

# Identifying Taylor Rules In Macro-Finance Models

David Backus, Mikhail Chernov,  
and Stanley Zin

Cooley Conference @ NYU | October 5, 2013

# Identifying the Taylor rule

- ▶ Long-term goal
  - ▶ Integrate models of bond pricing and monetary policy
- ▶ Open question
  - ▶ Can we identify the monetary policy parameter(s)?
  - ▶ Can we distinguish systematic policy from shocks to it?

# Identifying the Taylor rule

- ▶ Common views
  - ▶ Macro: not identified
  - ▶ Finance: can't extract policy component from affine model
- ▶ What we do
  - ▶ Describe conditions for identification
  - ▶ Revisit earlier work on dynamic rational expectations models

# Outline

- ▶ Setup
- ▶ Two examples
- ▶ Rational expectations solutions and identification
- ▶ More complex models
- ▶ What if you don't see the state?

# Setup

► State

$$x_{t+1} = Ax_t + Cw_{t+1}$$

$$V_x = AV_xA^\top + CC^\top$$

► Shocks

$$s_{it} = d_i^\top x_t$$

► Identification issue: we observe state  $x$ , but not shock  $s_i$

# Cochrane's example

► Model

$$i_t = r + E_t \pi_{t+1} \quad (\text{Fisher equation})$$

$$i_t = r + \tau \pi_t + s_t \quad (\text{Taylor rule})$$

► Expectational difference equation

$$E_t \pi_{t+1} = \tau \pi_t + s_t$$

► Solution:  $\pi_t = b^\top x_t$  with  $b^\top = -d^\top (\tau I - A)^{-1}$

► Identification problem: any  $\tau$  works for some  $d$

$$b^\top A = \tau b^\top + d^\top$$

# Affine example

► Model

$$m_{t+1}^{\$} = -\lambda^{\top} \lambda - \delta x_t + \lambda^{\top} w_{t+1} \quad (\text{Pricing kernel})$$

$$i_t = -\log E_t m_{t+1}^{\$} = \delta^{\top} x_t \quad (\text{Euler equation})$$

► Expectational difference equation

$$E_t \pi_{t+1} = \tau \pi_t + s_t$$

► Is second equation a Taylor rule?

$$\delta = \tau b^{\top} + d^{\top}$$

# Questions

- ▶ Would an extra shock help?

$$i_t = E_t \pi_{t+1} + s_{1t} \quad (\text{Fisher equation})$$

$$i_t = \tau \pi_t + s_{2t} \quad (\text{Taylor rule})$$

- ▶ If shocks are independent, can use  $s_{1t}$  as an instrument
- ▶ Would long rates help?
  - ▶ May span state, but we see state anyway