# Package 'rfcipDemand'

August 28, 2025

**Type** Package

**Title** Estimate Federal Crop Insurance Program Demand Models

**Version** 0.0.0.9000

**Author** Francis Tsiboe [aut, cre] (<https://orcid.org/0000-0001-5984-1072>)

**Maintainer** Francis Tsiboe <ftsiboe@hotmail.com>

**Contributor** -

**Reviewer** -

**Creator** Francis Tsiboe

**Description** Tools to construct county–crop–practice–plan–unit panels from the
USDA RMA Summary of Business (SOBTPU) and related sources, and to estimate
FCIP demand systems with two-way cluster-robust covariance. The pipeline
standardizes coverage measures, merges price and instrument variables, adds
rental-rate and price-index controls, reconciles county acreage (FSA/NASS),
and produces diagnostics including robust first-stage F-tests. Methods align
with the empirical design in ``The crop insurance demand response to premium
subsidies Evidence from U.S. Agriculture'' (Food Policy, 2023, 119(3)).

**License** GPL-3 + file LICENSE

**URL** https://github.com/you/rfcipDemand

**BugReports** https://github.com/you/rfcipDemand/issues

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**VignetteBuilder** knitr

**Depends** R (>= 4.1.0)

**Imports** rfsa, data.table, systemfit,sandwich,doBy,car,plyr,usmap,stats,utils, tidyr, nlsur

**Remotes** github::UrbanInstitute/urbnmapr, github::dylan-turner25/rfsa, github::JanMarvin/nlsur

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**LazyData** true

**Cite-us** If you find it useful, please consider staring the repository and citing the following studies
- Tsiboe, F. and Turner, D. (2025). ``Incorporating buy-
up price loss coverage into the United States
farm safety net.'' Applied Economic Perspectives and Policy.
- Tsiboe, F., et al. (2025). ``Risk reduction impacts of crop insurance in the United States.''

Applied Economic Perspectives and Policy.
- Gaku, S. and Tsiboe, F. (2024). Evaluation of alternative farm safety net program combination strategies. Agricultural Finance Review.

# Contents

---

calculate_mode            *Calculate the Statistical Mode*

---

## Description

Returns the element that occurs most frequently in a vector.

## Usage

```
calculate_mode(x, na.rm = TRUE)
```

## Arguments

x                 A vector of any atomic type (numeric, character, factor,).

na.rm             Logical; should missing values be ignored? Defaults to TRUE. If FALSE and x contains any NAs, the function returns NA.

## Details

Internally the function:

1. Optionally removes NAs (`na.rm = TRUE`).
2. Builds a lookup table of unique values via `unique(x)`.
3. Counts the frequency of each unique value with `tabulate(match(x, ux))`.
4. Returns the value with the maximum count.

Because it relies on base R functions, the implementation is vectorised and generally fast for typical data-frame column sizes.

## Value

A single value giving the modal element of `x`. If two or more values are tied for the highest frequency, the first one encountered in `x` is returned.

---

estimate_fcip_instruments

*Estimate FCIP Instrumental Variables (Unloaded Rates)*

---

## Description

Uses historical FCIP rate data to build instrumented unloaded-rate variables following:

1. Tsiboe & Turner (2023), Econometric identification of crop insurance participation *Agricultural and Resource Economics Review*, 52(3):476-497. https://doi.org/10.1017/age.2023.13

## Usage

```
estimate_fcip_instruments(year, statplan)
```

## Arguments

year        Integer. The target crop year for which to construct instruments.

statplan    A data.table containing FCIP rate elements, including at least:

   **commodity_year**  Year of the rate observation.

   **state_code, county_code**  County identifiers.

   **commodity_code**  Crop identifier.

   **insured_area, lcr, contiguous_state_code, contiguous_county_code**  Fields required by `estimate_fcip_unloaded_rate()`.

## Details

1. **Task list**: Identify all unique (state, county) pairs with data in the 2-21 years before `year`.
2. **Unloaded-rate calculation**: For each county in `task_list`, call `estimate_fcip_unloaded_rate()` on the same 2-21 year window to get `tau`. Errors return `NULL` so processing continues.
3. **Contiguous-county smoothing**:
   - Build a lookup table of contiguous counties (using `contiguous_county`).

- For each contiguous group, compute the mean `tau` to get `tau_c`.

4. **Merge & fill**: Left-join the raw `adm` and `contiguous_adm`; replace any zero/NA/Inf `tau` with the group mean `tau_c` into `tau_sob`.

