & \text{coefficients follow a martingale} \end{cases}$$

Nyblom and Hansen

$$\begin{cases} (H_0): & \text{constant coefficients} \\ (H_1): & \text{coefficients follow a martingale} \end{cases}$$

Hansen limits:

- Test for variance not stable (go through other tests)
- Joint test does not apply to dummies
- Applies only to stationary variables

code (1)

tvCoef::hansen_test()

```
hansen_test(model_c5)
```

```
Variable                      L         Stat     Conclusion
-------------------------------------------------------------
(Intercept)                     0.2725   0.47   FALSE
overhang_ipi1_c5                0.8392   0.47    TRUE
insee_bc_c5_m3                  0.2867   0.47   FALSE
diff(insee_tppre_c5_m3, 1)      0.2568   0.47   FALSE
diff(bdf_tuc_c5_m2, 1)          0.1491   0.47   FALSE
Variance                        0.4881   0.47    TRUE
Joint Lc                        1.3188   1.9   FALSE


Lecture: True means reject H0 at level 5%
```

Sommaire

Piecewise linear regressions

Associated to Bai Perron

$$\exists t_1, \ldots, t_{T-1} : \beta_t = \beta_1 1_{t \leq t_1} + \beta_2 1_{t_1 < t \leq t_2} + \cdots + \beta_T 1_{t_{T-1} < t}$$

Piecewise linear regressions

Associated to Bai Perron

$$\exists t_1, \ldots, t_{T-1} : \ \beta_t = \beta_1 1_{t \le t_1} + \beta_2 1_{t_1 < t \le t_2} + \cdots + \beta_T 1_{t_{T-1} < t}$$

Estimated by:

1. Dividing the regressors ($\mathbb{V}[\varepsilon_t]$ fixed in time) R `tvCoef::piece_reg()`
2. Piecewise linear regressions ($\mathbb{V}[\varepsilon_t]$ varies by subperiod) R `tvCoef::bp_lm()`

Piecewise linear regressions

Associated to Bai Perron

$$\exists t_1, \dots, t_{T-1} : \beta_t = \beta_1 1_{t \leq t_1} + \beta_2 1_{t_1 < t \leq t_2} + \dots + \beta_T 1_{t_{T-1} < t}$$

Estimated by:

1. Dividing the regressors ($\mathbb{V}[\varepsilon_t]$ fixed in time) `tvCoef::piece_reg()`
2. Piecewise linear regressions ($\mathbb{V}[\varepsilon_t]$ varies by subperiod) `tvCoef::bp_lm()`

➔ use case 1 because gives a single regression output.

In both cases, coefficient estimates remain the same, differences on variances and on real-time estimates.

➔ Pros:

- Simple to understand and implement
- Easily combined with other types of models (local regressions)

- Pros:
  - Simple to understand and implement
  - Easily combined with other types of models (local regressions)
- Cons:
  - Assumes the existence of an abrupt break
  - Imprecision in date selection

R code (1)

```r
pwr_mod <- piece_reg(model_c5)
summary(pwr_mod)
```

```

Time series regression with "ts" data:
Start = 1993(2), End = 2019(4)

Call:
dynlm::dynlm(formula = as.formula(formula), data = data2)

Residuals:
     Min       1Q   Median       3Q      Max
-1.54982 -0.38309 -0.07791  0.43409  1.27348

Coefficients:
                                      Estimate Std. Error t value Pr(>|t|)
`(Intercept)_2008.5`                 -5.516115   0.903736  -6.104 2.13e-08 ***
overhang_ipi1_c5_2008.5               0.098701   0.030054   3.284 0.001424 **
insee_bc_c5_m3_2008.5                 0.053113   0.008527   6.229 1.21e-08 ***
`diff(insee_tppre_c5_m3, 1)_2008.5`   0.030069   0.012007   2.504 0.013942 *
`diff(bdf_tuc_c5_m2, 1)_2008.5`       0.286032   0.108827   2.628 0.009977 **
```

