Research

Locproj & Lpgraph: Stata commands to estimate Local Projections

WP 25-09

Alfonso Ugarte-Ruiz

July 2025

LOCPROJ & LPGRAPH: Stata Commands to Estimate Local Projections

Alfonso Ugarte-Ruiz
BBVA Research
Madrid, Spain
alfonso.ugarte@bbva.com

Abstract. locproj estimates linear and nonlinear Impulse Response Functions (IRF) based on the local projections methodology first proposed by Jordà (2005). The procedure allows for the easy implementation of several options used in the growing literature of local projections. It reports the IRF, together with its standard error and confidence interval, as an output matrix and through an IRF graph. It allows for choosing different estimation methods for both time series and panel data, including instrumental variables and quantile regressions. It also allows for the use of marginal effects instead of regression coefficients, which is highly convenient when the response variable is binary and the user wants to estimate the response as a probability. We also show some cases in which it can be used to estimate an Event Study (DiD) regression. The user can easily choose different options for the desired IRF graph, as well as several other options to save and use the results. The module also includes a post-estimation command called lpgraph that can plot IRFs estimated by different models.

Keywords: locproj, lpgraph, local projection, IRFs, quantile regression, instrumental variables, D-i-D, event study, margins, binary dependent variable

1 Introduction

Local projections (LPs) have rapidly gained prominence in macroeconomic and applied econometrics literature due to their flexibility, robustness, and intuitive appeal in estimating impulse response functions (IRF). They have been employed in a wide variety of empirical applications, ranging from evaluating monetary interventions and fiscal consolidations to assessing the macroeconomic effects of financial crises, climate shocks, and carbon taxes.

Despite their growing popularity and broad range of applications, until recently there was no dedicated Stata package for estimating impulse responses using LPs. This may be partly attributed to the perceived simplicity of LP computation, which—depending on the modeling assumptions—involves estimating a series of OLS regressions. However, implementing LPs in practice requires a sequence of iterative steps, which can become cumbersome, particularly when exploring various methodological choices or model specifications.

This paper formally introduces the package `locproj`[1], which enables the estimation

1. The package LOCPROJ was presented in the US Stata Conference at Stanford in July 2023 and it

of linear and nonlinear IRF, focusing on describing how to implement some of the methodological alternatives used in the fast-growing LP literature that are not available in any other automatized procedure, either in Stata or elsewhere.

Starting with version 18.0, Stata featured its own native commands called `lpirf`, followed by the command `ivlpirf` in Stata 19.0. Estimation in the former assumes that the shock is part of a system of equations, which offers some advantages but also narrows the methodological alternatives. Neither command allows for the use of panel data methods. `locproj` relies on the assumption that the shock of interest is exogenous, has been previously identified, or that a set of instruments is available otherwise, which simplifies the estimation and opens up a wider set of methodological possibilities.

Dube et al. (2025) have also developed a new Stata command `lpdid` based on their proposed LP approach to Difference-in-Differences Event Studies. Although `lpdid` is more general and can estimate more cases within the DiD framework, `locproj` can be used in a wider range of LP applications beyond DiD. Moreover, we show how `locproj` can also be used in some particular cases to compute the LP-DID estimator.

Outside of Stata, the R-package lpirfs (Adammer (2019)) also allows the use of panel data and of instrumental variables methods and nonlinear options. However, `locproj` has a much larger set of methodological options and capabilities, some of them based on the higher flexibility of Stata for handling factor variables and time-series operators. `locproj` also has a wide range of tools that facilitate the analysis to the researcher that are not available in either Stata nor the R packages.

The `locproj` package also includes a post-estimation command called `lpgraph` that plots the results of previously estimated IRFs of more than one model into one single graph that can include up to four different IRFs, which is highly convenient when we want to compare the magnitude and the dynamics of the different IRFs, since they share the same axis.

`lpgraph` is a post-estimation command, and it uses the IRF results saved as variables by the command `locproj`. The command `lpgraph` can also be used to combine IRF results from other estimation methods, such as VAR, SVAR, arima, etc., as long as the results of those commands are saved with the same name structure in which `locproj` saves the IRFs results.

In the following section, we briefly discuss several methodological possibilities available in `locproj` such as:

- Choosing the best transformation: levels, differences or long-differences
- Panel data
- Instrumental variables
- Nonlinearities

has been available in the Repec-Ideas Stata Repository since May the same year.

- State-dependent LP
- Binary dependent variable
- Quantile LP
- Starting periods different than zero
- Difference-in-Differences & event studies

Then, in section 3 and section 4, we give a more detailed description of the syntax and options for the `locproj` command and the `lpgraph` command, respectively. Finally, in section 5 we illustrate usage of the package in several examples covering most of the topics discussed in section 2.

2 Methods

Following Jordà and Taylor (2024), we are interested in characterizing how an intervention today affects the average outcome at some time in the future relative to a baseline of no-intervention. Let y_t denote an outcome variable of interest, s_t the policy intervention variable, and let x_t denote a vector of controls variables. Formally, we define an impulse response as:

$$\mathcal{R}_{s\to h} = E[y_{t+h}|s_t = s_0 + \delta; x_t] - E[y_{t+h}|s_t = s_0; x_t]; \qquad h = 0, 1, ..., H, \tag{1}$$

where s_0 denotes the value of the variable s_t without intervention and δ is the size of the intervention, which is commonly normalized so that $\delta = 1$. This allows to omit δ from the notation and write $\mathcal{R}_{s\to h}(h, 1) \equiv \mathcal{R}_{s\to h}(h) \equiv \mathcal{R}_{sh}(h)$.

According to Jordà (2005), the local projection or LP of y_{t+h} on s_t can be estimated with the following regressions:

$$y_{t+h} = \alpha_h + \beta_h s_t + \gamma_h' x_t + \nu_{t+h}; \qquad h = 0, 1, ..., H \tag{2}$$

With $\mathcal{R}_{sh}(h) = \beta_h$. Initially, we assume that $E[s_t|\nu_{t+h}] = 0$, i.e. that s_t is exogenous.

Estimating the IRF through local projections is straightforward in most cases since β_h can be estimated directly through OLS. The usual steps for estimating an IRF through local projections imply:

1. Define the length of the estimation horizon (H) and the type of transformation of the dependent variable, usually:
    - Levels vs logarithms
    - Levels vs differences
    - differences vs log-differences
    - Levels vs. cumulative (long) differences

2. Construct a loop that runs the h-steps regressions using the desired estimation method and specification.
3. Extract and save the h-step estimated coefficient and standard error of the "shock" variable (or possibly more than one coefficient in nonlinear cases).
4. Construct confidence intervals.
5. Graph the IRF.

The command `locproj` automates all these necessary steps, allowing the user to focus on choosing the best specification and analyzing the results. In the rest of this section we review the main methodological options where `locproj` can be more useful.

2.1 Choosing the best transformation: Levels, differences, or long-differences

In practice we have several choices of LP specification to estimate impulse responses. In a stationary case, we can choose between levels or long-differences, whereas we can choose to differentiate if our dependent variable is non-stationary or if we are simply interested in analyzing the short-term evolution of the IRF. We also can choose between levels or taking logarithm before differentiating.

- Level: $y_{t+h} = \alpha_h^L + \beta_h^L s_t + \gamma_h'^L y_{t-1} + \nu_{t+h}$
- Differences: $y_{t+h} - y_{t+h-1} = \alpha_h^D + \beta_h^D s_t + \gamma_h'^D (y_{t-1} - y_{t-2}) + \nu_{t+h}$
- Long-differences: $y_{t+h} - y_{t-1} = \alpha_h^{LD} + \beta_h^{LD} s_t + \gamma_h'^{LD} (y_{t-1} - y_{t-2}) + \nu_{t+h}$

Moreover, `locproj` also includes the option of taking logs before differencing. Usually, the loop may require generating some new variables, for instance, the dependent variable at the different forecast steps or just the simple or log difference. For every transformation option, the command `locproj` generates all the necessary new (temporary) variables according to each transformation option. It also generates temporary variables with the corresponding transformation of the dependent variable needed in case the user wants to include lags of the dependent variable that are consistent with the chosen transformation.

2.2 Panel data

One of the key advantages of LPs is that they can be immediately extended to a panel data setting, which, in addition to having potentially more observations with which to increase the precision of the estimates, opens a large set of methodological possibilities like different types of nonlinearities, estimating the IRF of the probability of an event or estimating a difference-in-differences analysis. A typical panel data local projection could be specified as:

$$y_{i,t+h} = \alpha_{i,h} + \mu_{t,h} + \beta_h s_{i,t} + \gamma_h' x_{i,t} + \nu_{i,t+h} \tag{3}$$

where $\alpha_{i,h}$ denotes (cross-section) random or fixed effects and $\mu_{t,h}$ denotes time-fixed effects. `locproj` can adjust almost any panel data estimation method available in Stata, and it automatically switches from OLS (`reg`) to GLS (`xtreg`) as the default estimation method if the dataset has been defined with `xtset`.

The use of different robust standard errors estimators to adjust for heteroskedasticity and autocorrelation are highly recommended and supported by most panel data commands in Stata and thus in `locproj`.

2.3 Instrumental variables

Identification of LPs via the use of instrumental variables is a well established method, referred to as LP-IV (since its first appearance in Jordà et al. (2015)). As is usually the case with instrumental variables, conditions must be satisfied. One will need a relevance assumption (that is, the instrument is correlated with the endogenous variable) and an exogeneity assumption (the instrument is uncorrelated with the residuals).

`locproj` allows the use of different instrumental variable methods currently available in Stata such as `ivregress`, `xtivreg` and `ivqregress`.

One advantage of `locproj` is that it offers the option of performing and displaying the results of some of the tests that are available after using the commands `ivregress` and `ivqregress`. For instance, we can test for overidentifying restrictions and endogeneity after every step of the LP.

