Time-Varying OLS/IV estimators in gretl: the ketvals package

Riccardo (Jack) Lucchetti   Francesco Valentini

version 1.1

Abstract

This package implements non-parametric least squares estimation for linear models with time varying parameters with or without instrumental variables, proposed by Giraitis et al. (2021)

Contents

1 Introduction

The name of the package is a quasi-acronym: ***ke**rnel-based **t**ime-**va**rying **l**east **s**quares*. The package provides functions to perform non-parametric least squares estimation for linear models with time varying parameters and possibly endogenous explanatory variables as proposed by Giraitis et al. (2021), building on earlier work by the same authors (see Giraitis et al., 2014, 2018).

Apart from the estimators and their covariance matrix, `ketvals` provides the classic instrumental variable (IV) diagnostic tools (the Hausman and Sargan tests) adapted to the time-varying framework. Users can retrieve and visualise the results by using dedicated functions.

1.1 The Model

We consider the model, for $t = 1 \dots T$,

$$\begin{cases} y_t = \boldsymbol{x}_t' \boldsymbol{\beta_t} + u_t \\ \boldsymbol{x}_t = \boldsymbol{z}_t' \boldsymbol{\psi}_t + \boldsymbol{\nu}_t \end{cases} \tag{1}$$

where y_t is the dependent variable, $\boldsymbol{x}_t$ is a $k \times 1$ vector of (possibly endogenous) regressors, $\boldsymbol{z}_t$ denotes a $q \times 1$ vector of instruments and the parameters $\boldsymbol{\beta_t}$ and $\boldsymbol{\psi}_t$ are allowed to vary over time. Finally, u_t and $\boldsymbol{\nu}_t$ denote shocks.

Among others proposed in Giraitis et al. (2021), the package provides two different estimators:

1. Time-varying OLS estimator

$$\hat{\boldsymbol{\beta}}_t = \left(\sum_{j=1}^{T} b_{H,|j-t|} \boldsymbol{x}_j \boldsymbol{x}_j' \right)^{-1} \left(\sum_{j=1}^{T} b_{H,|j-t|} \boldsymbol{x}_j y_j \right) \tag{2}$$

2. Time-varying IV estimator[1]

$$\tilde{\boldsymbol{\beta}}_t = \left(\sum_{j=1}^{T} b_{H,|j-t|} \hat{\boldsymbol{\psi}}_j' \boldsymbol{z}_j \boldsymbol{x}_j' \right)^{-1} \left(\sum_{j=1}^{T} b_{H,|j-t|} \hat{\boldsymbol{\psi}}_j' \boldsymbol{z}_j y_j \right) \tag{3}$$

where $b_{H,|j-t|} = K\left(\frac{|j-t|}{H}\right)$ denotes a kernel weight with bandwidth parameter $H = T^{h_1}$ and $\hat{\boldsymbol{\psi}}_j$ denotes the coefficients $\boldsymbol{\psi}_t$ estimated by the time-varying first-stage OLS, so that

$$\hat{\boldsymbol{\psi}}_t = \left(\sum_{j=1}^{T} b_{L,|j-t|} \boldsymbol{z}_j \boldsymbol{z}_j' \right)^{-1} \left(\sum_{j=1}^{T} b_{L,|j-t|} \boldsymbol{z}_j \boldsymbol{x}_j' \right),$$

[1] Denoted $\tilde{\beta}_{1,t}$ in the original work.

allowing for a different kernel bandwidth $L = T^{h_2}$ in the "first stage" regression.

Apart from the estimators and their covariance matrix, `ketvals` provides the classic IV diagnostic tools adapted to the time-varying framework:

- the time varying Hausman test;
- the time varying Over-identification test;
- the global Hausman test, which tests the exogeneity hypothesis for a given time interval between T_0 and T_1, with $0 \leq T_0 < T_1 \leq T$.

We refer to Giraitis et al. (2021) for details.

2 Basic usage

Usage of the package typically proceeds as follows:

1. preprocess the data;
2. estimate the model(s) and store them as bundles;
3. retrieve the necessary results from the estimated bundles via the dedicated functions.

As a simple example, we will estimate a time-varying version of a 1970-style "consumption function", where we take GDP as endogenous and we use investment and lagged investment as instruments (note: this example is used for purely pedagogical purposes; we don't imply in any way that the following is a meaningful macroeconometric exercise). The script for this example is provided in the "examples" directory of the function package.