code (2)

```
`(Intercept)_2019.75`                  -5.569186   1.259533  -4.422 2.56e-05 ***
overhang_ipi1_c5_2019.75                0.443899   0.063963   6.940 4.44e-10 ***
insee_bc_c5_m3_2019.75                  0.054734   0.012796   4.278 4.43e-05 ***
`diff(insee_tppre_c5_m3, 1)_2019.75`    0.061845   0.016102   3.841 0.000219 ***
`diff(bdf_tuc_c5_m2, 1)_2019.75`        0.296779   0.080191   3.701 0.000357 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6178 on 97 degrees of freedom
Multiple R-squared:  0.8049,    Adjusted R-squared:  0.7848
F-statistic: 40.02 on 10 and 97 DF,  p-value: < 2.2e-16
```

To only split the second variable:

```
pwr_mod2 <- piece_reg(model_c5, break_dates = 2008.5, fixed_var = -2)
summary(pwr_mod2)
```

code (3)

```
Time series regression with "ts" data:
Start = 1993(2), End = 2019(4)

Call:
dynlm::dynlm(formula = as.formula(formula), data = data2)

Residuals:
     Min       1Q   Median       3Q      Max
-1.57740 -0.40673 -0.05138  0.44438  1.42772

Coefficients:
                              Estimate Std. Error t value Pr(>|t|)
`(Intercept)`                -5.261235   0.685772  -7.672 1.09e-11 ***
insee_bc_c5_m3                0.051493   0.006845   7.522 2.28e-11 ***
`diff(insee_tppre_c5_m3, 1)`  0.041475   0.009493   4.369 3.03e-05 ***
`diff(bdf_tuc_c5_m2, 1)`      0.324442   0.060278   5.382 4.79e-07 ***
overhang_ipi1_c5_2008.5       0.081865   0.019316   4.238 4.99e-05 ***
overhang_ipi1_c5_2019.75      0.458413   0.061602   7.441 3.38e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

R code (4)

```
Residual standard error: 0.6154 on 101 degrees of freedom
Multiple R-squared:  0.7984,    Adjusted R-squared:  0.7865
F-statistic: 66.68 on 6 and 101 DF,  p-value: < 2.2e-16
```

Local regressions: R `tvReg`

Assumption $\beta_t = \beta(z_t)$ with default $z_t = t/T$ and $\beta()$ locally constant (Nadaraya-Watson) or locally linear.

Local regressions: R tvReg

Assumption $\beta_t = \beta(z_t)$ with default $z_t = t/T$ and $\beta()$ locally constant (Nadaraya-Watson) or locally linear.

$$\beta(z_t) = \underset{\theta_0}{argmin} \sum_{j=1}^{T} \left(y_j - x_j\theta_0\right)^2 K_b(z_j - z_t)$$

With $K_b(x) = \frac{1}{b}K(x/b)$ a kernel function to weight the observations.

Local regressions: R tvReg

Assumption $\beta_t = \beta(z_t)$ with default $z_t = t/T$ and $\beta()$ locally constant (Nadaraya-Watson) or locally linear.

$$\beta(z_t) = \underset{\theta_0}{argmin} \sum_{j=1}^{T} (y_j - x_j\theta_0)^2 K_b(z_j - z_t)$$

With $K_b(x) = \frac{1}{b}K(x/b)$ a kernel function to weight the observations.

Remark:

- Bandwidth b fixed or estimated.
- If $b \geq 1$ all data used for each estimate.
- If $b \to 20$ the weight associated with each obs almost identical for all data $\simeq$ linear regression.

Pros:

- Simple model

Cons:

- All coefficients vary
- Problem of choosing b: by cross-validation (between 0 and 20) but not very discriminating.
- Strong real-time revisions possible (in estimates of b and due to the use of asymmetric kernel)

➔ Pros:

- Simple model

➔ Cons:

- All coefficients vary
- Problem of choosing b: by cross-validation (between 0 and 20) but not very discriminating.
- Strong real-time revisions possible (in estimates of b and due to the use of asymmetric kernel)

Note:

- Possibility of combining previous models by estimating a local regression on cut data
- By performing two regressions, we can fix the coefficients of certain variables

code (1)