2.4 Nonlinearities

The possibilities of different types of nonlinearities in local projections are plentiful and the applications are growing quickly. The most basic case is when the effect of the shock is nonlinear in the sense that it can be different at different levels of the shock variable, such as when we include a quadratic term:

$$y_{t+h} = \alpha_h + \beta_h^a s_t + \beta_h^b s_t^2 + \gamma_h' x_t + \nu_{t+h} \tag{4}$$

In this case the IRF would be $\mathcal{R}_{sh}(h) = \beta_h^a + \beta_h^b$ if the size of the shock is normalized to one, i.e. $\delta = 1$, and the initial shock level is zero, i.e. $s_0 = 0$. Using (4) in (1), we get that in general the response depends on the initial level of the variable, s_0, and the size of the intervention, δ, in the following way:

$$\mathcal{R}_{sh}(h) = \beta_h^a(s_0 + \delta) + \beta_h^b(s_0 + \delta)^2 + \gamma_h' x_t - [\beta_h^a(s_0) + \beta_h^b(s_0)^2 + \gamma_h' x_t)]$$

$$\mathcal{R}_{sh}(h) = \beta_h^a \delta + 2\beta_h^b s_0 \delta + \beta_h^b \delta^2 \tag{5}$$

2.5 State-dependent LPs

One of the most common nonlinearities is that of a state-dependent LP, when the shock interacts with another variable that defines a state, which is usually characterized by a binary (dummy) variable $D_t \in \{0, 1\}$ that defines whether a state is active or not. Depending on how we would want to describe the impact, if either as a difference with respect to the other state or as an absolute impact at each one of the two states, then we can specify the estimation as either (6) or as (7) respectively.

$$y_{t+h} = \alpha_h + \beta_h^a s_t + \beta_h^b D_t s_t + \gamma_h' x_t + \nu_{t+h} \tag{6}$$

$$y_{t+h} = \alpha_h + \beta_h^a (D_t = 0) s_t + \beta_h^b (D_t = 1) s_t + \gamma_h' x_t + \nu_{t+h} \tag{7}$$

If the IRF is expressed as a difference it would be given by $\mathcal{R}_{sh}(h) = \beta_h^b$ from equation (6) or if the IRF is expressed as the total impact, it would be given by $\mathcal{R}_{sh}(h) = \beta_h^a + \beta_h^b$ from equation (6) or by $\mathcal{R}_{sh}(h) = \beta_h^b$ from equation (7).

Obviously, there could be more than two states, or we can even interact our shock variable with a continuous variable that alters the reaction of our outcome variable.

$$y_{t+h} = \alpha_h + \beta_h^a s_t + \beta_h^b s_t * z_t + \gamma_h' x_t + \nu_{t+h} \tag{8}$$

In this case the IRF would be given by $\mathcal{R}_{sh}(h) = \beta_h^a + \beta_h^b$ if the size of the shock and the interaction variable are normalized. In a more general way,

$$\mathcal{R}_{sh}(h) = \beta_h^a \delta + \beta_h^b \delta * \theta \tag{9}$$

where δ is the size of the shock and we want to evaluate the IRF at $z_t = \theta$

2.6 Binary dependent variable

Local projections can also be estimated when the outcome variable is binary (Barattieri and Cacciatore (2023)) and the user wants to estimate the response as a probability. As in other cases of nonlinearities, the response depends on the level of the shock and of other variables. Therefore, it is easier to rely on the existing Stata command margins, more specifically, using the option of obtaining margins of derivatives of responses (a.k.a. marginal effects).

In this case, a practitioner may be interested in the probability of an outcome at some point in the future if there is a shock to a variable today. Then Equation 1 could be redefined as follows:

$$\mathcal{R}_{s \to h} = P[y_{t+h} = 1 | s_t = s_0 + \delta; x_t] - P[y_{t+h} = 1 | s_t = s_0; x_t]; \qquad h = 0, 1, ..., H, \tag{10}$$

This IRF could be estimated through LP with simple logit or probit models both in a time-series or a panel-data framework. `locproj` can accommodate several commands like `probit`, `logit`, `xtprobit`, `xtlogit`, `oprobit`, `ologit`, etc. Moreover, `locproj` has the option of expressing the IRF as the response of the probability of a positive outcome, exactly as expressed in 10, using the Stata command `margins`.

2.7 Quantile local projections

One possibility that has been explored in Makabe et al. (2022) and Jordà et al. (2022)) is that a shock may have no visible effects on the average outcome, but it may have considerable impact the tails of its distribution.

The approach to calculate quantile local projections is parallel to the way local projections are computed at the mean, as in equation (2). The only difference is that we are now dealing with a nonlinear model so the marginal effect of a change in the shock has to be evaluated accordingly.

In the case of `locproj` we only need to change the estimation method to `qreg`, and we can also use the IV case, `ivqreg`.

One advantage of `locproj` is that it has been adapted so that we can include lags of the dependent variable and the shock variable automatically even if the estimating method we are using does not allow time-series operators, as is the case with the commands `qreg` and `ivqreg`.

The command `lpgraph` is also well-suited to estimating quantile LP since normally we want to estimate and compare multiple IRFs at different moments of the distribution.

2.8 Starting periods different than zero

Throughout this section, we have specified that the estimation horizon h starts at the same moment that the shock occurs, i.e at $h = 0$. But, more generally, we can assume that the response of our outcome variable starts at a period before the initial shock, in which case we could have $h < 0$. This can be particularly useful when we want to evaluate whether there are anticipated responses or if we want to test the parallel trends hypothesis in a Difference-in-Differences estimation.

`locproj` allows having initial steps different from zero, including negative starting horizons, automatically adjusting the output and the way in which the lags of the dependent variable are included. `locproj` also adjusts the output when the shock variable is included with a lag, which happens when we assume that the impact on the outcome variable shows up with a delay.

In the cases of negative horizons and long-differences (cumulative), `locproj` also automatically adjusts the way in which the lags of the dependent variable are included if they are required by the user, since the lags of the dependent variable have to obviously predate the first period included in the calculation of the long-difference.

2.9 Difference-in-Differences & event studies

One of the most interesting applications of LP is the possibility of extending the D-i-D estimator to cases when there are more than two periods and two groups. In such cases, treatment effects may vary across groups depending on when treatment is received (i.e., they are heterogeneous) and the effects may also change over time after treatment (i.e., they are dynamic). Dube et al. (2025) show that most of the extensions to the basic two period, two groups setting can be accommodated with a simple modification of an LP estimator under standard assumptions.

Dube et al. (2025) have proposed a general estimator called LP-DiD estimator that can be expressed as:

$$y_{i,t+h} - y_{i,t-1} = \delta_{th} + \beta_h \Delta s_{i,t} + \sum_{j=1}^{p} \rho_{ij} \Delta y_{i,t-j} + \gamma_h x_{i,t} + \nu_{i,t+h}, \tag{11}$$

where δ_{th} are time fixed effects and where the estimation sample is restricted to observations that correspond to either $\Delta s_{i,t} = 1$ (newly treated units), or $s_{i,t+h} = 0$ (not yet treated units) to ensure "clean controls". Dube et al. (2025) have also created and shared the Stata command `lpdid`.

Although `lpdid` is currently the most appropriate command for estimating the Dube et al. (2025) estimator, `locproj` can be used perfectly to estimate the D-i-D cases of only two groups (treated and control), more than two periods ($t > 2$), and when treatment occurs at the exact same period for all treated individuals. Under those assumptions and given its flexibility, `locproj` is a convenient alternative to other event study estimators.

The population coefficient β_h^{LP} from an LP regression corresponds exactly to the estimand for the dynamic ATT h periods after treatment, which is a particular case of 11.

In cases where treatment occurs at the same period $t = t^*$, we can estimate the following:

$$y_{i,t^*+h} - y_{i,t^*-1} = \delta_h + \beta_h^{LP} \Delta s_{i,t^*} + \nu_{i,t^*+h}, \qquad h = -Q, .., 0, ...H \tag{12}$$

where $Q > 0$.

The specification in Equation (12) does not include individual fixed effects, since they are eliminated by long-differencing. We can also add control variables and lags of the dependent variable as in Equation 11.

Later on, in the Examples section, we will show that estimating Equation 12 with `locproj` is equivalent to estimating the common Event Study approach with treatment starting at $t = q$ for all units, which following Wooldridge (2021) would be given by:

$$y_{i,t} = \eta + \lambda D_i + \sum_{s=1}^{q-2} \gamma_s (D_i * fs_t) + \sum_{s=q}^{T} \delta_s (D_i * fs_t) + \theta_2 f2_t + ...\theta_T fT_t + U_{it} \tag{13}$$

Where $f1_t, f2_t, ..., fT_t$ are time dummies for each period, $D_i = 1$ if unit i is ever treated, and the coefficient on $D_i * f(q-1)_t$ (period just before treatment) is normalized to zero.

3 The locproj command

`locproj` reports the IRF, together with its standard error and confidence interval, as an output matrix and through an IRF graph. The user can easily choose different options for the desired IRF graph, as well as other options to save and use the results.

`locproj` uses the Stata command `lincom` to estimate the response to the shock variable or variables, allowing to estimate responses to linear combinations of variables, including interactions with factor or continuous variables. Importantly, it also allows for the use of marginal effects instead of regression coefficients, which is highly convenient when the response variable is binary and the user wants to estimate the response as a probability. In the latter case, `locproj` makes use of the Stata command margins, which could also facilitate the estimation of responses when the shock corresponds to an interaction of variables (factor or continuous) instead of just a single variable.

The options allow defining the desired specification in a fully automatic or in a more explicit way, with many alternatives in between. If the user chooses the automatic specification, the syntax is very close to a typical regression command in Stata, with the only restriction that `locproj` interprets the variable that corresponds to the shock (impulse) as the one just after the dependent variable or its lagged terms, and only that one variable represents the shock.

Alternatively, the user can choose to explicitly define the shock variable (or variables), the number of lags of the shock, the number of lags of the dependent variable, and the control variables. As mentioned before, the user can play with alternatives between the fully automatic or the fully explicit, depending on which option is easier or more convenient to use.

The explicit option is recommended when the shock should include more than one variable, for instance, an additional nonlinear term, or an interaction with another variable. Moreover, if the shock includes an interaction with a categorical variable, then we must use one of the options `lcs()` or `margins`, unless the categorical variable is treated as continuous.

