APPENDIX A

Short Simulation

The simulations were based on the following procedure:

1. Draw 800 observations from a standard normal distribution, x, using seed = 5 and specify the true relationship as $p = invlogit(\text{-}2\text{-}1x)$
2. Draw y from a binomial distribution, $y{\sim}B(p, N)/N$, where N is drawn from a Poisson distribution with $\lambda = 20$.
3. Estimate the various regression models and produce model predictions.
4. Repeat steps 1 to 3 10,000 times.
5. Generate graphs using 90% predictions bands.

Stata 15 Code for Estimation of Tobit Models and Useful Statistics and Quantities

The following example data set was used to generate Figure 2D as well as the examples throughout the paper.

```
// Load the example data set
webuse nlswork, clear

// Generate the PDV
drop if wks_work > 52
su wks_work
gen denominator = r(max)
gen work = wks_work/r(max) //% weeks employed

// Assign regressors to global variable
global variables collgrad ln_wage

// Two-limit Tobit with corners at zero and one
```

```
tobit work $variables, ll(0) ul(1)

// Predictions
* E(y | x) - the predictions on the observed pdv
margins, at(ln_wage = (0 (.5) 5.2)) predict(ystar(0,1))
      marginsplot // graphical presentation

* E(y | y < 1, x) - only consider the ones who didn’t work all year
margins, at(ln_wage = (0 (.5) 5.2)) predict(e(.,1))
      marginsplot // graphical presentation

* P(y < 1 | x) - the conditional probability of not working all year
margins, at(ln_wage = (0 (.5) 5.2)) predict(pr(.,1))
      marginsplot // graphical presentation

* P(y > 0 | x) - the conditional probability of working
margins, at(ln_wage = (0 (.5) 5.2)) predict(pr(0,.))
      marginsplot // graphical presentation

* Average marginal effects
margins, dydx(ln_wage) predict(ystar(0,1)) // E(y | x)
margins, dydx(ln_wage) predict(e(.,1)) // E(y | y < 1, x)
margins, dydx(ln_wage) predict(pr(.,1)) // P(y < 1 | x)
margins, dydx(ln_wage) predict(pr(0,.)) // P(y > 0 | x)

// Fractional probit
fracreg probit work $variables

* Average marginal effects
```

```
margins, dydx(ln_wage)

// Aggregated binomial regression
glm wks_work $variables, link(probit) family(binomial denominator) robust

* Average marginal effects on the scale of the numerator
margins, dydx(ln_wage)

* Average marginal effects on the PDV scale
margins, dydx(ln_wage) expression(normal(predict(xb)))

// R-squared measures incl. two-part models
* Tobit
tobit work $variables, ll(0) ul(1)
      capture drop y_hat
      quiet predict y_hat, ystar(0,1)
      local rdf = e(N) - e(rank)
      local N = e(N)
      quiet corr y_hat work
      local r2 = r(rho)^2
      di "r2 = " `r2'
      // adjusted
      local adjr2 = 1 - (1 - `r2') * ((`N' - 1)/`rdf')
      di "Adj. r2 = " `adjr2'
      drop y_hat

* TNH with corner at one
churdle linear work $variables, ///
      select($variables) ul(1)
```

```
	capture drop y_hat
	quiet predict y_hat
	local rdf = e(N) - e(rank)
	local N = e(N)
	*di `N'
	*di `rdf'
	quiet corr y_hat work
	local r2 = r(rho)^2
	di "r2 = " `r2'
	// adjusted
	local adjr2 = 1 - (1 - `r2') * ((`N' - 1)/`rdf')
	di "Adj. r2 = " `adjr2'
	*di r(rho)^2
	drop y_hat

* LH
** reverse code work as only the lower bound is allowed
gen reverse_work = 1-work
churdle exponential work $variables, ///
	select($variables) ll(0)
	capture drop y_hat
	quiet predict y_hat
	local rdf = e(N) - e(rank)
	local N = e(N)
	quiet corr y_hat reverse_work
	local r2 = r(rho)^2
	di "r2 = " `r2'
	// adjusted
	local adjr2 = 1 - (1 - `r2') * ((`N' - 1)/`rdf')
```

```
      di "Adj. r2 = " `adjr2'
      drop y_hat

* ET2T
** exclusion restriction needed - using race for illustration
capture generate dy = reverse_work > 0
capture gen log_reverse_work= log(reverse_work)
heckman log_reverse_work $variables, select(dy = $variables race) nolog
      capture drop yhatheck
      capture drop probpos
      capture drop x1b1
      capture drop x2b2
      predict probpos, psel
      predict x1b1, xbsel
      predict x2b2, xb
      scalar sig2sq = e(sigma)^2
      scalar sig12sq = e(rho)*e(sigma)^2
      display "sigma1sq = 1" " sigma12sq = " sig12sq " sigma2sq = " sig2sq
      generate yhatheck = exp(x2b2 + 0.5*(sig2sq))*(1 - normal(-x1b1-
sig12sq))
      local rdf = e(N) - e(rank)
      local N = e(N)
      quiet corr yhatheck reverse_work
      local r2 = r(rho)^2
      di "r2 = " `r2'
      // adjusted
      local adjr2 = 1 - (1 - `r2') * ((`N' - 1)/`rdf')
      di "Adj. r2 = " `adjr2'
      drop yhatheck probpos x1b1 x2b2
```

```
* Fractional probit r2
fracreg probit work $variables
      capture drop y_hat
      quiet predict y_hat
      local rdf = e(N) - e(rank)
      local N = e(N)
      quiet corr y_hat work
      local r2 = r(rho)^2
      di "r2 = " `r2'
      // adjusted
      local adjr2 = 1 - (1 - `r2') * ((`N' - 1)/`rdf')
      di "Adj. r2 = " `adjr2'
      drop y_hat

* Two-part fractional probit model
frm work $variables, linkfrac(probit) linkbin(probit) model(2P)

* Generalized two-part fractional probit model
** for details on estimation in Stata, please see Wulff (2019)
gen s = work > 0
cmp setup
cmp (work = $variables) (s = $variables race), ///
      indicators(s*$cmp_frac $cmp_probit)
```

REFERENCES

Wulff, J. N. (2019). Generalized two-part fractional regression with cmp. *The Stata Journal*, *19*(2), 375–389. https://doi.org/10.1177/1536867X19854017