```
lr_mod <- tvReg::tvLM(model_c5)
```

```
Calculating regression bandwidth... bw =  0.4203537
```

```
summary(lr_mod)
```

```
Call:
tvReg::tvLM(formula = model_c5)

Class:  tvlm

Summary of time-varying estimated coefficients:
================================================
        (Intercept) overhang_ipi1_c5 insee_bc_c5_m3 diff(insee_tppre_c5_m3, 1)
Min.         -5.953          0.06942        0.03446                  -0.007961
1st Qu.      -5.702          0.11492        0.03766                   0.032127
Median       -4.724          0.21878        0.04651                   0.037311
Mean         -4.790          0.22592        0.04577                   0.038769
3rd Qu.      -4.044          0.33430        0.05459                   0.051159
Max.         -3.817          0.38812        0.05791                   0.058446
```

code (2)

```
       diff(bdf_tuc_c5_m2, 1)
Min.                   0.2195
1st Qu.                0.2549
Median                 0.3190
Mean                   0.3388
3rd Qu.                0.4300
Max.                   0.4800

Bandwidth:  0.4204
Pseudo R-squared:  0.7908
```

State-space models

State-space modeling = general methodology for dealing with a wide range of time-series problems

State-space models

State-space modeling = general methodology for dealing with a wide range of time-series problems

Hypothesis: problem determined by a series of *unobserved* vectors $\alpha_1, \ldots, \alpha_n$ associated with observations $y_1, \ldots, y_n$, the relationship between α_t and y_t being specified by the state-space model.

State-space models

State-space modeling = general methodology for dealing with a wide range of time-series problems

Hypothesis: problem determined by a series of *unobserved* vectors $\alpha_1, \dots, \alpha_n$ associated with observations $y_1, \dots, y_n$, the relationship between α_t and y_t being specified by the state-space model.

Several forms of model are possible, the simplest being linear Gaussian models. Simplified version:

$$\begin{cases} y_t = X_t\alpha_t + \varepsilon_t, & \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \\ \alpha_{t+1} = \alpha_t + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma^2 Q) \end{cases}, \text{ with } \eta_t \text{ and } \varepsilon_t \text{ independent}$$

with y_t of dimension $p \times 1$ vector of observations, and α_t of dimension $m \times 1$ vector of states (*state vector*).

State-space models

State-space modeling = general methodology for dealing with a wide range of time-series problems

Hypothesis: problem determined by a series of *unobserved* vectors $\alpha_1, \dots, \alpha_n$ associated with observations $y_1, \dots, y_n$, the relationship between α_t and y_t being specified by the state-space model.

Several forms of model are possible, the simplest being linear Gaussian models. Simplified version:

$$\begin{cases} y_t = X_t\alpha_t + \varepsilon_t, & \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \\ \alpha_{t+1} = \alpha_t + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma^2 Q) \end{cases}, \text{ with } \eta_t \text{ and } \varepsilon_t \text{ independent}$$

with y_t of dimension $p \times 1$ vector of observations, and α_t of dimension $m \times 1$ vector of states (*state vector*).

σ^2 a factor simplifying the estimates (*Concentration of loglikelihood*).

Back to linear regression

Linear regression:

$$\begin{cases} y_t = X_t\alpha + \varepsilon_t, & \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \\ \alpha_{t+1} = \alpha_t = \dots = \alpha_0 = \alpha \end{cases}$$

Kalman filter estimation

Two classic operations: *filtering* and *smoothing*

- Smoothing: estimates the coefficient at each date using all available information. Close to the estimates in-sample forecasts.

$$\hat{\alpha}_t = E[\alpha_t | y_0, \ldots, y_n]$$

Ex: linear regression: $\hat{\alpha}_t = \hat{\alpha}$

Kalman filter estimation

Two classic operations: *filtering* and *smoothing*

- Smoothing: estimates the coefficient at each date using all available information. Close to the estimates in-sample forecasts.

$$\hat{\alpha}_t = E[\alpha_t | y_0, \dots, y_n]$$

Ex: linear regression: $\hat{\alpha}_t = \hat{\alpha}$

- Filtering: estimates the next coefficient (in $t+1$) with the information known in t. Close to real-time (out-of-sample) forecasts.

$$a_{t+1} = E[\alpha_{t+1} | y_0, \dots, y_t]$$

Ex: linear regression: $a_{2010T2} = \hat{\alpha}$ estimated using data up to 2010T1

Implementation

Usually the implementation can be difficult and variance has to be fixed...

Can be implemented easily with `rjd3sts`

`tvCoef::ssm_lm()` uses `rjd3sts::reg()` and `rjd3sts::locallevel()`.

code (1)