Step 1 may be accomplished as follows:

```
include ketvals.gfn
open fedstl.bin
data gdpc1 pcecc96 gpdic1

series GDP = log(gdpc1)
series Cons = log(pcecc96)
series Inv = log(gpdic1)

list X = const time GDP Cons(-1)         # --- regressors
list Z = const time Inv Inv(-1) Cons(-1) # --- instruments

smpl 1984:1 2020:4
```

where, after loading the `ketvals` package, we fetch the data from the St. Louis FED archive supplied with gretl and set the estimation sample from 1984Q1 to 2020:4.

As for step 2,

```
mod1 = tv_OLS(Cons, X, 0.6)
mod2 = tv_IV(Cons, X, Z, 0.6)
```

so we have the OLS estimates in the bundle `mod1` and the IV estimates in the bundle `mod2`. We use in both cases the standard choice for the kernel function (Gaussian) and we set the bandwidth parameters to $h_1 = h_2 = 0.6$. Since the number of observation T equals 148, the actual bandwidth is $T^{0.6} = 20.052$.

Successful execution of the block above will yield the following output:

```
ketvals: OLS estimation complete
Sample: 1984:1 - 2020:4 (148 observations)
Kernel type: Gaussian, bandwidth = 20.052 (param = 0.6)
Dependent variable: Cons
Explanatory variables:
        const time GDP Cons_1


ketvals: IV estimation complete
Sample: 1984:1 - 2020:4 (148 observations)
Kernel type: Gaussian, bandwidth = 20.052 (param = 0.6)
First stage bandwidth = 20.052 (param = 0.6)
Dependent variable: Cons
Explanatory variables:
        const time GDP Cons_1
Instruments:
        const time Inv Inv_1 Cons_1
```

Finally, Step 3 may be something like

```
series b_ols = coeff_save(mod1, 3)
series b_iv = coeff_save(mod2, 3)
gnuplot b_ols b_iv --with-lines --time-series --output=display
```

in which we store the coefficient for income for both models: note that `GDP` is the third element of the regressors list `X`, so we have to pass this number as the second parameter of the `coeff_save()` function. Finally, we generate a plot. You should see something similar to Figure 1.

3 Automatic bandwidth selection

The version `v0.2` of the package introduces an automatic selection procedure for the bandwidth parameter, based on the modified AIC proposed by Cai (2007).

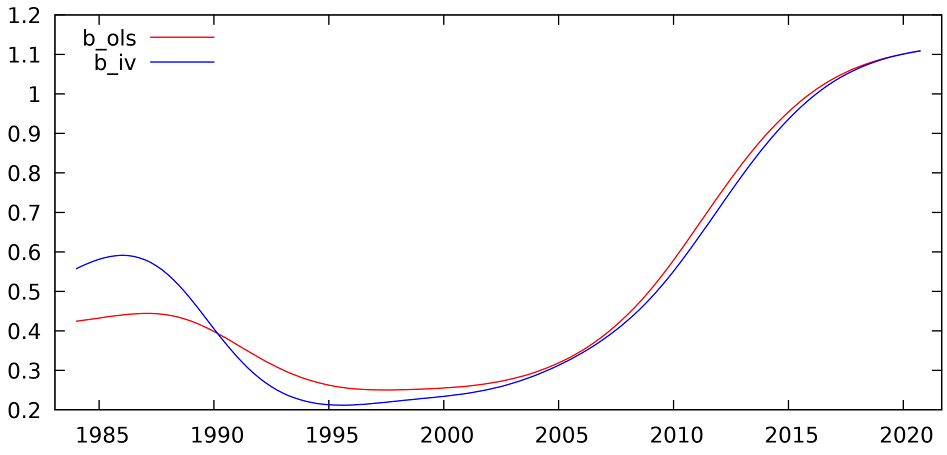


Figure 1: Simple usage example

3.1 Optimal bandwidth: background

The estimators in Equations (2) and (3) can be rewritten as

$$\bar{\beta}_t = \left[\mathbf{W}_t'\mathbf{X}\right]^{-1}\mathbf{W}_t'\mathbf{y} \tag{4}$$