The `locproj` command has the following syntax:

Automatic Specification (Shock and Lags)

`locproj` *depvar shock* [*depvar lagged-terms*] [*shock lagged-terms*] [*controls*] [*if*]
[*in*] [*weight*] [, `hor(`*numlist integer*`)` `lcs(`*string*`)` `lcopt(`*string*`)`
`fcontrols(`*varlist*`)` `instr(`*string*`)` `transf(`*string*`)` `met(`*string*`)` `hopt(`*string*`)`
`conf(`*numlist integer*`)` `noisily` `stats` `saveirf` `irfname(`*string*`)` `fact(real)`
`margins` `mrfvar(`*varlist*`)` `mrpred(`*string*`)` `mropt(`*string*`)` `nograph` `title(`*string*`)`
`label(`*string*`)` `zero` `lcolor(`*string*`)` `ttitle(`*string*`)` `grname(`*string*`)`
`grsave(`*string*`)` `as(`*string*`)` `gropt(`*string*`)` `ivtest(`*string*`)` *model_options*]

Explicit Specification (Shock and Lags)

`locproj` *depvar* [*if*] [*in*] [*weight*] [, `hor(`*numlist integer*`)` `shock(`*varlist*`)` `controls(`*varlist*`)` `ylags(`*integer*`)` `slags(`*integer*`)` `lcs(`*string*`)` `lcopt(`*string*`)` `fcontrols(`*varlist*`)` `instr(`*string*`)` `transf(`*string*`)` `met(`*string*`)` `hopt(`*string*`)` `conf(`*numlist integer*`)` `noisily` `stats` `saveirf` `irfname(`*string*`)` `fact(real)` `margins` `mrfvar(`*varlist*`)` `mrpred(`*string*`)` `mropt(`*string*`)` `nograph` `title(`*string*`)` `label(`*string*`)` `zero` `lcolor(`*string*`)` `ttitle(`*string*`)` `grname(`*string*`)` `grsave(`*string*`)` `as(`*string*`)` `gropt(`*string*`)` `ivtest(`*string*`)` *model_options*]

Model specification options

`hor(`*numlist integer*`)` Specifies the number of steps or horizon length for the IRF. The initial horizon could be negative. It can be specified either as a range (e.g. hor(0/6) or hor(-3/6)), or just as the final horizon period (e.g. hor(6)) in which case the command assumes the horizon starts at period 0 and ends in period 6. The default horizon range is $hor = 0, ..., 5$ if nothing is specified.

`shock(`*varlist*`)` Allows to explicitly define the variable or variables that represent the shock or impulse that will generate the response and the IRF. If this option is not specified, the command will automatically choose the first variable that is immediately after the depvar and its lagged terms if they are included in the main varlist. This option should be used when the desired shock includes more than one variable, for instance a non-linear term or an interaction term.

`lcs(`*string*`)` Specifies an expression, usually an addition of variables, that defines a linear combination of variables that represents the desired impulse (shock). This option should be used when the desired shock includes more than one variable and the name of one of them is not explicitly included in the syntax variable list, for instance the constant term _cons), or the expansion of an expression that includes factor variable terms, e.g. 12.code#c.xvar. The expression that should go inside the parenthesis is analogous to any expression that is tested using the commands `lincom` or `test`.

`slags(`*integer)*`)` Specifies explicitly the number of lags of the shock variable or variables that should be included in the specification. The lagged terms of the shock could also be included directly in the main varlist next to the first variable that represents the shock. If more than one variable is specified through the option shock() then the specification will include lags of all of them.

`ylags(`*integer)*`)` Specifies explicitly the number of lags of the depvar that should be included in the specification. The way the lags of the dependent variable are included changes depending on the type of transformation that is defined by the user through the option transf().

`controls` Allows to explicitly define the variable or variables that represent the control variables. If this option is not specified, the command simply includes all the variables that are immediately after the shock variable(s) and its lagged terms if they are included in the main varlist. The control variables could include any number of lags, interactions, or any other desired transformations.

`fcontrols(`*varlist*`)` Specifies any control variable(s) that should be included at the same horizon as the IRF, i.e. that their forecast should be included depending on the horizon, i.e. fcontrol(t+h) with h = 0...hor.

`lcopt(`*string*`)` Specifies any option available in the command `lincom`.

Transformation options

`transf(string)` Specifies the type of transformation that should be applied to the dependent variable when generating the forecasts that are used for each horizon of the local projection. The available transformations are the ones in the following list, and they should be written exactly as they are shown:

1. *(level)* Levels: It keeps the dependent variable as originally specified and uses its forecast h periods ahead for each horizon of the IRF, i.e. y_{t+h} with $h = 0...hor$. It is the default option in case no transformation is specified. When the option ylags is specified, it includes lags of the variable in levels, i.e. y_{t-l} with $l = 1, ..., ylags$

2. *(diff)* Differences: It uses forecasts of the dependent variable in simple "differences", i.e. $y_{t+h} - y_{t+h-1}$ with $h = 0...hor$. When the option ylags() is specified, it includes lags of the variable in differences, i.e. $y_t - y_{t-1}$ with $l = 1, ..., ylags$

3. *(cmlt)* Long-term differences: It uses forecasts of the dependent varible in cumulative differences, i.e. $y_{t+h} - y_{t-1}$ with $h = 0...hor$. When the option ylags() is specified, it includes lags of the variable in differences, i.e. $y_t - y_{t-1}$ with $l = 1, ..., ylags$

4. *(logs)* Logs: It uses forecasts of the logarithm of the dependent varible, i.e. $ln(y_{t+h})$ with $h = 0...hor$. When the option ylags is specified, it includes lags of the logarithm of the variable, i.e. $ln(y_{t-1})$ with $l = 1, ..., ylags$

5. *(logs diff)* Log-differences: It uses forecasts of the dependent variable in differences of its natural logarithm, i.e. $ln(y_{t+h}) - ln(y_{t+h-1})$ with $h = 0...hor$. When the option ylags() is specified, it includes lags of the variable in log-differences, i.e. $ln(y_t) - ln(y_{t-1})$ with $l = 1, ..., ylags$

6. *(logs cmlt)* Cumulative log-differences: It uses forecasts of the dependent variable in cumulative differences of its natural logarithm, i.e. $ln(y_{t+h}) - ln(y_{t-1})$ with $h = 0...hor$. When the option ylags is specified, it includes lags of the variable in log-differences, i.e.$ln(y_t) - ln(y_{t-1})$ with $l = 1, ..., ylags$

Marginal effects options

`margins` Specifies that the marginal effect of the shock variable is used instead of the regression coefficients. For simplicity, it only allows using the dydx option of the command `margins`.

`mrfvar(`*varlist*`)` Specifies the factor or continuous variable that is interacted with the shock variable in the specification. This option should be used together with the shock(varlist) option and the margins option.

`mrpred(`*string*`)` Specifies the option to be used with the predict command to produce the variable that will be used as the response when using the margins option, e.g. pr, pc1, pu0, xb. It this option is not specified it uses the default option of the estimation method being used.

`mropt(`*string*`)` Allows to specify other options available in the command margins that have not been specified in the previous marginal effect options. See margins for specific help about using the command margins.

Estimation method options

`met(`*string*`)` Specifies the estimation method. The default is xtreg when using panel data and reg when using time-series data. Any estimation method with a standard syntax is allowed. Additionally, the command allows for the use of instrumental variable commands ivregress and xtivreg and other IV methods with a similar syntax. In the specific case of ivregress, the user also has to specify the "estimator" within the met() option in the following way: met(ivregress estimator), where estimator could be either on o 2sls, liml or gmm. When any IV method is specified, a list of instruments must also be provided through the option instr(*varlist*).

`instr(`*varlist*`)` Specifies the variables to use as instruments for the impulse (shock) variable when using an instrumental variable method such as ivregress or xtivreg. The shock variable must be defined as in any of the model specification available options.

`hopt(`*string*`)` Specifies any methodological option that depends directly on the horizon of the IRF, i.e. any option that must change with every step/horizon of the IRF $h = 0...hor$.

model_options Specifies any other estimation options specific to the method used and not defined elsewhere. If the user wants to specify any methodological option corresponding to the estimation method being used, she only has to enter them alongside the rest of `locproj` options.

Displaying results options

`noisily` If this option is specified, the command displays a regression output for each horizon. If this option is not specified, the command only returns a matrix with the IRF, its standard error and the confidence bands.

`stats` If this option is specified, the command displays a table with the summary statistics of the estimated regression at each step/horizon. The table includes the number of observations, the R-squared or pseudo-R-squared, the F-statistic or Chi2-statistic, and the p-value (prob) of the respective statistic.

IRFs options

`conf(`*numlist)* Specifies one or (max) two confidence levels for calculating the confidence bands. The default is 95%.

`saveirf` If this option is specified, the IRF, its standard error, and the confidence bands are saved as new variables, otherwise no new variables are created. If this option is specified, the command assigns a default name to the new generated variables.

`irfname(`*string*`)` Specifies a name/prefix for the new IRF variable and the other new generated variables (standard error and confidence bands).

`fact(real)` Specifies a factor to scale the IRF. For example, if the user wants to express the log-difference transformation in percentage terms, this option should be specified as fact(100).

Graphs options

`nograph` If this option is specified, a graph is not displayed.

`zero` If this option is specified, the graph includes a dashed line for the value 0.

`title(`*string*`)` Specifies a title for the IRF graph.

`lcolor(`*string*`)` Specifies a color for the IRF line and the confidence bands.

`label(`*string*`)` Specifies a label for the IRF line in the IRF graph.

`ttitle(`*string*`)` Specifies a name for the time axis in the IRF graph.

`grname(`*string*`)` Specifies a graph name that could be used, for instance, when combining various graphs.

`grsave(`*string*`)` Specifies a file name and path that should be used to save the IRF graph on disk.

`as(`*string*`)` Specifies the desired file format of the saved graph.

`gropt(`*string*`)` Specifies any other graph options not defined elsewhere.

Instrumental variable tests

`ivtest(`*string*`)` If this option is specified, the command performs and displays one of the three post-estimation tests available after using the command ivregress, for each step/horizon. The three available tests are the endogeneity test (endogenous), first-stage regression statistics (firststage), and the test of overindentifying restrictions (overid). The user has to write inside the parenthesis the exact name of the test, eithe endogenous, firststage or overid, and if necessary/desired, a comma and the corresponding/available test options.

Saved Results

`locproj` stores the following in `e()`:

Matrices	
`e(irf)`	A matrix including the Impulse Response Function (IRF), its standard error and its confidence interval
`e(stats)`	A matrix including the regression statistics at each step/horizon.
Variables if no name is given	
`_birf`	estimated impulse response function (IRF)
`_seirf`	IRF's standar error
`_irfup`	IRF's upper confidence interval
`_irflo`	IRF's lower confidence interval
`_irfup2`	second IRF's upper confidence interval
`_irflo2`	second IRF's lower confidence interval
Variables if name given to the IRF is *irfname*	
`irfname`	estimated impulse response function (IRF)
`irfname_se`	IRF's standar error
`irfname_up`	IRF's upper confidence interval
`irfname_lo`	IRF's lower confidence interval
`irfname_up2`	second IRF's upper confidence interval
`irfname_lo2`	second IRF's lower confidence interval

4 The lpgraph command

`lpgraph` plots together the results of previously estimated IRFs of more than one model in one graph. The graph can include up to four IRFs. It can also create four separate IRF graphs and combine them in one in the same way as the `graph combine`.