5. **Cleanup**: Drop helper columns (`tau`, `tau_c`), remove invalid rows, add `commodity_year`, and return the result.

### Value

A data.table with one row per county-crop for the specified `year`, containing:

**state_code, county_code, commodity_code** Keys.

**tau_sob** Smoothed unloaded rate (uses contiguous-county means to fill zeros/NAs).

**commodity_year** The input `year`, repeated.

### See Also

Other FCIP instruments: [get_yu2018_instrument](#)()

---

fcip_contiguous_county

*fcip_contiguous_county*

---

### Description

A combined dataset for fcip_contiguous_county

### Usage

```
data(fcip_contiguous_county)
```

### Format

A data frame with 24307 rows and 12 columns covering Inf–Inf.

### Source

USDA-RMA, Actuarial Data Master - A0123

---

```
fcip_demand_data_dispatcher
```
*Build dataset to estimate Federal Crop Insurance Program (FCIP) demand (modular pipeline)*

---

### Description

End-to-end pipeline that: (1) prepares SOBTPU and coverage aggregates, (2) adds prices/instruments/rental rates/price index, (3) reconciles county acreage (FSA + NASS), and (4) finalizes bins/labels/pooling for demand estimation.

### Usage

```
fcip_demand_data_dispatcher(
  study_years = 2001:(as.numeric(format(Sys.Date(), "%Y")) - 1),
  identifiers = c("commodity_year", FCIP_INSURANCE_POOL, "insurance_plan_code",
    "unit_structure_code")
)
```

### Arguments

study_years    Integer vector of commodity years to include. Defaults to `2001:(as.numeric(format(Sys.Date(),`
               `"%Y")) - 1)`.

identifiers    Character vector of grouping keys that define the aggregation grain. Must be
               columns present in SOBTPU (e.g., `"commodity_year"`, `FCIP_INSURANCE_POOL`,
               `"insurance_plan_code"`, `"unit_structure_code"`, and-if desired-additional
               keys like `"commodity_code"` or `"practice_code"`). Enrichment joins for re-
               codes are performed only when the required keys are included in `identifiers`

### Details

Aligned with Asche, Bekkerman, & Li (2023), *Food Policy*, 119(3):102505 ([doi:10.1016/j.foodpol.2023.102505](doi:10.1016/j.foodpol.2023.102505)). Requires internet access to download release `.rds` assets and several in-memory lookup tables (see stage docs).

### Value

A `data.table` ready for FCIP demand estimation.

---

```
fcip_demand_sys_coeff_table
```
*Tidy coefficient table with cluster-robust SEs (from supplied VCOV)*

---

### Description

Builds a clean coefficient table for a `systemfit` model using a **user-supplied covariance matrix** (e.g., two-way clustered from `fcip_demand_sys_vcov()`). Estimates come from `coef(fit)`, standard errors from `diag(vcMat)`, then Z-scores and two-sided normal p-values are computed. The demand column is inferred from the equation prefix in the coefficient names:

- `"Gamma_*"` to `"Gamma"`
- `"Theta_*"`, `"Theta1_*"`, `"Theta2_*"`, ... to `"Theta"` Otherwise the prefix itself is used.

## Usage

```
fcip_demand_sys_coeff_table(fit, vcMat, p_digits = 5)
```

## Arguments

| | |
|---|---|
| `fit` | A fitted `systemfit` object. |
| `vcMat` | A covariance matrix conformable with `coef(fit)`. Row/column names are used to align; if missing, positional alignment is assumed. |
| `p_digits` | Integer; number of digits to keep for p-values (default 5). |

## Value

A `data.frame` with columns: demand, coef, Estimate, StdError, Zvalue, Pvalue.

---

`fcip_demand_sys_effect`

*Delta-method "total protection response"*

---

## Description

Combines equation-specific effects into a single "Total" effect for each regressor in `c(fields$endogenous, fields$included)` using `car::deltaMethod` and a supplied covariance matrix.

## Usage

```
fcip_demand_sys_effect(fit, vcMat, fields, data)
```

## Arguments

| | |
|---|---|
| `fit` | A fitted `systemfit` object (the structural system). |
| `vcMat` | Covariance matrix conformable with `coef(fit)` (e.g., from `fcip_demand_sys_vcov()`). |
| `fields` | Named list carrying model fields; must include `outcome`, `endogenous`, and `included`. |
| `data` | Estimation data used to check variable availability and build delta-method expressions. |

## Value

A `data.frame` with rows demand=`"Total"` and columns: demand, coef, Estimate, StdError, Zvalue, Pvalue.