where $\mathbf{W}_t$ and $\mathbf{X}$ are matrices with T rows and k columns and $\mathbf{y}$ is a T-element vector. While the matrices $\mathbf{X}$ and $\mathbf{y}$ simply contain the observations of the explanatory and dependent variables, respectively, the $\mathbf{W}_t$ matrix contains kernel-weighted entries, where

$$\mathbf{W}_t^{OLS} = D_t\mathbf{X},$$

where D_t is a diagonal $T \times T$ matrix whose element $[D_t]_{i,i}$ equals

$$[D_t]_{i,i} = b_{H,|t-i|}.$$

and

$$\mathbf{W}_t^{IV} = D_t\hat{\mathbf{X}}$$

with $\hat{\mathbf{X}}$ being the first-stage fitted values of explanatory variables $\mathbf{X}$.

In this way, it is possible to rewrite the model in Equation (1) in matrix form as

$$\hat{\boldsymbol{Y}} = \boldsymbol{H}_h\boldsymbol{Y}$$

where

$$H_h = \begin{bmatrix} \boldsymbol{x}_1' & & & \\ & \boldsymbol{x}_2' & & \\ & & \ddots & \\ & & & \boldsymbol{x}_T' \end{bmatrix} \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{G}_2 \\ \vdots \\ \boldsymbol{G}_T \end{bmatrix}$$

with

$$\boldsymbol{G}_t = \left[\mathbf{W}_t'\mathbf{X}\right]^{-1}\mathbf{W}_t'$$

depending on h through $\mathbf{W}_t$.

The optimal bandwidth parameter h_{opt} will be the minimiser of $AIC(h)$:

$$AIC(h) = \log(\hat{\sigma}^2) + \frac{2 * (T_h + 1)}{(T - T_h - 2)}$$

where $\hat{\sigma}^2 = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2$ and where T_h is the trace of the "smoothing matrix" $\boldsymbol{H}_h$.

3.2 Optimal bandwidth: automatic selection

To illustrate how automatic bandwidth selection works, consider step 2 of the example shown in Section 2. When the estimation method is OLS, automatic bandwidth selection is triggered by setting the bandwidth parameter h_1 to `NA`, or omit it altogether. In this case, the bandwidth parameter h_{opt} will be chosen by minimising the AIC criterion among a range of values between 0.01 and 0.99, and estimation will follow:

```
mod1 = tv_OLS(Cons, X, NA)
```

will produce the following output

```
ketvals: OLS estimation complete
Sample: 1984:1 - 2020:4 (148 observations)
Kernel type: Gaussian, bandwidth = 3.95673 (param = 0.275237)
Auto bandwidth selection (min = 0.01, max = 0.99), AIC = -11.412
Dependent variable: Cons
Explanatory variables:
        const time GDP Cons_1
```

The usage of the automatic bandwidth selection in the `tv_IV` function works in a similar manner but the user is asked to provide the bandwidth parameter for the first step (the input `h2`), so that

```
mod2 = tv_IV(Cons, X, Z, NA, 0.6)
```

will produce the output based on the optimal bandwidth, conditional on h_2 set by the user, which is

```
ketvals: IV estimation complete
Sample: 1984:1 - 2020:4 (148 observations)
Kernel type: Gaussian, bandwidth = 5.08321 (param = 0.32537)
Auto bandwidth selection (min = 0.01, max = 0.99), AIC = -11.3245
First stage bandwidth = 20.052 (param = 0.6)
Dependent variable: Cons
Explanatory variables:
        const time GDP Cons_1
Instruments:
        const time Inv Inv_1 Cons_1
```

The rest of the example follows in the usual way.

4 Sample script

The sample script provided with the package is a somewhat more elaborate version of the example provided in the previous subsection. In this case, we estimate by OLS a dynamic consumption function on European data, but we also demonstrate the usage of the dedicated plotting function and how to retrieve the internal elements of the model bundle for later use.

Consider the following model (in ECM form):

$$\Delta c_t = \beta_{1,t} + \beta_{2,t}\Delta c_{t-1} + \beta_{3,t}\Delta y_t + \beta_{4,t}\Delta y_{t-1} + \beta_{5,t}c_{t-1} + \beta_{6,t}y_{t-1} \tag{5}$$

where c_t is the log of private consumption and y_t is GDP. All the parameters are assumed to be time-varying. In this example, we will estimate the time-varying long run multiplier $\kappa_t = -\beta_{6,t}/\beta_{5,t}$, measuring the "steady-state" income elasticity of consumption and calculate its standard error via the delta method.