The first option is convenient when we want a graph that compares the magnitudes of the different IRFs, since they share the same axis. The second option is more convenient when you want separate IRF graphs of previously estimated and saved results, and then combine them into a single graph.

`lpgraph` has the following syntax:

`lpgraph` *irfname1 irfname2 irfname3 irfname4* [`,` `hor(`*numlist integer*`)` `separate` `zero` *lab1(string) lab2(string) lab3(string) lab4(string) title(string) ti1(string) ti2(string) ti3(string) ti4(string) ttitle(string) ytitle(string) lcolor(string) lc1(string) lc2(string) lc3(string) lc4(string) nolegend grname(string) grsave(string) as(string)* `combine(`*string*`)` *other_options*]

Options

`hor(`*numlist integer*`)` Specifies the number of steps or horizon length for the IRF. The initial horizon could be negative. It can be specified either as a range (e.g. hor(0/6) or hor(-3/6)), or just as the final horizon period (e.g. hor(6)) in which case the command assumes the horizon starts at period 0 and ends in period 6. The default horizon range is $hor = 0, ..., 5$ if nothing is specified.

`separate` If this option is specified, each IRF is plotted in a separate graph and then all are combined in a new graph, in the same way as if we were combining them using `graph combine`. The only difference between using the options separate and `graph combine` is that `lpgraph` will first generate the new graphs for each IRF.

`zero` If this option is specified, the graphs include a dashed line for the value 0.

`lab#(`*string*`)` Specifies a label for each IRF, e.g. lab1 `(Label A)`, lab2 `(Label B)`, lab3 `(Label C)` and lab4 `(Label D)`.

nolegend Specifies that legends should not be shown. It could be useful if you have separate graphs and each one of them has a title.

`title(`*string*`)` Specifies a title for the final graph.

`ti#(`*string*`)` Specifies a title for each graph, e.g., `ti1(Title A)`, `ti2(Title B)`, `ti3(Title C)`, `ti4(Title D)`. These options should be used when using the option separate.

`lcolor(`*string*`)` Specifies a unique color for the IRF line and the confidence bands of each one of the IRFs.

`lc#(`*string*`)` Specifies a color for the IRF line and confidence bands of each one of the IRFs (up to four), e.g., lc1(gray), lc2(green), lc3(blue), lc4(red). These options should be used when using the option separate.

`ttitle(`*string*`)` Specifies a name for the time axis in the IRF graph.

`ytitle(`*string*`)` Specifies a name for the y-axis in the IRF graph.

`grname(`*string*`)` Specifies a graph name that could be used, for instance, when combining various graphs.

`grsave(`*string*`)` Specifies a file name and path that should be used to save the IRF graph to a disk.

`as(`*string*`)` Specifies the desired file format of the saved graph.

other_options Specifies any other graph options not defined elsewhere. The user only needs to enter any other graph option not included in the list before alongside the rest of `lpgraph` options.

`combine(`*string*`)` Specifies any other options specific to the `graph combine` command not defined elsewhere. The user only needs to enter any other graph combine option not included before inside the option parenthesis.

5 Examples

This section provides several examples of `locproj` usage and functionality. All the examples presented here have only a didactic purpose, and are shown to demonstrate how to use `locproj` and `lpgraph` and not because of their economic or statistical meaning or relevance. Many of the examples are probably flawed from a statistical point of view.

5.1 Defining the basic options

In examples 5.1.1 to 5.1.8, we are interested in estimating the IRF from a shock to the variable n (Growth rate of hours worked) into the variable y (Growth rate of real GDP)

```
. webuse "https://www.stata-press.com/data/r17/usmacro2.dta"
```

or

```
.  use usmacro2.dta
```

5.1.1 Specification of shock and control variables

What the automatic vs. explicit specification means is that the user can let `locproj` interpret in an automatic way which variables provided on the varlist correspond to the dependent variable, which one is the shock, which ones are control variables, and also which ones are just lags of each type of variable. Alternatively, in more complicated cases, the user can specify all those details in an explicit way using the available options for such cases.

The simplest local projection specification in which the response variable is y and the shock variable is n would be (automatic specification):

```
. use "usmacro2.dta"
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. locproj y n
Impulse Response Function
```

	IRF	Std.Err.	IRF LOW	IRF UP
0	0.74401	0.04677	0.65187	0.83615
1	0.36875	0.06193	0.24676	0.49074
2	0.11844	0.06591	-0.01140	0.24827
3	-0.03372	0.06640	-0.16452	0.09707
4	-0.04416	0.06651	-0.17517	0.08686
5	-0.07651	0.06640	-0.20732	0.05430

Which generates the graph in Figure 1:

Which would also be equivalent to the following (explicit specification):

```
.  locproj y, shock(n)
```

We can also add control variables by adding them after the shock one or using the option controls. In this case our control variable is r

```
.  locproj y n r
```

Which would be equivalent to the following (explicit specification):

```
.  locproj y, shock(n) controls(r)
```

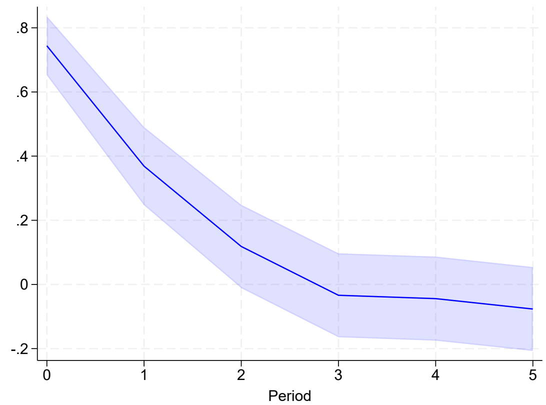

Figure 1: Impulse Response Function

5.1.2 Specification of lags of dependent variable, shock and control variables

We might want to include lags of the dependent variable, the shock, or the control variables. For the dependent variable and the shock, there are explicit options, whereas for the control variables you need to add them either on the varlist or as additional variables in the `controls()` option.

In the following examples, we define that the dependent variable has one lag, that the shock variable has 2 lags, and that the control variable has 3 lags. They are all exactly equivalent:

Automatic specification

```
locproj y l.y n l.n l2.n r l.r l2.r l3.r

locproj y l.y l(0/2).n l(0/3).r

locproj l(0/1).y l(0/2).n l(0/3).r
```

Explicit or intermediate specification

```
locproj y n, controls(l.y l2.y l.n l2.n r l.r l2.r l3.r)

locproj y, shock(n) controls(l.y l2.y l.n l2.n r l.r l2.r l3.r)

locproj y, s(n) controls(l(1/2).y l(1/2).n l(0/3).r)
```

Using the ylags and slags options

```
locproj y n, ylags(2) slags(2) controls(l(0/3).r)

locproj y, s(n) ylags(2) slags(2) controls(l(0/3).r)
```

5.1.3. Specification of horizon length (steps)

We can define the horizon either as an interval or just by specifying the final step/horizon:

```
locproj y n, hor(0/12) yl(2) sl(2) c(l(0/2).r)
```

```
locproj y n, hor(12) yl(2) sl(2) c(l(0/2).r)
```

The initial step can be different from zero, either negative or positive, but it always has to be an integer number. In cases in which the horizon is different from zero, `locproj` adjusts the output accordingly, and in the cases of negative horizons, it also adjusts the way in which the lags of the dependent variable are included.

```
locproj y n, h(1/12) yl(2) sl(2) c(l(0/2).r)
```

We can see the output of the example with a negative starting horizon:

```
. locproj y n, h(-3/9) yl(2) sl(2) c(l(0/2).r)
Impulse Response Function
```

	IRF	Std.Err.	IRF LOW	IRF UP
-3	0.08586	0.07932	-0.07042	0.24214
-2	0.11698	0.06447	-0.01004	0.24401
-1	0.12394	0.06453	-0.00321	0.25108
0	0.80732	0.06441	0.68043	0.93421
1	0.46783	0.08251	0.30528	0.63039
2	0.33490	0.08883	0.15987	0.50993
3	0.02058	0.09435	-0.16531	0.20647
4	0.05464	0.09378	-0.13015	0.23942
5	0.01809	0.09343	-0.16600	0.20218
6	0.02326	0.09510	-0.16414	0.21066
7	0.04984	0.09335	-0.13412	0.23380
8	-0.07028	0.09180	-0.25117	0.11061
9	-0.15789	0.09437	-0.34386	0.02808

Which generates the graph in Figure 2:

5.1.4. Estimation method options

We want to use the Newey-West as the estimation method in order to correct for autocorrelation, which consequently requires specifying that the option ”lag” in the Newey-West command should depend on the horizon of the IRF in the following way:

```
locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag)
```

We can add any other existing methodological option corresponding to the method introduced in the option met() simply by writing it down after the comma, as long as that option has a different name to any of the existing locproj options. For instance, we can include the option of no-constant term by adding the option noconstant:

```
locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) noconstant
```