```
ssm_mod <- ssm_lm(
    model_c5, fixed_var_variables = FALSE, fixed_var_intercept = FALSE,
    var_intercept = 0.01, var_variables = 0.01)
summary(ssm_mod)
```

```
Summary of time-varying estimated coefficients (smoothing):
        (Intercept) overhang_ipi1_c5 insee_bc_c5_m3 diff(insee_tppre_c5_m3, 1)
Min.         -5.002           0.1299        0.04326                   0.002605
1st Qu.      -4.912           0.2108        0.04326                   0.026746
Median       -4.769           0.2741        0.04326                   0.029662
Mean         -4.703           0.2695        0.04326                   0.031070
3rd Qu.      -4.436           0.3423        0.04326                   0.037051
Max.         -4.344           0.3707        0.04326                   0.065948
        diff(bdf_tuc_c5_m2, 1)      noise
Min.                    0.2460 -1.292e+00
1st Qu.                 0.2711 -3.415e-01
Median                  0.2946 -9.216e-03
Mean                    0.2957 -7.919e-17
3rd Qu.                 0.3078  3.612e-01
Max.                    0.3794  1.234e+00
```

code (2)

To fix all the variables except one:

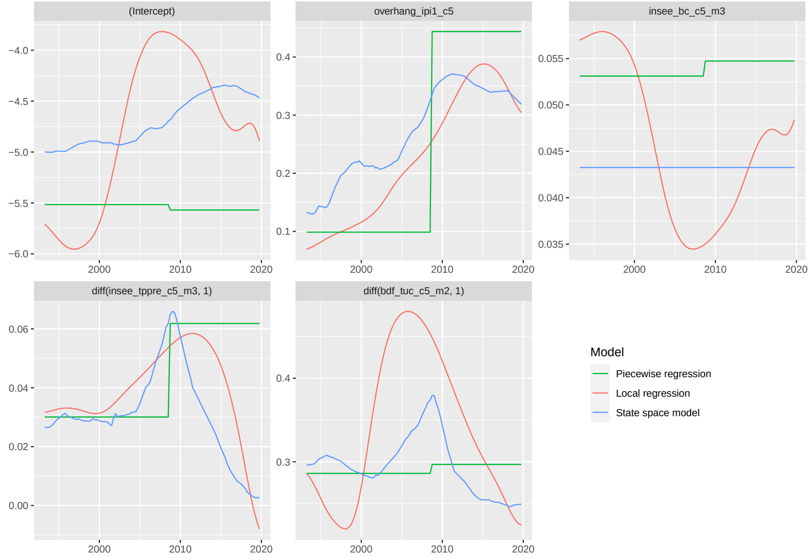
```
ssm_mod2 <- ssm_lm(
    model_c5,
    fixed_var_variables = c(FALSE, rep(TRUE, 5)),
    var_variables = c(0.01, rep(0, 5))
)
summary(ssm_mod2)
```