First we load the ketvals package by

```
include ketvals.gfn
```

and open the AWM dataset. We the create the needed variables and define the list of regressors for Equation (5) via standard gretl commands:

```
open AWM18.gdt --select="YER PCR"
y = log(YER)
c = log(PCR)
diff y c
smpl 1975:1 ;
list X = const d_c(-1) d_y(0 to -1) c(-1) y(-1)
```

We can now estimate the time-varying OLS coefficients by

```
mod = tv_OLS(d_c, X, 0.7)
```

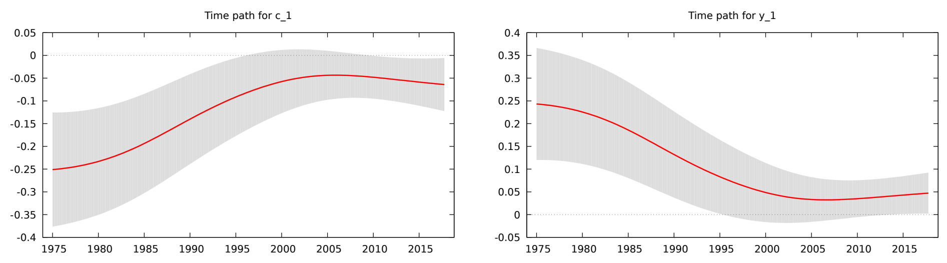
using, again, a Gaussian kernel with bandwidth parameter $h_1 = 0.7$. When estimation is complete we get the following message:

```
ketvals: OLS estimation complete
Sample: 1975:1 - 2017:4 (172 observations)
Kernel type: Gaussian, bandwidth = 36.7172 (param = 0.7)
Dependent variable: d_c
Explanatory variables:
        const d_c_1 d_y d_y_1 c_1 y_1
```

We can now plot the series of $\beta_{5,t}$ and $\beta_{6,t}$ by

```
coeff_plot(mod, 5)
coeff_plot(mod, 6)
```

Figure 2: Estimated Coefficients



and the result is reported in Figure 2. The confidence interval is set at its default value of 95%, but may be adjusted if necessary (see the documentation for the `coeff_plot` function).

Moreover, we extract the series of estimated coefficients, together with their standard errors, as series. Note that the series containing the standard errors must be given in pointer form, and hence must be declared separately.

```
series se_c1 = NA
series se_y1 = NA
b_c = coeff_save(mod, 5, &se_c1)
b_y = coeff_save(mod, 6, &se_y1)
```

To compute the long run multiplier, we use the formula $\kappa = -\frac{\beta_6}{\beta_5}$. Therefore, the time-varying long run multiplier can be immediately computed as

```
series lrm = - b_y/b_c
```

As for its standard error, we use the Delta Method: the asymptotic variance of κ_t is given by

$$V(\kappa_t) = R_t' V \left(\begin{bmatrix} \beta_{5,t} \\ \beta_{6,t} \end{bmatrix} \right) R_t$$

where R_t is the Jacobian term

$$R_t \equiv \begin{bmatrix} \frac{\partial \kappa_t}{\partial \beta_{5,t}} \\ \frac{\partial \kappa_t}{\partial \beta_{6,t}} \end{bmatrix} = - \begin{bmatrix} \kappa_t / \beta_{5,t} \\ 1/\beta_{5,t} \end{bmatrix} \tag{6}$$

The time-varying covariance matrix of the parameters will be present, in the model bundle, under the `vcv` key, as a $T \times (k \cdot (k+1)/2)$ matrix, whose t-th row is the covariance matrix of $\boldsymbol{\beta_t}$ in vech form. The following code fragment computes the standard errors for κ_t by extracting the suitable part of the covariance matrix at time t as V_t and computing the Jacobian term R_t; then, the standard error is computed as $se_t = \sqrt{R_t' V_t R_t}$.