5.1.5. Displaying all the regression outputs

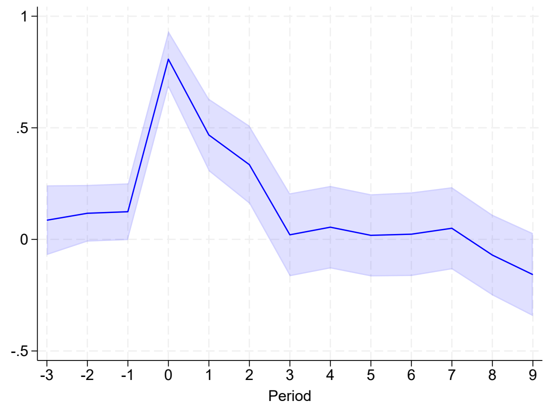

Figure 2: IRF with negative starting horizon

If we want to take a look at the regression output for each of the horizons of the IRF, we can use the options `noisily` and `stats`. The regression outputs displayed when using the option `noisily` are not the exact outputs from whatever estimation method we are using, but a simplified output table. The reason for this is that `locproj` uses temporary variables whose given names do not have any meaning and would be difficult to understand. `locproj` generates a new output table with variable names related to the variable list defined by the user.

```
. locproj y l.y l(0/1).n l(0/1).r, h(-1/1) noisily
y_h(-1)
------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           n |
         --. |   .1372814   .0635615     2.16   0.031     .0127032    .2618595
         L1. |    .702149   .0658452    10.66   0.000     .5730949    .8312032
             |
           y |
         L2. |  -.0749726   .0525482    -1.43   0.154    -.1779652      .02802
             |
           r |
         --. |   -.054914   .1438184    -0.38   0.703     -.336793     .226965
         L1. |  -.0084252   .1440649    -0.06   0.953    -.2907871    .2739367
             |
       _cons |   2.508633   .3054069     8.21   0.000     1.910047     3.10722
------------------------------------------------------------------------------
y_h(0)
------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           n |
         --. |   .8242872   .0639428    12.89   0.000     .6989616    .9496129
         L1. |  -.1749075   .0742624    -2.36   0.019    -.3204592   -.0293558
             |
           y |
```

```
         L1.  |   .0288524   .0644984     0.45   0.655    -.0975622    .1552669
              |
           r  |
         --.  |   .0347293    .143932     0.24   0.809    -.2473722    .3168308
         L1.  |  -.0887312   .1441751    -0.62   0.538    -.3713092    .1938468
              |
       _cons  |   2.368041   .3182235     7.44   0.000     1.744335    2.991748
--------------+----------------------------------------------------------------
y_h(1)
--------------+----------------------------------------------------------------
              |  Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
--------------+----------------------------------------------------------------
           n  |
         --.  |   .4231084    .084143     5.03   0.000     .2581912    .5880256
         L1.  |  -.2823765   .0975591    -2.89   0.004    -.4735888   -.0911642
              |
           y  |
         L1.  |   .1698436   .0846935     2.01   0.045     .0038474    .3358399
              |
           r  |
         --.  |   .1786659   .1890213     0.95   0.345     -.191809    .5491408
         L1.  |  -.2975255    .189286    -1.57   0.116    -.6685193    .0734683
              |
       _cons  |   2.896414    .419691     6.90   0.000     2.073834    3.718993
-------------------------------------------------------------------------------

Impulse Response Function
              |      IRF    Std.Err.    IRF LOW     IRF UP
--------------+-------------------------------------------
           -1 |   0.13728   0.06356    0.01206    0.26250
            0 |   0.82429   0.06394    0.69832    0.95026
            1 |   0.42311   0.08414    0.25734    0.58888
```

The stats option generates a table with each regression statistics for every horizon, i.e. number of observations, R-squared or pseudo-R-squared, F-statistic or Chi2-statistic and their respective p-values.

```
. locproj y l.y l(0/1).n l(0/1).r, h(-1/3) stats
Statistics by step
         |          N          R2        psR2           F        Chi2        Prob
---------+-----------------------------------------------------------------------
      -1 |        242       0.528           .       52.81           .       0.000
       0 |        243       0.526           .       52.58           .       0.000
       1 |        242       0.183           .       10.57           .       0.000
       2 |        241       0.116           .        6.18           .       0.000
       3 |        240       0.026           .        1.24           .       0.286
Impulse Response Function
              |      IRF    Std.Err.    IRF LOW     IRF UP
--------------+-------------------------------------------
           -1 |   0.13728   0.06356    0.01206    0.26250
            0 |   0.82429   0.06394    0.69832    0.95026
            1 |   0.42311   0.08414    0.25734    0.58888
            2 |   0.30676   0.08771    0.13395    0.47957
            3 |   0.01101   0.09229   -0.17082    0.19285
```

5.1.6. Use of the transformation options

In this example we are going to estimate the IRF from a shock to the variable r (FED funds rate) into the variable i (corporate bond interest rate (AAA)) using the different transformation options. In order to express the result in percentage terms, we also make use of the option `fact()` and we scale the response by a factor of 100.

We first need to generate the variables in logarithm and their differences to compare the results:

```
gen lni = ln(i)

gen lnr = ln(r)

gen dlni = d.lni

gen dlnr = d.lnr
```

We first estimate the IRF using the two variables in logarithm:

```
locproj lni lnr, f(100) yl(2) sl(2)
```

But we can also estimate the same by using the option transf(logs) for the dependent variable:

```
locproj i lnr, f(100) yl(2) sl(2) tr(logs)
```

We can estimate the model in differences by entering the log-difference as the dependent variable:

```
locproj d.lni lnr, f(100) yl(2) sl(2)
```

However, we can also use the option transf(diff) with the dependent variable in logs:

```
locproj lni lnr, f(100) yl(2) sl(2) tr(diff)
```

Or we can use the option transf(logs diff) with the dependent variable in levels:

```
locproj i lnr, f(100) yl(2) sl(2) tr(logs diff)
```

For estimating the model in cumulative differences we can do it with both variables in logarithm:

```
locproj lni lnr, f(100) yl(2) sl(2) tr(cmlt)
```

Which would be equivalent to estimate the model with the variable i in levels and using the option tr(logs cmlt):

```
locproj i lnr, f(100) yl(2) sl(2) tr(logs cmlt)
```

5.1.7. Saving the IRF results into new variables

If we want to save the estimated IRF into a new variable that can be used later, we can use it through the options `saveirf` and `irfname()`. If we just type `saveirf`, `locproj` generates four (or six) new variables with the IRF, its standard error and the confidence bands. `locproj` uses some predetermined default names to save the corresponding variables (`_irf`, `_seirf`, `_irf_lo` and `_irf_up`)), but if we want to give them a name of our preference (e.g. newirf), we can do it through the option `irfname()`:

```
. locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) saveirf
```

```
. locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) save
irfname(newirf)
```

5.1.8. Some graph options

If we do not want locproj to produce a graph, we just have to type nograph:

```
. locproj y l(0/4).n l(0/4).r, h(12) m(newey) hopt(lag) yl(3) nograph
```

In the following example we are going to produce a graph in which a dashed-line with the value of zero is included, we are going to give the graph the tittle "LP Example", include a label "Hours worked", change the color of the IRF line and its confidence interval to red instead of blue, and define the time axis as "Number of Days":

```
. locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) zero
title("LP Example") label("Hours worked") lcolor(red)
ttitle("Number of quarters") conf(66 95)
```

Which generates the graph in Figure 3:

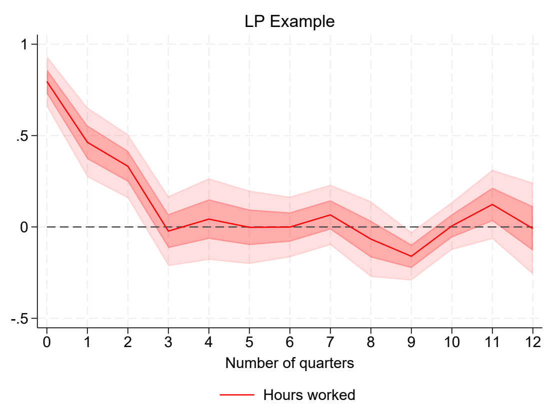

Figure 3: IRF graph options

Next, we are going to give the graph a name, we are going to save it in a folder in our disk as a png file named "example1":

```
. locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) zero
title(LP Example) grname(Example1) grsave(C:\Documents\example1.png) as(png)
```

We can also add other graph options inside the `gropt()` option, for instance, and we can define the labels of the y-axis and change the background color to white:

```
. locproj y l.y l(0/2).n l(0/3).r, h(12) met(newey) hopt(lag) zero
```

```
title(LP Example) gropt(graphregion(fcolor(white)) ylabel(-0.25(0.25)1))
```

5.2 Interaction of a dummy variable with the shock, (State dependent IRF)

We want to specify a different reaction to our shock variable before and after the global financial crisis (GFC). We first need to generate two dummy variables. The first dummy variable `bef_gfc` is equal to one before the first quarter of 2009 and zero afterwards, meanwhile the second dummy variable `aft_gfc` is equal to one after the first quarter of 2009 and zero before that.

```
. gen bef_gfc = dateq<tq(2009q1)
. gen aft_gfc = dateq>=tq(2009q1)
```

We can also generate two interaction variables, i.e., the product of the dummy variables times the shock variable n. The first interaction variable is equal to n before the GFC and equal to zero afterwards. The second interaction variable is equal to zero before the GFC and equal to n afterwards.

```
. gen n_bef = n*bef_gfc
. gen n_aft = n*aft_gfc
```

The estimated IRF after the GFC corresponds to the addition of the individual coefficients of the variables n and `n_aft`. Thus we have to specify that the shock corresponds to both variables, which is done by including both of them inside the option `shock()`. `locproj` will take all the variables that are included in the `shock()` option and add their individual effects:

```
. locproj y l.y l(0/3).r, s(n n_aft) sl(2) hor(12)
```

We are going to display the regression output of the first two steps, $h = 0, 1$, without any other control variable so that it is easier to see how the coefficients of the two variables are added:

```
. locproj y, s(n n_aft) hor(1) noi
y_h(0)
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
n	.7901824	.0484706	16.30	0.000	.6951817	.8851831
n_aft	-.4268065	.1410519	-3.03	0.002	-.7032631	-.1503498
_cons	2.063529	.1665531	12.39	0.000	1.737091	2.389967

```
y_h(1)
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
n	.4041814	.0649363	6.22	0.000	.2769086	.5314541
n_aft	-.3338654	.1917116	-1.74	0.082	-.7096133	.0418824
_cons	2.508294	.2234139	11.23	0.000	2.070411	2.946178

```
Impulse Response Function
             |      IRF   Std.Err.    IRF LOW     IRF UP
-------------+-------------------------------------------
           0 |  0.36338    0.13394    0.09953    0.62722
           1 |  0.07032    0.18212   -0.28845    0.42908
```

We can also directly include the interaction between the dummy variable **aft_gfc** and the variable **n** inside the option `shock()`, but crucially, we have to specify in the interaction that both variables are continuous even though the first one is a dummy, i.e. `c.aft_gfc#c.n`, and we would get the same results:

```
. locproj y, s(n c.aft_gfc#c.n) hor(1) noi
y_h(0)
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
n	.7901824	.0484706	16.30	0.000	.6951817	.8851831
c.aft_gfc#c.n	-.4268065	.1410519	-3.03	0.002	-.7032631	-.1503498
_cons	2.063529	.1665531	12.39	0.000	1.737091	2.389967

```
y_h(1)
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
n	.4041814	.0649363	6.22	0.000	.2769086	.5314541
c.aft_gfc#c.n	-.3338654	.1917116	-1.74	0.082	-.7096133	.0418824
_cons	2.508294	.2234139	11.23	0.000	2.070411	2.946178

```
Impulse Response Function
             |      IRF   Std.Err.    IRF LOW     IRF UP
-------------+-------------------------------------------
           0 |  0.36338    0.13394    0.09953    0.62722
           1 |  0.07032    0.18212   -0.28845    0.42908
```

The interpretation of each individual coefficient is the following: the coefficient of the variable n corresponds to the impact before the GFC, and the coefficient of the variable `n_aft` (or `c.aft_gfc#c.n`) corresponds to the difference between the two periods.