```
Summary of time-varying estimated coefficients (smoothing):
        (Intercept) overhang_ipi1_c5 insee_bc_c5_m3 diff(insee_tppre_c5_m3, 1)
Min.          -4.58          0.06429          0.044                    0.04181
1st Qu.       -4.58          0.11283          0.044                    0.04181
Median        -4.58          0.19523          0.044                    0.04181
Mean          -4.58          0.22462          0.044                    0.04181
3rd Qu.       -4.58          0.34704          0.044                    0.04181
Max.          -4.58          0.37637          0.044                    0.04181
        diff(bdf_tuc_c5_m2, 1)      noise
Min.                     0.369 -1.385e+00
1st Qu.                  0.369 -4.042e-01
Median                   0.369  2.758e-02
Mean                     0.369 -5.895e-16
```

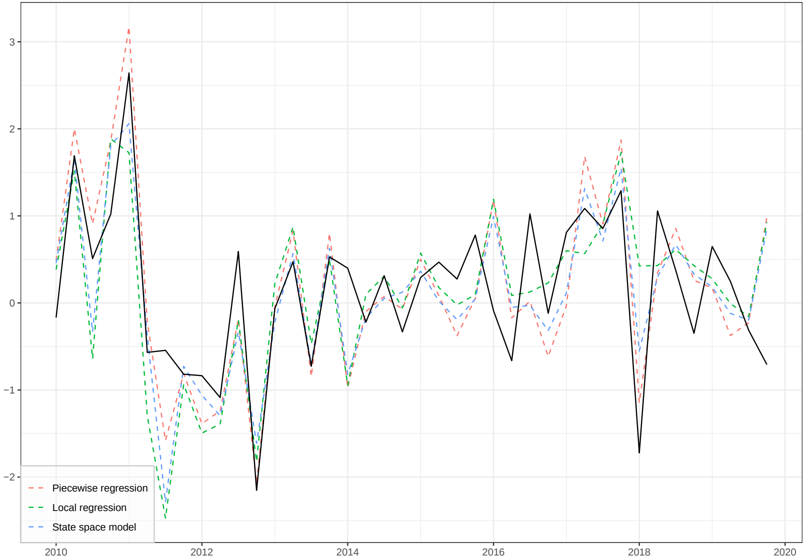
R code (3)

```
3rd Qu.                         0.369   3.523e-01
Max.                            0.369   1.423e+00
```

Results (1)


(Intercept)
overhang_ipi1_c5
insee_bc_c5_m3
diff(insee_tppre_c5_m3, 1)
diff(bdf_tuc_c5_m2, 1)
−4.0
−4.5
−5.0
−5.5
−6.0
0.4
0.3
0.2
0.1
0.055
0.050
0.045
0.040
0.035
0.06
0.04
0.02
0.00
2000
2010
2020
Model
Piecewise regression
Local regression
State space model

Results (2)


3
2
1
0
−1
−2
2010
2012
2014
2016
2018
2020
Piecewise regression
Local regression
State space model

Conclusion

- Many models can be estimated around linear regressions: the framework remains simple, but the modeling is more complex. ➔ modeling choices must be made

Conclusion

- Many models can be estimated around linear regressions: the framework remains simple, but the modeling is more complex. ➔ modeling choices must be made
- Can improve the performance of "classical" models, they do not replace them.
  (Study of ~ 30 forecasts models: in-sample and out-of-sample errors always reduced with state space models)

Conclusion

- Many models can be estimated around linear regressions: the framework remains simple, but the modeling is more complex. ➔ modeling choices must be made
- Can improve the performance of "classical" models, they do not replace them.
  (Study of $\sim$ 30 forecasts models: in-sample and out-of-sample errors always reduced with state space models)
- Models sometimes complex to implement (especially state-space) ➔ `tvCoef` can help (InseeFrLab/tvCoef)

See workshop for complete example: https://aqlt.github.io/AteliertvCoef/

Thanks for you attention

TODO for `tvCoef`: be able to handle AR-X models.

https://github.com/InseeFrLab/tvCoef