---

```
fcip_demand_sys_estimate
```
*System estimator (modular wrapper; preserves original outputs)*

---

### Description

Runs: per-level prep -> partial/tilda -> systemfit -> clustered VCOV -> delta-method totals -> (optional) restricted NLSUR -> diagnostics -> bind rows.

### Usage

```
fcip_demand_sys_estimate(model, data)
```

### Arguments

model      List with elements: `outcome, endogenous, included, excluded` (opt), `partial` (opt), `FE` (logical), `disag` (string colname), optional `restrict` (logical), `name` (string).

data       Input data.frame/data.table with all referenced columns plus `pool` and `commodity_year`.

### Value

A data.frame aggregating results across all disaggregation levels.

---

```
fcip_demand_sys_fit
```
*Build systemfit formulas and estimate the system*

---

### Description

Constructs the list of structural equations (g) and instrument sets (h), then runs `systemfit()` using OLS (when no excluded instruments) or 3SLS-GMM (when excluded instruments are present).

### Usage

```
fcip_demand_sys_fit(
  data,
  fields,
  tilda_included,
  tilda_endogenous,
  tilda_excluded
)
```

## Arguments

| | |
|---|---|
| `data` | Estimation `data.frame`/`data.table` containing the `tilda_*` and `instr_*` variables referenced by the formulas. |
| `fields` | Named list with at least `outcome`, `included`, endogenous, and optionally `excluded`. |
| `tilda_included` | Character vector of residualized included regressor names (e.g., `"tilda_x1"`). |
| `tilda_endogenous` | |
| | Character vector of residualized endogenous regressor names (e.g., `"tilda_z1"`). |
| `tilda_excluded` | Character vector of instrument names (e.g., `"instr_z1"`), or `NULL` when no excluded instruments are used. |

## Value

A list with elements:

| | |
|---|---|
| `fit` | Fitted `systemfit` object. |
| `g` | List of structural formulas. |
| `h` | List of instrument formulas. |

---

fcip_demand_sys_level_prep

*Prepare data for a single level*

---

## Description

Filters to one disaggregation `level`, drops incomplete/invalid rows, removes constant partials, and optionally demeans via a fixed-effects helper.

## Usage

```
fcip_demand_sys_level_prep(data, fields, level)
```

## Arguments

| | |
|---|---|
| `data` | Estimation dataset that already contains all columns referenced by `fields`. |
| `fields` | Named list: `outcome`, `endogenous`, `included`, `excluded` (opt), `partial` (opt), `FE` (logical), `disag` (column name). |
| `level` | One element of the disaggregation levels. |

## Value

A list: `data` (prepped), `NFE` (number of FE), `partial` (possibly reduced).

---

fcip_demand_sys_level_run

*Orchestrate one disaggregation level*

---

### Description

Runs the full pipeline for a single level: level prep -> partial/tilda creation -> systemfit -> two-way clustered VCOV -> delta-method totals -> optional restricted step -> diagnostics; then returns a tidy coefficient table with metadata.

### Usage

```
fcip_demand_sys_level_run(base_data, fields, level)
```

### Arguments

| | |
|---|---|
| base_data | Full estimation dataset (before subsetting to the level). |
| fields | Named list carrying model fields (see fcip_demand_estimation()), including disag, FE, outcome, endogenous, included, optional excluded, partial, restrict, and name. |
| level | One value of fields$disag to estimate for (e.g., a crop name). |

### Value

A data.frame with columns demand, coef, Estimate, StdError, Zvalue, Pvalue and meta-columns model, endogenous, FE, name, disag, level.

---

fcip_demand_sys_partial

*Residualize ("partial out") and build tilded / instrument variables*

---

### Description

If excluded instruments exist, runs first-stage OLS for each endogenous variable e: e ~ 1 + partial + included + excluded, storing the fitted values as instr_e. If partial is non-empty, it then regresses instr_e ~ 1 + partial and replaces instr_e <- instr_e - fitted(instr_e ~ partial) (i.e., removes the partial component; conceptually $\widehat{\text{instr}}_e(\text{partial})$ but expressed here without raw LaTeX macros).

Outcomes, included, and endogenous variables are residualized on partial to create tilda_<var> (or copied if partial is empty).