Alternatively, we can use the interaction variable **n_bef** to estimate the IRF resulting from the shock variable n before the GFC in the following way:

```
. locproj y l.y l(0/3).r, s(n_bef n) sl(2) hor(12)
```

Which is equivalent to (using an interaction of two continuous variables):

```
. locproj y l.y l(0/3).r, s(c.bef_gfc#c.n n) sl(2) hor(12)
```

In this case, the IRF before the GFC corresponds to the addition of the individual coefficients of the variables **n** and **n_bef**, meanwhile the coefficient of the variable **n** corresponds to the impact after the GFC, and the coefficient of the variable `n_bef` corresponds to the difference between the two periods.

5.3 Nonlinear effects and interactions: Using the option lcs()

We will replicate Example 5.2 but using the option `lcs()`. The use of the option `lcs()` is equivalent to the use of the command `lincom` after estimating any regression command. The expression that goes inside the parenthesis is analogous to any expression that is tested using the commands `lincom` or `test`.

We are going to use the dummy variable `aft_gfc` that is equal to one after the first quarter of 2009. But now we are going to use the factor variables syntax to specify the shock, and thus, we will need to use the option `lcs()`, since the syntax of factor variables could be more complicated, although sometimes more convenient to use.

The option `lcs()` allows us to specify the shock in any way we want, as long as it is expressed as a linear combination of the variables included in the model specification.

In this case, the model specification includes both the variable `n` and the interaction of the GFC dummy `aft_gfc` and the variable `n`. The interpretation of the coefficients of each variable is the same as in Example 5.2, the coefficient of the variable `n` corresponds to the response during the period preceding the GFC, meanhwile the coefficient of the variable `1.aft_gfc#c.n` corresponds to the difference in the response between the two periods. Thus the total response after the GFC is equal to the sum of the two coefficients. (In this specification the variable `0.aft_gfc#c.n` is omitted).

```
. locproj y l(0/2)(n aft_gfc#c.n) l(0/3).r, ylags(1)
lcs(n+1.aft_gfc#c.n) noi
```

However, if we only include the interaction term between the dummy and the continuous variable, i.e., `aft_gfc#c.n`, then the response after the GFC is equal to the coefficient of the variable `1.aft_gfc#c.n`, meanwhile the response before the GFC is equal to the coefficient of the variable `0.aft_gfc#c.n`

IRF before the GFC:

```
. locproj y l(0/2)(aft_gfc#c.n) l(0/3).r, ylags(1) lcs(0.aft_gfc#c.n) noi
```

IRF after the GFC:

```
. locproj y l(0/2)(aft_gfc#c.n) l(0/3).r, ylags(1) lcs(1.aft_gfc#c.n) noi
```

5.4 More complicated nonlinear interactions: using lcs() or margins options

If we want to estimate the response to a nonlinear shock, we need to take into account that the estimated response might depend on the size of the shock, the level of the interaction variable, and the shock variable initial level. Therefore, the estimation depends on the coefficients and levels of more than one variable.

5.4.1. Quadratic terms

We are going to show how to estimate the case of the response when the shock includes a quadratic term, as defined in Equation (5). For instance, a quadratic term of the variable n.

In Stata we can generate a new variable equal to the square of n:

```
gen n_2 = n^2
```

Alternatively, we can use an interaction term such as `c.n#c.n`. For simplicity of the syntax, in this example we use the new generated variable `n_2`, but the result will be exactly the same if we use `c.n#c.n` instead.

Since the shock is composed of two variables we are going to include both `n` and `n_2` into the option `shock()` and we are going to write down the expression in Equation (5) into the option `lcs()`, making β_h^a equal to the estimated coefficient of `n`, β_h^b equal to the estimated coefficient of `n_2`. We first assume that the shock is equal to one, i.e. $\delta = 1$ and $\delta^2 = 1$, and we are going to use as the initial level of the variable `n`, i.e. $s0$, equal its estimated sample mean:

```
. sum n

. scalar nm=r(mean)

. locproj y, shock(n n_2) ylags(1) slags(2) controls(l(0/3).r) hor(12)
lcs(n*1+2*n_2*1*nm+n_2*1)
```

If we want to estimate the IRF for other values of the variable n, for instance for $n = 3$ and $n = 5$, and thus, $n^2 = 9$ and $n^2 = 25$ we would need to change the expression that goes into the option `lcs()`:

```
. locproj y, shock(n n_2) ylags(1) slags(2) controls(l(0/3).r) hor(12)
lcs(n*3+2*n_2*3*nm+n_2*9)

. locproj y, shock(n n_2) ylags(1) slags(2) controls(l(0/3).r) hor(12)
lcs(n*5+2*n_2*5*nm+n_2*25)
```

We can also use the option `margins`, although it has some limitations. We cannot use the option `shock()` with the two variables since margins can obtain the derivative of only one variable. Since we are including lags of the shock, we need to change the way in which we include the lags, making sure that specification includes the same lags of both n and its quadratic term. We are also going to use the interaction `c.n#c.n` instead of the variable `n_2`. We also need to use the option `mropt()` to specify that the shock variable is evaluated at the desired initial level ($s0$), introducing the option `atmeans`.

```
. locproj y l(0/2)(n c.n#c.n), ylags(1) controls(l(0/3).r) h(12)
margins mropt(atmeans)
```

5.4.2.Interaction with a continuous variable

In this example we will estimate an IRF when the shock interacts with another continuous variable as in (9). The variable that interacts is e, the percent change in US exchange rate. As in other examples we are only interested in this interaction as a way to show how to do it with `locproj` and not because its economic meaning or relevance.

This case is pretty similar to the one of a quadratic term, however in this case the initial level of the shock variable $s0$ does not intervene. However, the idea of the interaction is to evaluate the response at different levels of the variable e.

Initially, if in Equation (9) we assume that the size of the shock $\delta = 1$ and we want to estimate the response at a level of the variable $e = 1$, i.e. $\theta = 1$, we just need to include our shock variable n and the interaction term `c.n#c.e` in the option `shock()`:

```
. locproj y, s(n c.n#c.e) ylags(1) sl(3) controls(l(0/3).r) hor(12)
```

However, if we want to estimate the response at other levels of the variable n, for instance, at its sample mean, we need to use the option `lcs()`:

```
. sum e
```

```
. scalar em=r(mean)
```

```
. locproj y l(0/3)(n c.n#c.e), ylags(1) controls(l(0/3).r) hor(12)
lcs( n + c.n#c.e*em)
```

We would obtain exactly the same result if we use the option `margins` in the following way (3.1645 is the mean of e):

```
. locproj y l(0/3)(n c.n#c.e), ylags(1) controls(l(0/3).r) hor(12) margins
mropt(atmeans at(e=3.164551))
```

We can evaluate the IRF at different levels of the variable e (at different values of θ). Evaluating the IRF at $\theta = 6$ using the option `lcs()` would be given by:

```
. locproj y l(0/3)(n c.n#c.e), ylags(1) controls(l(0/3).r) hor(12)
lcs(n + c.n#c.e*6)
```

We would obtain exactly the same results using the option `margins` in the following way:

```
. locproj y l(0/3)(n c.n#c.e), ylags(1) controls(l(0/3).r) hor(12)
margins mropt(atmeans at(e=6))
```

5.5 Example using Quantile Regression and LPGRAPH command

In this example we are going to estimate the IRF of the GDP growth rate to a shock in the monetary policy interest rate r. We want to estimate the IFR for different quantiles of the distribution of our dependent variable, using the quantile regression method `qreg`.

We want our shock variable to have an impact with a one period lag. However, the `qreg` command does not allow the use of time-series operators. Thus we first need to generate the variable r_{t-1} and using it as our shock variable:

```
. gen lr=l.r
```

Nevertheless, `locproj` has been adapted so that we can include lags of the dependent variable and the shock variable automatically even if the estimating method we are using does not allow time-series operators. We can do it by using the options `ylags()` and `slags()` respectively. For instance, in this example we want to include three lags of y and of r_{t-1}. Normally, if we are using `qreg` we would need to generate all these lagged variables, but with `locproj` we can just write `yl(3)` and `sl(3)`.

However, if we want to include lags of our control variables, we do need to do it by hand, generating each one of the lagged-terms we want. In our example, we are going to introduce three lags of the variable n:

```
. gen ln=l.n
. gen l2n=l2.n
. gen l3n=l3.n
```

We are going to estimate the IRF for three moments of the variable y distribution: its mean, the 20th percentile, the median, and the 80th percentile. In all cases we use a robust estimator of the variance-covariance matrix. For the average outcome, we use OLS:

```
. locproj y lr n ln l2n l3n, yl(3) sl(3) h(-4/12) save irfn(Mean) r nograph
```

For the other moments of the distribution we use the command `qreg`:

```
. locproj y lr n ln l2n l3n, yl(3) sl(3) h(-4/12) m(qreg) q(20) nograph
save irfn(Q20) vce(r)
. locproj y lr n ln l2n l3n, yl(3) sl(3) h(-4/12) m(qreg) q(50) nograph
save irfn(Q50) vce(r)
. locproj y lr n ln l2n l3n, yl(3) sl(3) h(-4/12) m(qreg) q(80) nograph
save irfn(Q80) vce(r)
```

In all the cases we used the option "nograph" since we want to compare the four IRFs plotting them together in one graph using the command `lpgraph`. Therefore, we have also used the options `save` and `ifrname` to save the results of each LP into some variables that we can use.