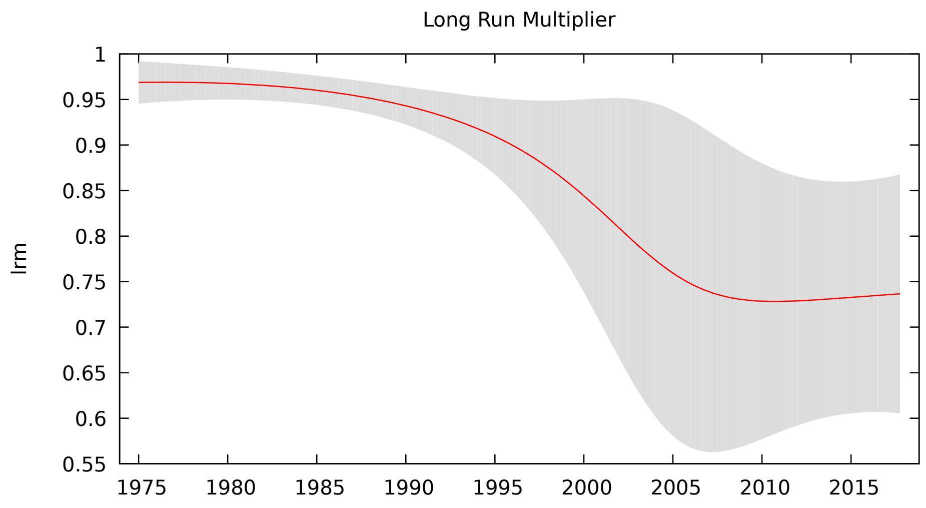


Figure 3: Estimated `lrm`

```
smpl mod.valid --restrict
series lrm_se = NA

i = 1
loop t = $t1 .. $t2
    matrix V = unvech(mod.vcv[i++,]')[5:6,5:6]
    matrix R = - {lrm[t], 1} / b_c[t]
    scalar vt = qform(R,V)
    lrm_se[t] = sqrt(vt)
endloop
smpl full
```

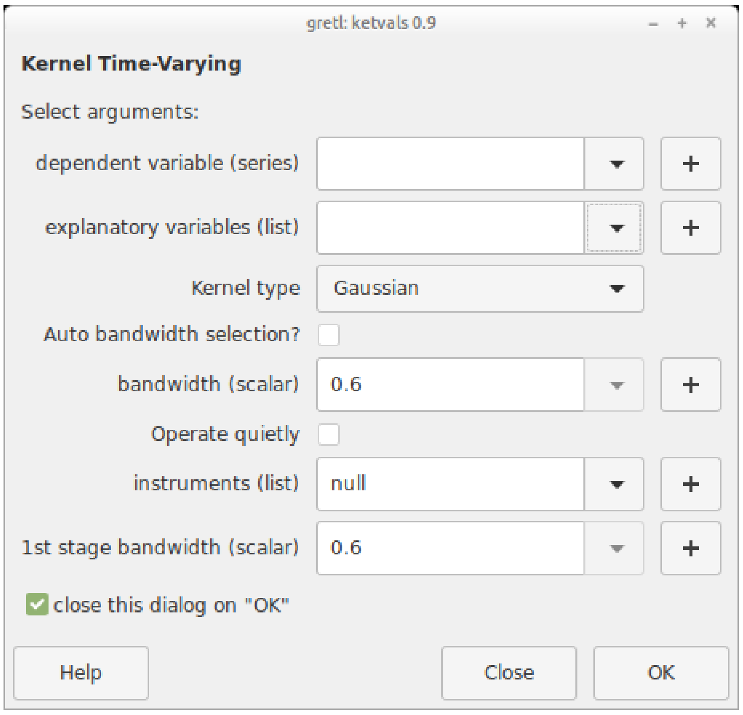
Finally, we plot `lrm` and its 90% Confidence Interval by

```
plot lrm
    options time-series with-lines
    options band=lrm,lrm_se,1.64 band-style=fill
    literal set title 'Long Run Multiplier'
end plot --output=display
```

and the output is reported in Figure 3.

5 GUI usage

A graphical interface is provided, under the "Model>Univariate time series" menu. See Figure 4. The input list should be reasonably obvious, given the discussion above.


gretl: ketvals 0.9

Kernel Time-Varying

Select arguments:

dependent variable (series)	
explanatory variables (list)	
Kernel type	Gaussian
Auto bandwidth selection?	☐
bandwidth (scalar)	0.6
Operate quietly	☐
instruments (list)	null
1st stage bandwidth (scalar)	0.6

✓ close this dialog on "OK"

Figure 4: The GUI interface

6 List of public functions

The package provides 5 public functions. Via scripting, the functions are the following:

- `tv_OLS`: this function provides the OLS estimator in Equation (2);
- `tv_IV`: this function provides the IV estimator in Equation (3), together with diagnostic tests;
- `coeff_plot`: easily plots time path for coefficients and standard errors;
- `coeff_save`: store series of coefficients and (optionally) their standard errors;
- `global_hausman_test`: reports the global Hausman test statistic.

