GREENERS: ECONOMETRIC REFERENCE MANUAL

Flávio de Vasconcellos Corrêa

v0.1.0

1. LINEAR MODELS & ROBUST INFERENCE

Ordinary Least Squares (OLS)

The point estimator is the standard projection of y onto the column space of X:

$$\hat{\boldsymbol{\beta}}_{OLS} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

1.1.1 Variance-Covariance Estimation

Greeners implements three types of variance estimators $\widehat{\mathrm{Var}}(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}\hat{\Omega}(\boldsymbol{X}'\boldsymbol{X})^{-1}$.

- **Homoskedastic (Standard):** Assumes $E[\epsilon\epsilon'] = \sigma^2 I$.

$$\hat{\Omega} = \hat{\sigma}^2(\boldsymbol{X}'\boldsymbol{X})$$

- **White's Robust (HC1):** Consistent under arbitrary heteroskedasticity.

$$\hat{\Omega}_{HC1} = \sum_{i=1}^{n} \hat{\epsilon}_i^2 \boldsymbol{x}_i \boldsymbol{x}_i'$$

- **Newey-West (HAC):** Consistent under heteroskedasticity and autocorrelation up to lag L. Uses the Bartlett Kernel to ensure positive semi-definiteness.

$$\hat{\Omega}_{HAC} = \hat{\Omega}_0 + \sum_{l=1}^{L} w_l(\hat{\Omega}_l + \hat{\Omega}_l') \quad \text{where} \quad w_l = 1 - \frac{l}{L+1}$$

Rust Implementation:

```rust
use greeners::{OLS, CovarianceType};

// Estimates OLS with Newey-West HAC errors (Lags = 4)
let model = OLS::fit(&y, &x, CovarianceType::NeweyWest(4))?;
println!("Beta: {:.4}, StdErr: {:.4}", model.params[1], model.std_errors
    [1]);
```

Instrumental Variables (2SLS)

Used when regressors are endogenous ($E[\boldsymbol{X}\epsilon] \neq 0$). Instruments $\boldsymbol{Z}$ satisfy $E[\boldsymbol{Z}\epsilon] = 0$.

$$\hat{\boldsymbol{\beta}}_{IV} = (\hat{\boldsymbol{X}}'\hat{\boldsymbol{X}})^{-1}\hat{\boldsymbol{X}}'\boldsymbol{y} \quad \text{where} \quad \hat{\boldsymbol{X}} = \boldsymbol{Z}(\boldsymbol{Z}'\boldsymbol{Z})^{-1}\boldsymbol{Z}'\boldsymbol{X}$$

```
use greeners::IV;
let model = IV::fit(&y, &x, &z, CovarianceType::HC1)?;
```

2. GENERALIZED METHOD OF MOMENTS (GMM)

The estimator minimizes the quadratic distance of sample moments from zero. Let $g_n(\boldsymbol{\beta}) = \frac{1}{n}\boldsymbol{Z}'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$. The objective function is:

$$J(\boldsymbol{\beta}) = n \cdot g_n(\boldsymbol{\beta})'\boldsymbol{W} g_n(\boldsymbol{\beta})$$

Two-Step Efficient GMM

Greeners implements the optimal feasible GMM estimator:

1. **Step 1:** Estimate consistent $\hat{\boldsymbol{\beta}}_1$ using $\boldsymbol{W} = (\boldsymbol{Z}'\boldsymbol{Z})^{-1}$ (2SLS).
2. **Step 2:** Estimate the optimal weighting matrix $\hat{\boldsymbol{S}}$ (Hansen's Matrix) using Step 1 residuals:

$$\hat{\boldsymbol{S}} = \frac{1}{n}\sum_{i=1}^{n} \hat{\epsilon}_i^2 \boldsymbol{z}_i \boldsymbol{z}_i'$$

3. **Step 3:** Minimize $J(\boldsymbol{\beta})$ using $\boldsymbol{W}_{opt} = \hat{\boldsymbol{S}}^{-1}$.

Rust Implementation:

```
use greeners::GMM;
// Automatically performs the two-step procedure and computes Hansen's J
let res = GMM::fit(&y, &x, &z)?;
println!("J-Stat P-val: {:.4}", res.j_p_value);
```

3. DISCRETE CHOICE MODELS (MLE)

Estimates binary outcome models $P(y_i = 1|\boldsymbol{x}_i) = F(\boldsymbol{x}_i'\boldsymbol{\beta})$ via Newton-Raphson optimization.

$$\hat{\boldsymbol{\beta}}_{new} = \hat{\boldsymbol{\beta}}_{old} - \boldsymbol{H}^{-1}\nabla\mathcal{L}$$

Logit vs. Probit

- **Logit:** $F(z) = \Lambda(z) = (1 + e^{-z})^{-1}$. Hessian weighting matrix $\boldsymbol{W} = \text{diag}(p(1-p))$.
- **Probit:** $F(z) = \Phi(z)$ (Standard Normal). Hessian weighting relies on the ratio of PDF squared to CDF variance: $\frac{\phi^2}{\Phi(1-\Phi)}$.

Rust Implementation:

```
use greeners::{Logit, Probit};

let logit = Logit::fit(&y, &x)?;
let probit = Probit::fit(&y, &x)?; // Uses Normal CDF

println!("Logit Pseudo-R2: {:.4}", logit.pseudo_r2);
println!("Probit LogLikelihood: {:.4}", probit.log_likelihood);
```

4. CAUSAL INFERENCE (DID)

Estimates the *Average Treatment Effect on the Treated* (ATT) using the canonical 2×2 difference-in-differences design.

$$y_{it} = \beta_0 + \beta_1 \text{Treat}_i + \beta_2 \text{Post}_t + \delta_{ATT}(\text{Treat}_i \times \text{Post}_t) + \epsilon_{it}$$

The estimator identifies the causal effect under the *Parallel Trends Assumption*:

$$\hat{\delta}_{ATT} = (\bar{y}_{T,Post} - \bar{y}_{T,Pre}) - (\bar{y}_{C,Post} - \bar{y}_{C,Pre})$$

Rust Implementation:

```
use greeners::DiffInDiff;
// Automatically handles interaction terms and group means
let did = DiffInDiff::fit(&y, &treated_dummy, &post_dummy,
    CovarianceType::HC1)?;
println!("ATT Effect: {:.4}", did.att);
```

5. TIME SERIES DIAGNOSTICS

Augmented Dickey-Fuller (ADF)

Tests the null hypothesis of a Unit Root ($H_0 : \gamma = 0$) in the regression:

$$\Delta y_t = \alpha + \gamma y_{t-1} + \sum_{j=1}^{p} \delta_j \Delta y_{t-j} + \epsilon_t$$

The t-statistic is compared against MacKinnon's critical values.

Rust Implementation:

```
use greeners::TimeSeries;
// Tests stationarity. Returns test-stat and critical values.
let adf = TimeSeries::adf(&series, None)?;
if adf.is_stationary { println!("Series is I(0)"); }
```