A Supply Shifter Partially Identifies the Supply Slope

TheoryGuru applications

Load Economicreasoning package only if it is not already loaded

```
In[1]:= If[Length@Names["PLTools`*"] < 10,
  Get["http://economicreasoning.com"]]
```

Proof & Logic Tools 6.3

Type ERCommands for a list of commands in the package.

Introduction to Automated Economic Reasoning		
Tutorials:	Entering calculus	General Mathematica tips
Get started	Load extras	Browse examples

Notes

Interpret the vector variables (quantity, price, z, u_s, u_d) as demeaned.

In the Wolfram Language, quantity.price refers to the tensor DOT PRODUCT, NOT scalar multiplication. For TheoryGuru purposes, tensor means vector, so that the result of quantity.price is a scalar.

Setup

Supply-demand system

In[2]:=
```
u_s = quantity - β_s price - γ z;
u_d = quantity - β_d price;
```

In[4]:=
```
SlopeAssumptions = {β_s > 0, β_d < 0};
```

In[5]:=
```
Nontrivial = {u_s.u_s > 0, u_d.u_d > 0, z.z > 0, price.price > 0, quantity.quantity > 0, γ ≠ 0};
```

IV properties

In[6]:= zproperties =

$$\left\{z.u_s == 0,\ z.u_d == 0,\ \frac{(z.price)^2}{(z.z)\ (price.price)} < 1,\ \frac{(z.quantity)^2}{(z.z)\ (quantity.quantity)} < 1\right\};$$

In[7]:=
```
γbounds = 0 ≤ L < γ < H;
```

Define candidate bounds

In[8]:=

$$DefineCandidates = \left\{\beta_L == \frac{L\ z.z - quantity.z}{-price.z},\ \beta_H == \frac{H\ z.z - quantity.z}{-price.z}\right\};$$

Results

Demand slope is point identified

In[9]:=

$$TheoryGuru\left[\{zproperties, Nontrivial\},\ \beta_d == \frac{quantity.z}{price.z}\right]$$

Out[9]= True

Supply shifter moves price and quantity in opposite directions

In[10]:=
```
TheoryGuru[{zproperties, Nontrivial, SlopeAssumptions},
 SameSign[-price.z, γ]
  ∧
  SameSign[quantity.z, γ]]
```

Out[10]= True

Derive the supply slope’s identified set

In[11]:=
```
idset = TheoryPossibilities[{zproperties, Nontrivial, SlopeAssumptions, γbounds},
   {βs, quantity.z, price.z, z.z, L, H},
   True]
```

```
You are in vector mode: include the hypothesis if it is
  necessary for classifying your assumption symbols as scalars vs. vectors.
```

Out[11]= L ≥ 0 && price.z < 0 && quantity.z > 0 && z.z > 0 && β_s > 0 &&
quantity.z < H z.z + price.z β_s && L z.z + price.z β_s < quantity.z

Equivalent way of writing idset

In[12]:=
```
TheoryGuru[{zproperties, Nontrivial, SlopeAssumptions, γbounds, DefineCandidates},
  0 ≤ βL < βs < βH ∨
   βL < 0 < βs < βH]
```

Out[12]= True

In[13]:=
```
TheoryOverlap[
  {zproperties, Nontrivial, SlopeAssumptions, γbounds, DefineCandidates},
  idset,
  0 ≤ βL < βs < βH ∨
   βL < 0 < βs < βH]
```

Out[13]= {quantity.z > 0 ∧ z.z > 0 ∧ β_s > 0 ∧ L ≥ 0 ∧ price.z < 0 ∧ quantity.z < H z.z + price.z β_s ∧ L z.z + price.z β_s < quantity.z, (β_s > 0 ∧ β_L < 0 ∧ β_s < β_H) ∨ (β_s > 0 ∧ β_L < β_s ∧ β_s < β_H) ∨ (β_L ≥ 0 ∧ β_L < 0 ∧ β_s < β_H) ∨ (β_L ≥ 0 ∧ β_L < β_s ∧ β_s < β_H)} are equivalent

Variable interpretations

Let the γ bounds be in units of β_s

In[14]:=
```
γbounds2 = 0 ≤ L βs < γ < H βs;
```

In[15]:= DefineCandidates2 = $\left\{\beta_L == \frac{\text{quantity.z}}{\text{price.z} + \text{H z.z}}, \beta_H == \frac{\text{quantity.z}}{\text{price.z} + \text{L z.z}}\right\}$;

In[16]:=
```
idset2 = TheoryPossibilities[{zproperties, Nontrivial, SlopeAssumptions, γbounds2},
   {βs, quantity.z, price.z, z.z, L, H},
   True]
```

```
You are in vector mode: include the hypothesis if it is
  necessary for classifying your assumption symbols as scalars vs. vectors.
```

Out[16]= L ≥ 0 && price.z < 0 && quantity.z > 0 && z.z > 0 && β_s > 0 &&
quantity.z < (price.z + H z.z) β_s && (price.z + L z.z) β_s < quantity.z

Equivalent way of writing idset2

In[17]:=
```
TheoryOverlap[
 {zproperties, Nontrivial, SlopeAssumptions, γbounds2, DefineCandidates2},
 idset2,
 0 ≤ βL < βs < βH
  ∨
  βH < 0 < βL < βs]
```

Out[17]=

$\{quantity.z > 0 \wedge z.z > 0 \wedge \beta_s > 0 \wedge L \geq 0 \wedge price.z < 0 \wedge quantity.z < \beta_s (H\, z.z + price.z) \wedge \beta_s (L\, z.z + price.z) < quantity.z,$ are equivalent

$(\beta_L > 0 \wedge \beta_H < 0 \wedge \beta_L < \beta_s) \vee (\beta_L \geq 0 \wedge \beta_L < \beta_s \wedge \beta_s < \beta_H)\}$