Now we can create one graph with the four IRFs plotted together, while also choosing the color of each one of the IRFs as in Figure 4:

```
. lpgraph Mean Q20 Q50 Q80, h(-4/12) tti(Quarters) lab1(OLS) lab2(Low - Q20)
lab3(Median) lab4(High - Q80) lc1(red) lc2(green) lc3(blue) lc4(brown)
title(Example of qreg & lpgraph, size(0.9)) z
```

We can also create four separate graphs and then combine them into a single graph. To do so, we need to specify the option separate. In this case, we are giving each separate graph a title, and therefore, we also specify the option nogelend. Additionally, we are choosing the color red for the IRFs lines of the four graphs, as in Figure 5:

```
. lpgraph Mean Q20 Q50 Q80, h(-4/12) separate nolegend tti(Quarters)
ti1(OLS) ti2(Low - Q20) ti3(Median) ti4(High - Q80) lcolor(red)
```

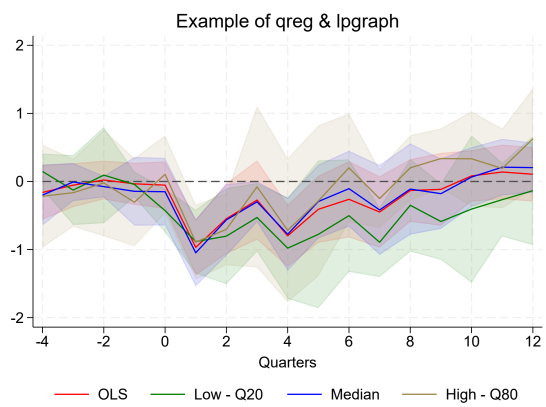

Figure 4: IRFs in the same graph

```
title(Example of qreg & lpgraph, size(0.9)) z
```

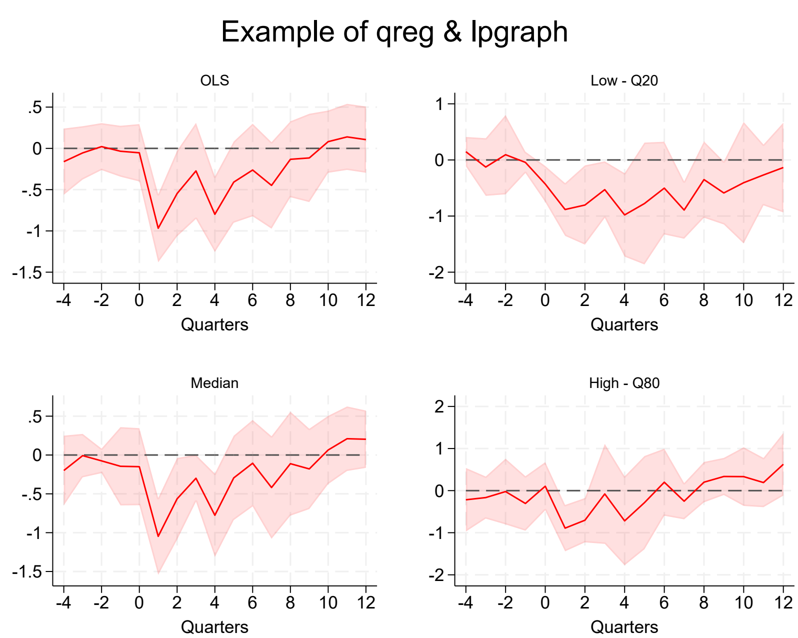

Figure 5: IRFs in separated graphs

5.6 Instrumental Variables

We follow the example of the command `ivlpirf` in the Stata 19.0 manual and we use data on US industrial production growth (ip growth), inflation rate (inflation), and the interest rate (fedfunds) to estimate the effects of an interest rate increase on economic activity and prices.

Following the example, we are concerned that the change in fedfunds is endogenous. We have available an instrument, money_inst, that captures monetary shocks. It is correlated with change in fedfunds but uncorrelated with any nonmonetary shocks. We use this variable as an instrument for the change in fedfunds.

In the following equation `locproj` interprets that the shock variable is *ip_growth* and we need to specify the instruments we want to use using the option `instr`. Initially, we will only use the instrument *money_inst*. We also need to specify which method we want to use through the option `met`, which in this case is `ivregress gmm`. Notice that in this case the `met` option should include the `gmm` sub-method.

```
. use usmacro3.dta

. locproj ip_growth d.fedfunds, yl(2) sl(2) m(ivregress gmm)
instr(money_inst)
```

If we have more available instruments we should include them in the `instr` option. For example, we can also include the instrument variable oil_instr. If our estimation method is `ivregress` we can test whether we have overidentification at every step of the LP. This can be done by using the option `ivtest`, which performs and displays one of the three postestimation tests available after using the command `ivregress`, for each step/horizon. For testing overindentifying restrictions we need to use the suboption `overid` inside the option `ivtest`:

```
.
. locproj ip_growth d.fedfunds, ivtest(overid) instr(money_inst oil_inst) ///
> z h(0/2) yl(2) sl(2) m(ivregress gmm)
IV Test Step = 0
  Test of overidentifying restriction:
  Hansen´s J chi2(1) = 1.33844 (p = 0.2473)

IV Test Step = 1
  Test of overidentifying restriction:
  Hansen´s J chi2(1) = 5.57007 (p = 0.0183)

IV Test Step = 2
  Test of overidentifying restriction:
  Hansen´s J chi2(1) = 2.06762 (p = 0.1505)

Impulse Response Function
```

	IRF	Std.Err.	IRF LOW	IRF UP
0	0.37225	0.23671	-0.09169	0.83620
1	0.32364	0.30674	-0.27755	0.92483

```
           2 |   0.15792    0.21822   -0.26978    0.58562
.
```

We can also use the suboption `endogenous` in `ivtest` to perform tests to determine whether endogenous regressors in the model are in fact exogenous at every step:

```
.
. locproj ip_growth d.fedfunds, instr(money_inst) ivtest(endogenous) ///
> h(0/2) yl(2) sl(2) z  m(ivregress gmm)
IV Test Step = 0
  Test of endogeneity (orthogonality conditions)
  H0: Variables are exogenous
  GMM C statistic chi2(1) =  .342906  (p = 0.5582)

IV Test Step = 1
  Test of endogeneity (orthogonality conditions)
  H0: Variables are exogenous
  GMM C statistic chi2(1) =  .763462  (p = 0.3822)

IV Test Step = 2
  Test of endogeneity (orthogonality conditions)
  H0: Variables are exogenous
  GMM C statistic chi2(1) =  .288881  (p = 0.5909)

Impulse Response Function
             |       IRF   Std.Err.    IRF LOW     IRF UP
-------------+--------------------------------------------
           0 |   0.39021    0.23816   -0.07658    0.85700
           1 |   0.36100    0.30897   -0.24456    0.96657
           2 |   0.16297    0.21899   -0.26624    0.59219
.
```

We can also use the command `ivqregress` if we want to use instrumental-variables quantile regression as our estimation method. In the same way as in the case of the command `qreg`, we can specify lags of the dependent variable and the shock as options, but we cannot use time-series operators in the main syntax. Then we would first need to generate a new variable equal to the change in the fed funds rate, **dfedfunds**. We also need to specify the submethod in the option `met` together with the command `ivqregress`:

```
. gen dfedfunds = d.fedfunds

. locproj ip_growth dfedfunds, yl(2) sl(2) m(ivqregress iqr)
instr(money_inst)
```

5.7 Binary dependent variable: using the margins options

We are going to use the JST dataset and the `"RecessionDummies"` dataset that contains data on recessions and financial crises:

```
. use "http://data.macrohistory.net/JST/JSTdatasetR5.dta"
```

```
. merge 1:1 year iso using "RecessionDummies.dta", nogen
```

```
. xtset ifs year
```

We also need to drop WWI and WWII years from JST dataset:

```
. drop if year >=1914 & year <=1919
```

```
. drop if year >=1939 & year <=1947
```

In our first example, we will estimate the IRF of the probability of a banking crisis to an increase in the US short-term interest rate. Our dependent variable in this case is the dummy variable crisisJST that is equal to 1 for banking crises.

We need to generate a new variable `stir_us` with the US interest rate as a common variable for all the countries in the sample, in order to estimate the response of the probability of a banking crisis to the short-term interest rate in the US:

```
. gen stir_us0=stir if iso=="USA"
```

```
. egen stir_us=mean(stir_us0), by(year)
```

Now we are going to estimate the IRF using the option margins. The option margins estimates the marginal effect of a unit of our shock variable (`stir_us`) on the probability of a banking crisis, which is our dependent (outcome) variable. We are using as estimation method the command `xtlogit` with fixed effects:

```
. locproj crisisJST l(0/2).stir_us, margins m(xtlogit) fe
```

We can also interact the shock variable with a dummy variable, for instance, whether a country has a "PEG" foreign exchange regime.

The option margins allow us to estimate a separate IRF for each category of the dummy variable PEG. To do that we need to use the option `mrfvar()`. In this option we need to specify the expansion of the categorical variable that has been interacted with our shock variable. We also need to use the explicit option to define which variable is our shock without any interaction term, since the command margins does not accept an interaction term expression in its `dydx()` option:

```
. locproj crisisJST peg#c.l(0/2).stir_us, s(stir_us) margins m(xtlogit)
fe mrfvar(1.peg)
```

```
. locproj crisisJST peg#c.l(0/2).stir_us, s(stir_us) margins m(xtlogit)
fe mrfvar(0.peg)
```

Alternatively, instead of entering the shock variable as peg#c.l(0/2).stir_us in the main syntax, we can enter the expression l(0/2).stir_us peg#c.l(0/2).stir_us and in this way the command would use the variable stir_us as the shock variable, without the need to specify it through the option `shock()`:

```
. locproj crisisJST l(0/2).stir_us peg#c.l(0/2).stir_us, margins m(xtlogit)
fe mrfvar(1.peg)
```

```
. locproj crisisJST l(0/2).stir_us peg#c.l(0/2).stir_us, margins m(xtlogit)
```

```
fe mrfvar(0.peg)
```

5.8 D-i-D Event Study

As described in Section 2.9 we can use `locproj` to estimate an event study based on the DiD estimator in the case where the treatment period is the same for all treated individuals.

In this example, we have a dataset that contains a response variable y and a variable d that is equal to one for a subset of individuals who were all treated in the year 2004. The dataset also contains a set of dummy variables $f01$, $f02$,..., $f06$ that are equal to one for the years 2001 to 2006 respectively.