### Usage

```
fcip_demand_sys_partial(data, fields, partial_override = NULL)
```

**Arguments**

| | |
|---|---|
| `data` | A `data.frame`/`data.table` with referenced variables. |
| `fields` | List with: `outcome`, `endogenous`, `included`, optional `excluded`, optional `partial`. |
| `partial_override` | |
| | Optional character vector to override `fields$partial`. |

**Value**

List with `data`, `tilda_included`, `tilda_endogenous`, `tilda_excluded`.

---

`fcip_demand_sys_restricted`

*Optional restricted NLSUR step (when* `restrict` = TRUE*)*

---

**Description**

If enabled and feasible, estimates a nonlinear SUR with re-parameterized coefficients (negative exponents) using the optional **nlsur** package and appends "restricted_" rows to the results. Skips gracefully if **nlsur** is not installed or the step fails.

**Usage**

```
fcip_demand_sys_restricted(
  restrict,
  res,
  fit,
  data,
  outcome,
  tilda_endogenous,
  tilda_excluded,
  tilda_included
)
```

**Arguments**

| | |
|---|---|
| `restrict` | Logical flag; when `TRUE` attempt the restricted step. |
| `res` | Coefficient table from the unrestricted system (used to check signs). |
| `fit` | Fitted `systemfit` object from the unrestricted system. |
| `data` | Estimation data used to fit the restricted NLSUR model. |
| `outcome` | Character vector of outcome equation names (length 2 expected). |
| `tilda_endogenous` | |
| | Character vector of endogenous regressors used in the tilded system (e.g., `"tilda_z1"`). |
| `tilda_excluded` | Character vector of excluded instruments (e.g., `"instr_z1"`). |
| `tilda_included` | Character vector of included regressors (`"tilda_x1"`, ...). |

**Value**

A `data.frame` with rows for `Gamma`, `Theta`, and `Total` labeled `restricted_*`, or an empty `data.frame` if skipped.

## Note

This step uses `nlsur::nlsur()` if available; it is optional and should be listed under `Suggests` in `DESCRIPTION`.

---

fcip_demand_sys_tests    *System diagnostics: two-way robust first-stage F (+ optional approx. J)*

---

## Description

Produces diagnostics **without** re-running GMM:

- **FTest**: joint relevance of excluded instruments in each first stage, using the same two-way (pool by crop year) cluster-robust covariance via [`fcip_demand_sys_vcov()`](fcip_demand_sys_vcov()) with `kind = "lm"`. Reports the **minimum** F across endogenous regressors.

- **JTest** (optional): an *approximate* over-identification test computed as the sum of per-equation Sargan statistics $J_k \approx n_k R_k^2$ from regressions of equation residuals on that equation's instrument set. This is a quick check (not the system Hansen J).

## Usage

```
fcip_demand_sys_tests(g, h, data, fit, NFE, approx_j = FALSE)
```

## Arguments

| | |
|---|---|
| g | List of system equations (the same formulas passed to `systemfit`). |
| h | List of instrument formulas (the same formulas passed to `systemfit`). |
| data | Estimation `data.frame`/`data.table` containing all variables in g/h plus clustering columns `pool` and `crop_yr`. |
| fit | A fitted `systemfit` object (used for N and `residCov_*` extraction). |
| NFE | Integer: number of absorbed fixed effects (for reporting only). |
| approx_j | Logical, compute the approximate (non-robust) Sargan J as described above. Default `FALSE` (returns `NA` for JTest). |

## Value

A `data.frame` with rows: N, NFE, `residCov_11`, `residCov_22`, `residCov_12`, JTest, FTest.

fcip_demand_sys_vcov       *Two-way cluster-robust covariance for FCIP demand models*

### Description

Computes a Cameron-Gelbach-Miller two-way cluster-robust covariance matrix using inclusion-exclusion: $V = V_{pool} + V_{year} - V_{pool\_year}$. Works for both `systemfit` (stacked system) and `lm` (first-stage).