6.1 The function `tv_OLS`

```
tv_OLS(series yy, list XX, scalar h1[NA], bool verbose[1], type[0:3:1],
matrix optparm[null])
```

Return type : `bundle`

`yy` : the dependent variable;

`XX` : the list of regressors;

`h1` : the scalar h_1 that governs the bandwidth H, automatically selected if not provided or omitted;

`verbose` : a Boolean controlling the output verbosity, `1` is the default;

`type` : a scalar denoting the kernel function, `1` is the default;

`optparm` : a matrix with the parameters of the exponential kernel, `null` is the default.

If the Boolean `verbose` switch is set to 1 (as by default), a short message is reported when the estimation process is complete. The scalar `type` can assume the following values:

1. Gaussian kernel: $K(x) = \exp(-x^2/2)$
2. Rolling window
3. Epanechnikov kernel: $K(x) = 0.75(1 - x^2)$ for $|x| < 1$
4. exponential kernel: $K(x) = \exp(-cx^\alpha)$

for the exponential kernel, c and α are set to 1 by default, but can be specified by the optional matrix argument `optparm`.
The output of the function is a bundle containing:

- `method`: the string `"OLS"`.
- `t1`: 1-based index of the first observation in the currently selected sample.
- `t2`: 1-based index of the last observation in the currently selected sample.
- `T`: actual sample size, T.
- `coeff`: a $T \times k$ matrix. Each column corresponds to the series of estimated coefficients for each regressor.
- `uhat`: the $T \times 1$ matrix of residuals.

- `valid`: a binary series denoting observations actually included in the estimation process (=1).
- `vcv`: a $T \times (k \cdot (k+1)/2)$ matrix. Each row is the vech of the parameters covariance matrix at time t.
- `stderr`: a $T \times k$ matrix. Each column corresponds to the series of estimated standard error for each regressor.
- `names`: an array of strings. The names of the covariates.
- `depvarname`: a string. The name of the dependent variable.
- `ktype`: a scalar. The kernel type (`type`).
- `bwid`: the bandwidth of the kernel function H
- `bw_parm`: the scalar h_1 that governs the bandwidth H (`h1`).

6.2 The function `tv_IV`

```
tv_OLS(series yy, list COVAR, list INST, scalar h1, scalar h2[NA],
bool verbose[1], type[0:3:1], matrix optparm[null])
```

Return type : `bundle`

`yy` : the dependent variable;

`COVAR` : the list of regressors;

`INST` : the list of instruments;

`h1` : the scalar h_1 that governs the bandwidth H; automatically selected if omitted, conditional on `h2`;

`h2` : the scalar h_2 that governs the bandwidth L; optional: it defaults at h_1 if omitted.

`verbose` : a Boolean controlling the output verbosity, `1` is the default;

`type` : a scalar denoting the kernel function, `1` is the default;

`optparm` : a matrix with the parameters of the exponential kernel, `null` is the default.

Please note that either `h1` and `h2` must be provided. For details on Boolean `verbose`, scalar `type` and matrix `optparm`, see `tv_OLS`.
The output of the function is a bundle containing:

