Frisch–Waugh–Lovell Theorem Using Nonlinear Regression Residuals

Demonstrating the approximation of Frisch–Waugh–Lovell (FWL) theorem with nonlinear regression residuals using `NNS.reg()` and `np`.

Basic FWL result with OLS

```
set.seed(123)
x1 = rnorm(100)
x2 = rnorm(100)
y1 = 1 + x1 - x2 + rnorm(100)

r1 = residuals(lm(y1 ~ x2))
r2 = residuals(lm(x1 ~ x2))

# ols
coef(lm(y1 ~ x1 + x2))
```

```
## (Intercept)          x1          x2
##   1.1350654   0.8668285  -0.9761887
```

```
# fwl ols
coef(lm(r1 ~ -1 + r2))
```

```
##        r2
## 0.8668285
```

```
require(NNS)
require(data.table)
require(rgl)
require(np)
options(np.messages=FALSE)
```

`NNS` Residuals to Capture β_1

Step 1: `NNS` regression $y1$ on $x1$ and store residuals

```
nns_r1 = NNS.reg(x2,y1, plot=FALSE)$Fitted.xy$residuals
```

Step 2: `NNS` regression $x1$ on $x2$ and store residuals

```
nns_r2 = NNS.reg(x2,x1, plot=FALSE)$Fitted.xy$residuals
```

Step 3: OLS of `NNS` residuals is very close to β_1 of FWL result: 0.8668285

```
lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##     -0.1354       0.8540
```

Step 4: Reverse $x1$ and $x2$ for β_2 FWL result: -0.9761887

```
nns_r1 = NNS.reg(x1,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x1,x2, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##    -0.08349     -0.93324
```

Step 4a: Let's check `np`

```r
# Beta 1
np_1 = npreg(y1 ~ x2, residuals = TRUE, np.messages = FALSE)$resid

np_2 = npreg(x1 ~ x2, residuals = TRUE, np.messages = FALSE)$resid

coef(lm(np_1 ~ np_2))
```

```
## (Intercept)        np_2
##  0.00160974  0.81275266
```

```r
# Beta 2
np_1 = npreg(y1 ~ x1,residuals = TRUE)$resid

np_2 = npreg(x2 ~ x1,residuals = TRUE)$resid

coef(lm(np_1 ~ np_2))
```

```
## (Intercept)        np_2
##  0.01321047 -0.92755887
```

Increase the number of observations

```
set.seed(123)
x1 = rnorm(1000)
x2 = rnorm(1000)
y1 = 1 + x1 - x2 + rnorm(1000)

r1 = residuals(lm(y1 ~ x2))
r2 = residuals(lm(x1 ~ x2))
# ols
coef(lm(y1 ~ x1 + x2))
```

```
## (Intercept)          x1          x2
##   0.9790660   0.9785085  -0.9724932
```

```
# nns Beta 1
nns_r1 = NNS.reg(x2,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x2,x1, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##    -0.06086      0.97834
```

```
# nns Beta 2
nns_r1 = NNS.reg(x1,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x1,x2, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##     -0.0228      -0.9723
```

```
# np Beta 1
np_1 = npreg(y1 ~ x2,residuals = TRUE)$resid

np_2 = npreg(x1 ~ x2,residuals = TRUE)$resid

coef(lm(np_1 ~ np_2))
```

```
##   (Intercept)          np_2
## -0.0009034721  0.9720788103
```

```
# np Beta 2
np_1 = npreg(y1 ~ x1,residuals = TRUE)$resid

np_2 = npreg(x2 ~ x1,residuals = TRUE)$resid

coef(lm(np_1 ~ np_2))
```

```
##   (Intercept)          np_2
## -0.0006426288 -0.9716061300
```

Increase the number of observations. . . again

`np` takes way too long for this size regression. . .

```r
set.seed(123)
x1 = rnorm(10000)
x2 = rnorm(10000)
y1 = 1 + x1 - x2 + rnorm(10000)

r1 = residuals(lm(y1 ~ x2))
r2 = residuals(lm(x1 ~ x2))

# ols
coef(lm(y1 ~ x1 + x2))
```

```
## (Intercept)          x1          x2
##   0.9929192   1.0200607  -1.0031849
```

```r
# nns Beta 1
nns_r1 = NNS.reg(x2,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x2,x1, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##    0.006231     1.020007
```

```r
# nns Beta 2
nns_r1 = NNS.reg(x1,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x1,x2, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##    -0.04896     -1.00337
```

Completely Different Functional Form

```r
set.seed(123)
x1 = runif(1000)
x2 = runif(1000)
y1 = x1^2 * x2^2 + runif(1000)

r1 = residuals(lm(y1 ~ x2))
r2 = residuals(lm(x1 ~ x2))

# ols
coef(lm(y1 ~ x1 + x2))
```

```
## (Intercept)          x1          x2
##   0.1994670   0.4185737   0.3946490
```

```r
# nns Beta 1
nns_r1 = NNS.reg(x2,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x2,x1,  plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##     -0.0240       0.4152
```

```r
# nns Beta 2
nns_r1 = NNS.reg(x1,y1, plot=FALSE)$Fitted.xy$residuals

nns_r2 = NNS.reg(x1,x2, plot=FALSE)$Fitted.xy$residuals

lm(nns_r1 ~ nns_r2)
```

```
##
## Call:
## lm(formula = nns_r1 ~ nns_r2)
##
## Coefficients:
## (Intercept)       nns_r2
##    -0.03709      0.39440
```

```r
# np Beta 1
np_1 = npreg(y1 ~ x2,residuals = TRUE)$resid

np_2 = npreg(x1 ~ x2,residuals = TRUE)$resid

coef(lm(np_1 ~ np_2))
```

```
##  (Intercept)         np_2
## 0.0004302861 0.4141854361
```

```
# np Beta 2
np_1 = npreg(y1 ~ x1,residuals = TRUE)$resid

np_2 = npreg(x2 ~ x1,residuals = TRUE)$resid

coef(lm(np_1 ~ np_2))
```

```
##  (Intercept)         np_2
## 0.0001407786 0.3875706675
```