### Usage

```
fcip_demand_sys_vcov(
  object,
  data,
  kind = c("systemfit", "lm"),
  pool_col = "pool",
  year_col = "commodity_year",
  NFE = 0L,
  n_partial = 0L,
  n_eq = NULL
)
```

### Arguments

| | |
|---|---|
| object | Fitted model: either a `systemfit` or `lm`. |
| data | Estimation data containing pool and year identifiers. |
| kind | One of `c("systemfit","lm")`. If omitted, auto-detected. |
| pool_col | Name of the pool/cluster id column in `data` (default `"pool"`). |
| year_col | Name of the year/time id column in `data` (default `"crop_yr"`). |
| NFE | Integer; number of absorbed fixed effects (for df rescaling). |
| n_partial | Integer; count of variables partialed out per equation. |
| n_eq | Integer; number of equations (`length(object$eq)` for `systemfit`, 1 for `lm`). You can override if needed. |

### Details

**Rescaling.** Let $n$ be the number of observations (stacked across equations for `systemfit`). With $k_old$ the number of coefficients and $k_new = k_old + NFE + n\_partial * n\_eq$, the returned matrix is scaled by $(n - k_old - 1)/(n - k_new - 1)$.

**Row alignment (lm).** Rows used by `lm` are inferred from `rownames(model.matrix(object))`. If they cannot be mapped back to `data`, the first `nobs(object)` rows are used.

### Value

Covariance matrix aligned with `coef(object)`.

---

fcip_recodes_commodity_groupings

*fcip_recodes_commodity_groupings*

---

### Description

A combined dataset for fcip_recodes_commodity_groupings

### Usage

```
data(fcip_recodes_commodity_groupings)
```

### Format

A data frame with 3572 rows and 10 columns covering 1997-2025.

### Source

USDA-RMA, Actuarial Data Master - A00400 and A00420 supplemented data from legacy ADM files

---

fcip_recodes_insurance_plan

*fcip_recodes_insurance_plan*

---

### Description

A combined dataset for fcip_recodes_insurance_plan

### Usage

```
data(fcip_recodes_insurance_plan)
```

### Format

A data frame with 773 rows and 10 columns covering 1989-2025.

### Source

USDA-RMA, Actuarial Data Master - A00460 supplemented data from legacy ADM files

---

fcip_recodes_practice *fcip_recodes_practice*

---

### Description

A combined dataset for fcip_recodes_practice

### Usage

```
data(fcip_recodes_practice)
```

### Format

A data frame with 28639 rows and 8 columns covering 1997-2025.

### Source

USDA-RMA, Actuarial Data Master - A00510 supplemented data from legacy ADM files

---

fcip_recodes_type *fcip_recodes_type*

---

### Description

A combined dataset for fcip_recodes_type

### Usage

```
data(fcip_recodes_type)
```

### Format

A data frame with 232709 rows and 7 columns covering 1999-2025.

### Source

Generated internally, using harmonize_crop_type_codes()

```
fixed_effect_model_data_prep
```
*Prepare and demean data for fixed-effects models*

**Description**

This function

1. Filters to complete cases on the specified panel, time, weight, variables, and output

2. If `output` is NULL, creates a dummy output column filled with 1s

3. Drops any panel with only one observation

4. Computes within-panel means for the output + each variable in `varlist` (`_mean_i`)

5. Computes overall sample means for the same set of variables (`_mean`)

6. Replaces each variable in `varlist` by `value - within_panel_mean + overall_mean`

**Usage**

```
fixed_effect_model_data_prep(
  data,
  varlist,
  panel,
  time,
  wvar = NULL,
  output = NULL
)
```

**Arguments**

| | |
|---|---|
| `data` | A data.frame or data.table containing the data. |
| `varlist` | Character vector of variable names to be demeaned. |
| `panel` | Character vector of column name(s) defining the panel identifier. |
| `time` | Character scalar name of the time variable. |
| `wvar` | Character scalar name of a variable to keep but *not* demean (optional, default NULL). |
| `output` | Character scalar name of an output variable whose means are computed but not altered; if NULL, a dummy column named `"output"` is created (optional, default NULL). |

**Value**

A list with components

- **data**: a data.table containing
  - the original `panel`, `time`, `wvar`, `varlist`, and `output` columns
  - two mean columns for each of `c(output, varlist)`: `<name>_mean_i` (within-panel) and `<name>_mean` (overall)
- **NFE**: the number of panels with more than one observation

## See Also

Other Estimators panel models: [panel_based_spatial_smoothing_estimator](#)()

---

fsa_crop_linker          *Simulator Helper Datasets*

---

## Description

A combined dataset for fsa_crop_linker

## Usage

```
data(fsa_crop_linker)
```

## Format

A data frame with 8594 rows and 8 columns covering Inf–Inf.

## Source

Internal innovation

---

get_yu2018_instrument   *Formulate & Merge National Subsidy Rate Instrument (Yu et al., 2018)*

---

## Description

Downloads the historical Summary of Business RDS and computes national subsidy-rate instruments at specified coverage levels, following Yu et al. (2018).