Following Wooldridge (2021) we can estimate the Event Study regression without covariates using the specification shown in Equation 13:

```
.
. use did_common_6, clear

. xtset id year

Panel variable: id (strongly balanced)
 Time variable: year, 2001 to 2006
         Delta: 1 unit
.
. reg y c.d#c.f01 c.d#c.f02 c.d#c.f04 c.d#c.f05 c.d#c.f06 d ///
>      i.year, vce(cluster id)

Linear regression                                  Number of obs     =      3,000
                                                   F(11, 499)        =     128.77
                                                   Prob > F          =     0.0000
                                                   R-squared         =     0.2022
                                                   Root MSE          =     2.8893

                                      (Std. err. adjusted for 500 clusters in id)
------------------------------------------------------------------------------
             |               Robust
           y | Coefficient  std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
   c.d#c.f01 |  -.5507778   .3096218    -1.78   0.076    -1.159101    .0575453
             |
   c.d#c.f02 |  -.3420594   .3340021    -1.02   0.306     -.998283    .3141643
             |
   c.d#c.f04 |   3.176513   .3660582     8.68   0.000     2.457307    3.895718
             |
   c.d#c.f05 |   4.894176   .3581913    13.66   0.000     4.190427    5.597925
             |
   c.d#c.f06 |    5.83863   .3515392    16.61   0.000     5.147951     6.52931
             |
           d |  -1.628416   .3403956    -4.78   0.000    -2.297201   -.9596306
             |
        year |
       2002  |  -.0423039    .146694    -0.29   0.773     -.330518    .2459102
       2003  |  -.2627626   .1342313    -1.96   0.051    -.5264907    .0009655
       2004  |   .6043256   .1397316     4.32   0.000     .3297907    .8788604
       2005  |   .3253793   .1430212     2.28   0.023     .0443814    .6063772
       2006  |   .5981061   .1549947     3.86   0.000     .2935834    .9026288
             |
```

```
       _cons |   20.33051   .1467731   138.52   0.000     20.04214    20.61888
------------------------------------------------------------------------------

.
```

All effects are measured relative to the period just before intervention, i.e., 2003. The event-study coefficients are thus **-0.551, -.342, 0, 3.177, 4.894, 5.839**.

We can reproduce these results with `locproj` using Equation 12. We need to use the option `transf(cmlt)` to obtain the long-term differencing. We need to define the horizon period as `hor(-3/2)` since the coefficient of the period $h = -1$ is normalized to zero and we want to evaluate two periods before the intervention $h = -3, -2$ and three periods after the intervention $h = 0, 1, 2$.

We also need to have a variable that is equal to one only for treated individuals and only after the intervention, i.e., from 2004 onward, which we are going to call *treat*. Following Equation 12 again, our shock variable is the change in this new variable *d.treat*:

```
. gen treat=year>=2004 & d
. locproj y d.treat i.year, tr(cmlt) vce(cluster id) h(-3/2) z gropt(xline(-1, lc(gray)))
Impulse Response Function
             |        IRF    Std.Err.     IRF LOW      IRF UP
-------------+-----------------------------------------------
          -3 |   -0.55078     0.30936    -1.15712     0.05556
          -2 |   -0.34206     0.33372    -0.99614     0.31203
          -1 |    0.00000     0.00000     0.00000     0.00000
           0 |    3.17651     0.36575     2.45965     3.89337
           1 |    4.89418     0.35789     4.19272     5.59563
           2 |    5.83863     0.35125     5.15020     6.52706
```

We also included the necessary options to display lines at $period = 0$ and at $y = 0$ as can be seen in Figure 6:

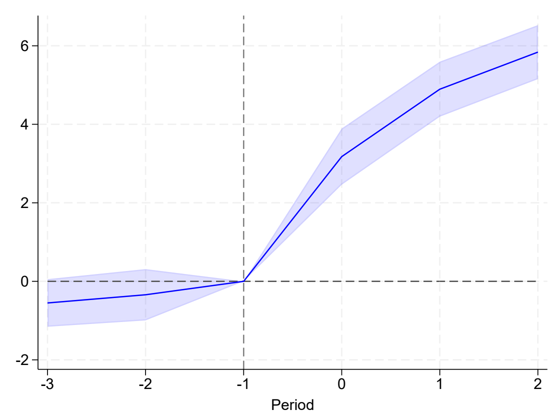

Figure 6: Event Study graph

6 Acknowledgments

I am grateful to Enrique Pinzon (StataCorp) and to the participants of the 2023 US Stata Conference at Stanford for their valuable comments and suggestions. I would also like to thank Jamel Saadaoui, my colleague David Sarasa, and the many users of `locproj` who have provided feedback and suggestions over the past few years.

Special thanks go to Christopher Baum for identifying some bugs in the earliest version of `locproj`, as well as for his ongoing efforts in maintaining the RePEc Repository.

The views expressed in this paper are those of the author and do not necessarily reflect those of BBVA Research.

7 References

Adammer, P. 2019. lpirfs: An R Package to Estimate Impulse Response Functions by Local Projections. *The R Journal* 11(2): 421–438.

Barattieri, A., and M. Cacciatore. 2023. Self-harming trade policy? Protectionism and production networks. *American Economic Journal: Macroeconomics* 15(2): 97–128.

Dube, A., D. Girardi, Ò. Jordà, and A. M. Taylor. 2025. A Local Projections Approach to Difference-in-Differences Event Studies. *Working Paper Series* (May).

Jordà, Ò. 2005. Estimation and inference of impulse responses by local projections. *American economic review* 95(1): 161–182.

Jordà, Ò., M. Kornejew, M. Schularick, and A. M. Taylor. 2022. Zombies at large? Corporate debt overhang and the macroeconomy. *The Review of Financial Studies* 35(10): 4561–4586.

Jordà, Ò., M. Schularick, and A. M. Taylor. 2015. Betting the house. *Journal of International Economics* 2015(96): 2–18.

Jordà, Ò., and A. M. Taylor. 2024. Local Projections. *Federal Reserve Bank of San Francisco Working Paper Series* 2024(24).

Makabe, Y., Y. Norimasa, et al.. 2022. The term structure of inflation at risk: A panel quantile regression approach. Technical report, Bank of Japan.

Wooldridge, J. M. 2021. Two-way fixed effects, the two-way mundlak regression, and difference-in-differences estimators. *Available at SSRN 3906345* .

About the authors

Alfonso Ugarte-Ruiz is a Principal Economist at the Global Macrofinancial Scenarios Unit of BBVA Research.

Working papers

2025

25/09 **Alfonso Ugarte-Ruiz:** Locproj & Lpgraph: Stata commands to estimate Local Projections

25/08 **Gergely Buda, Vasco M. Carvalho, Giancarlo Corsetti, João B. Duarte, Stephen Hansen, Afonso S. Moura, Álvaro Ortiz, Tomasa Rodrigo, José V. Rodríguez Mora, Guilherme Alves da Silva:** The Short Lags of Monetary Policy.

25/07 **Ángel de la Fuente, Pep Ruiz:** Series largas de VAB y empleo regional por sectores, 1955-2023. Actualización de RegData-Sect hasta 2023.

25/06 **David Sarasa-Flores:** Buy Guns or Buy Roses?: EU Defence Spending Fiscal Multipliers.

25/05 **Angel de la Fuente:** Las finanzas autonómicas en 2024 y entre 2003 y 2024.

25/04 **Joxe Mari Barrutiabengoa Ortubai, Rodrigo Enrique Falbo Piacentini, Agustín García Serrador, Juan F. Rubio-Ramírez:** Unraveling the impact of a carbon price shock on macroeconomic variables: a Narrative Sign Restrictions approach.

25/03 **Alvaro Ortiz, Tomasa Rodrigo, David Sarasa, Pedro Torinos, Sirenia Vazquez:** What can 240,000 new credit transactions tell us about the impact of NGEU funds?.

25/02 **Gergely Buda, Vasco M. Carvalho, Giancarlo Corsetti, João B. Duarte, Stephen Hansen, Afonso S. Moura, Álvaro Ortiz, Tomasa Rodrigo, José V. Rodríguez Mora, Guilherme Alves da Silva:** The Short Lags of Monetary Policy.

25/01 **Ángel de la Fuente:** Series largas de algunos agregados económicos y demográficos regionales: actualización de RegData hasta 2023.

2024

24/14 **Joxe M. Barrutiabengoa, Giancarlo Carta, Nara González, Pilar Más, Diego Pérez, Gül Yücel:** The Impact of Climate Change on Tourism Demand in Türkiye.

24/13 **Clodomiro Ferreira, José Miguel Leiva, Galo Nuño, Álvaro Ortiz, Tomasa Rodrigo and Sirenia Vazquez:** The heterogeneous impact of inflation on households' balance sheets.

24/12 **Ángel de la Fuente:** La evolución de la financiación de las comunidades autónomas de régimen común, 2002-2022.

24/11 **J.M. Barrutiabengoa, G. Carta, N. González, D. Pérez, P. Más and G. Yücel:** Climate change scenarios and the evolution of Spanish tourism.

24/10 **Federico D. Forte:** Pronóstico de inflación de corto plazo en Argentina con modelos *Random Forest*.

24/09 **Ángel de la Fuente:** La liquidación de 2022 del sistema de financiación de las comunidades autónomas de régimen común.

24/08 **Prachi Mishra, Alvaro Ortiz, Tomasa Rodrigo, Antonio Spilimbergo, and Sirenia Vazquez:** Ecommerce during Covid in Spain: One “Click” does not fit All.

24/07 **A. Castelló-Climent and R. Doménech:** Convergence in Human Capital and Income.

24/06 **J. Andrés, J.E. Boscá, R. Doménech and J. Ferri:** TheWelfare Effects of Degrowth as a Decarbonization Strategy.

24/05 **Ángel de la Fuente:** Las finanzas autonómicas en 2023 y entre 2003 y 2023.

24/04 **Ángel de la Fuente y Pep Ruiz:** Series largas de VAB y empleo regional por sectores, 1955-2022. Actualización de RegData-Sect hasta 2022.

24/03 **Ángel de la Fuente:** Series largas de algunos agregados económicos y demográficos regionales: Actualización de RegData hasta 2022.

24/02 **J. Andrés, E. Bandrés, R. Doménecha and M.D. Gadea:** SocialWelfare and Government Size.

24/01 **J. Andrés, J.E. Boscá, R. Doménech and J. Ferri:** Transitioning to net-zero: macroeconomic implications and welfare assessment.

CLICK HERE TO ACCESS THE WORKING DOCUMENTS PUBLISHED IN

Spanish and **English**

DISCLAIMER