- `method`: the string `"IV"`.
- `t1`: 1-based index of the first observation in the currently selected sample.
- `t2`: 1-based index of the last observation in the currently selected sample.
- `T`: actual sample size, T.
- `coeff`: a $T \times k$ matrix. Each column corresponds to the series of estimated coefficients for each regressor.
- `uhat`: the $T \times 1$ matrix of residuals of the "second step" equation.
- `valid`: a binary series denoting observations actually included in the estimation process (`=1`).
- `vcv`: a $T \times (k \cdot (k+1)/2)$ matrix. Each row is the vech of the parameters covariance matrix at time t.
- `stderr`: a $T \times k$ matrix. Each column corresponds to the series of estimated standard error for each regressor.
- `depvarname`: a string. The name of the dependent variable.
- `names`: an array of strings. The names of the covariates.
- `instnames`: an array of strings. The name of the instruments.
- `ktype`: a scalar. The kernel type (`type`).
- `bwid`: the bandwidth of the kernel function H
- `bw_parm`: the scalar h_1 that governs the bandwidth H (`h1`).
- `bw_parm2`: the scalar h_2 that governs the bandwidth L (`h2`).
- `Jstat`: a $T \times 1$ matrix, reporting the Over-identification J statistic for each period.
- `Jpval`: a $T \times 1$ matrix, reporting the p-value for the J statistic for each period.
- `Hstat`: a $T \times 1$ matrix, reporting the Hausman specification test statistic for each period.
- `Hpval`: a $T \times 1$ matrix, reporting the p-value for the Hausman test statistic for each period.
- `xfitted`: a $T \times k$ matrix, reporting the fitted values from the "first stage" regression by columns.
- `yhat`: a $T \times 1$ matrix, reporting the fitted values of the dependent variable.

6.3 The function `coeff_plot`

```
coeff_plot(bundle model, scalar v, scalar alpha[0.95],
string dest[null], matrix yr[null])
```

Return type : `void`

`model` : the bundle generated by `tv_OLS` or `tv_IV`;

`v` : a scalar denoting the element in the list of regressors;

`alpha` : a scalar between 0 and 1; the confidence level, `0.95` is the default;

`dest` : the string with the name of a graphic file where the plot is saved. If `null` (default), the series will be displayed.

`yr` : if provided, this matrix contains the maximum and the minimum values for the y-axis.

6.4 The function `coeff_save`

```
coeff_save(bundle model, scalar v, series *se[null])
```

Return type : `series`

`model` : the bundle generated by `tv_OLS` or `tv_IV`;

`v` : a scalar denoting the element in the list of regressors;

`*se[null]` : an existing series (see below). The default is `null`.

The output of the function is a series containing the values of the estimated coefficient. If provided, the values of the series `*se` will be replaced by the standard errors of the variable defined by `v`.

6.5 The function `global_hausman_test`

```
global_hausman_test(scalar T0, scalar T1, bundle beols, bundle beiv,
series yy, list COVAR, list INST, scalar type[0:3:1],
scalar h1, matrix optparm[null])
```

Return type : `bundle`

`T0` : a scalar denoting the starting observation;

`T1` : a scalar denoting the ending observation;

`beols` : the bundle generated by `tv_OLS`;

`beiv` : the bundle generated by `tv_IV`;

`yy` : the dependent variable;

`COVAR` : the list of regressors;

`INST` : the list of instruments;

`type` : a scalar denoting the kernel function, `1` is the default;

`h1` : the scalar h_1 that governs the bandwidth H;

`optparm` :a matrix with the parameters of the exponential kernel, `null` is the default.

For details on scalar `type`, see `tv_OLS`. The output of the function is a bundle containing:

- `ght_stat`: the Global Hausman test statistic, a scalar;
- `ght_pval`: the p-value for the Global Hausman test statistic, a scalar.

References

Cai, Z. (2007). Trending time-varying coefficient time series models with serially correlated errors. *Journal of Econometrics*, 136(1):163–188.

Giraitis, L., Kapetanios, G., and Marcellino, M. (2021). Time-varying instrumental variable estimation. *Journal of Econometrics*, 224(2):394–415.

Giraitis, L., Kapetanios, G., and Yates, T. (2014). Inference on stochastic time-varying coefficient models. *Journal of Econometrics*, 179(1):46–65.

Giraitis, L., Kapetanios, G., and Yates, T. (2018). Inference on multivariate heteroscedastic time varying random coefficient models. *Journal of Time Series Analysis*, 39(2):129–149.

Changelog

- `v1.1`: robustify treatment of the `dest` parameter to `coeff_plot` (could fail if filename contained spaces).
- `v1.0`: add optional parameter to `coeff_plot` for custom y-range in; bump version requirement to 2021c
- `v0.92`: fix bug with bundle initialisation in `global_hausman_test`
- `v0.91`: fix bug with IV auto bandwidth selection
- `v0.9`: add functionality for auto bandwidth selection and graphical interface; also, reorganize the code internally
- `v0.12`: pre-release