## Usage

```
get_yu2018_instrument(
  dt,
  delivery_systems = c("RBUP", "FBUP"),
  plan_codes = c(1:3, 90, 44, 25, 42),
  coverage_levels = c(65, 75)
)
```

## Arguments

dt                sobcov

delivery_systems

                  Character vector. Delivery systems to include; default c("RBUP","FBUP").

plan_codes        Integer vector. Insurance plan codes to include; default c(1:3, 90, 44, 25, 42).

coverage_levels

                  Numeric vector. Percent coverage levels to keep; default c(65, 75).

### Value

A data.table with columns: commodity_year, subsidy_rate_65, subsidy_rate_75.

### See Also

Other FCIP instruments: `estimate_fcip_instruments`()

---

nass_census_state_beginning_farmer_and_rancher_data
*nass_census_state_beginning_farmer_and_rancher_data*

---

### Description

A combined dataset for nass_census_state_beginning_farmer_and_rancher_data

### Usage

```
data(nass_census_state_beginning_farmer_and_rancher_data)
```

### Format

A data frame with 255 rows and 16 columns covering Inf–Inf.

### Source

USDA NASS Quick Stats

---

nass_index_for_price_recived
*nass_index_for_price_recived*

---

### Description

A combined dataset for nass_index_for_price_recived

### Usage

```
data(nass_index_for_price_recived)
```

### Format

A data frame with 35 rows and 3 columns covering 1990-2024.

### Source

USDA NASS Quick Stats

nass_marketing_year_avg_price

*nass_marketing_year_avg_price*

### Description

A combined dataset for nass_marketing_year_avg_price

### Usage

```
data(nass_marketing_year_avg_price)
```

### Format

A data frame with 31139 rows and 7 columns covering 1866-2024.

### Source

USDA NASS Quick Stats

nass_state_rental_rates

*nass_state_rental_rates*

### Description

A combined dataset for nass_state_rental_rates

### Usage

```
data(nass_state_rental_rates)
```

### Format

A data frame with 1792 rows and 5 columns covering 1994-2025.

### Source

Output from get_state_rental_rates() function

nass_us_ag_price_index_monthly

*nass_us_ag_price_index_monthly*

### Description

A combined dataset for nass_us_ag_price_index_monthly

### Usage

```
data(nass_us_ag_price_index_monthly)
```

### Format

A data frame with 2255 rows and 8 columns covering Inf–Inf.

### Source

USDA NASS: https://www.nass.usda.gov/Charts_and_Maps/graphics/data

panel_based_spatial_smoothing_estimator

*Panel-based spatial smoothing estimator*

### Description

This function

1. Constructs spatially-varying treatment interactions (one variable per spatial unit)

2. Applies within-panel/time fixed-effects demeaning to both outcome and interactions

3. Fits an OLS model by hand (`lm.fit`) to recover one coefficient per spatial unit

### Usage

```
panel_based_spatial_smoothing_estimator(
  data,
  output,
  treatment,
  time,
  panel,
  spatialvar
)
```

## Arguments

| | |
|---|---|
| data | A `data.table` or `data.frame` containing panel data. |
| output | Name of the outcome variable (character scalar). |
| treatment | Name of the treatment variable whose spatial effects we estimate (character scalar). |
| time | Name of the time variable (character scalar). |
| panel | Name(s) of the panel identifier variable(s) (character vector). |
| spatialvar | Name of the spatial grouping variable (e.g. county FIPS; character scalar). |

## Details

Internally, we

1. Build `treatment_code = I(spatialvar==code) * treatment` for each spatial unit code.
2. Call `fixed_effect_model_data_prep()` to demean the outcome and all `treatment_code` variables.
3. Assemble the design matrix `X = [output_mean_i, treatment_*]` and response `y`.
4. Solve $\hat{\beta} = (\tilde{X}'\tilde{X})^{-1}\tilde{X}'\tilde{y}$ via `lm.fit`.
5. Return a row per spatial unit with its coefficient.

## Value

A `data.table` with columns:

- `estimate`: the estimated spatial-unit coefficient
- `county_fips`: the spatial unit identifier (5-digit FIPS)
- `state_code`, `county_code`: parsed FIPS components

## See Also

Other Estimators panel models: [fixed_effect_model_data_prep](fixed_effect_model_data_prep)()

# Index