Macroeconomics

Lecture 16: incomplete markets, part two

Chris Edmond

1st Semester 2019

This class

- Solving the Huggett (1993) model
- A simple 2-state Markov example
- Approximating continuous-state processes using Markov chains (the Tauchen-Hussey (1991) procedure)

Discrete state space approximation

- Consider discrete grid of asset levels

$$a_{\min} < \ldots < a_i < \ldots < a_{\max} \qquad i = 1, ..., n$$

where $a_{\min}$ is the borrowing constraint

- Suppose endowment process is a Markov chain with support

$$y_{\min} < \ldots < y_k < \ldots < y_{\max} \qquad k = 1, ..., m$$

and transition probabilities

$$\pi_{kl} = \text{Prob}[y' = y_l \,|\, y = y_k]$$

Discrete state space approximation

- Given price q, let c_{ijk} denote current consumption if current asset level is $a = a_i$, the asset level for next period is $a' = a_j$ and the current endowment is y_k

$$c_{ijk} = a_i + y_k - q a_j$$

- We will need to be careful to respect the feasibility constraints

$$a_{\min} \leq a_j \leq q^{-1}(a_i + y_k)$$

- Let u_{ijk} denote the flow utility associated with c_{ij}

$$u_{ijk} = u(c_{ijk})$$

Discrete state space approximation

- In this notation, our value function is an $n \times m$ matrix $\boldsymbol{V}$ with typical element

$$v_{ik} = \max_{j} \left[u_{ijk} + \beta \sum_{l=1}^{m} v_{jl} \pi_{kl} \right]$$

- The maximization on the RHS implies a policy function, i.e., an $n \times m$ matrix $\boldsymbol{G}$ with typical element

$$g_{ik} = a_{j^*}, \qquad j^* = \operatorname*{argmax}_{j} \left[u_{ijk} + \beta \sum_{l=1}^{m} v_{jl} \pi_{kl} \right]$$

State vector

- Endowment process y_t follows an exogenous Markov chain

$$\text{Prob}[y_{t+1} \,|\, y_t]$$

- State $s_t = (a_t, y_t)$ follows an *endogenous* Markov chain

$$\text{Prob}[s_{t+1} = \,|\, s_t]$$

- Need to calculate transition probabilities for this Markov chain

Transition probabilities for the state vector

- Write the transition probabilities for the state

$$\text{Prob}[a_{t+1}, y_{t+1} \,|\, a_t, y_t]$$

- But the distribution of y_{t+1} is independent of a_{t+1} so this is

$$\text{Prob}[a_{t+1} \,|\, a_t, y_t] \times \text{Prob}[y_{t+1} \,|\, y_t]$$

- But a_{t+1} is given by the optimal policy $a_{t+1} = g(a_t, y_t)$ so

$$\text{Prob}[a_{t+1} \,|\, a_t, y_t] = \mathbb{1}[a_{t+1} = g(a_t, y_t)]$$

where $\mathbb{1}[\cdot]$ denotes the indicator function

Transition probabilities for the state vector

- Hence

$$\text{Prob}[a_{t+1}, y_{t+1} \,|\, a_t, y_t] = \mathbb{1}[a_{t+1} = g(a_t, y_t)] \times \text{Prob}[y_{t+1} \,|\, y_t]$$

- So once we have computed the policy function $g(a, y)$ we can also compute these transition probabilities

- In this sense, the Markov process for the state $s_t = (a_t, y_t)$ is a coupling of the exogenous process for y_t with the policy function

Huggett: 2-state example

Uses Matlab files in "*huggett_example.zip*" in LMS

```
%%%%% economic parameters

beta  = 0.95;       %% time discount factor
alpha = 1.5;        %% CRRA (=1/IES)

%%%%% 2-state markov chain for endowments

ymin  = 0.1;
ymax  = 1.0;
ygrid = [ymin;ymax];

p11 = 0.500; p12 = 1 - p11;
p22 = 0.925; p21 = 1 - p22;

P   = [p11,p12;p21,p22];
```

Asset grid

```
%%%%% asset grid

phi  =  (ymin/(1-beta))-eps;      %% borrowing constraint

na       = 1000;
amin     = -phi;
amax     = 12;

agrid    = nodeunif(na , amin, amax);
```

State vector

```
%%%% state grid

s  = gridmake(agrid,ygrid); % ns-by-2 matrix where ns=na*ny
ns = size(s,1);

a  = s(:,1);
y  = s(:,2);
```

Inner dynamic programming loop

```
%%%%% initialize inner dynamic programming loop

v        = log(0.5*y)/(1-beta);
iter     = 0;

for i=1:max_iter,

RHS       = u+beta*kron(P,ones(na,1))*reshape(v,na,2)';

[Tv,argmax] = max(RHS,[],2);

%%%%% policy that attains the maximum

g = a(argmax);
```

Kronecker product calculates conditional expectation of value function next period

Transition matrix for the state

```
%%%%% construct transition matrix for the state s=(a,y)

A = zeros(ns,na);
Q = zeros(ns,ns);

PP = kron(P,ones(na,1));

for s=1:ns,

    A(s,:) = (agrid==g(s))';  %% puts a 1 if g(s)=a

    Q(s,:) = kron(PP(s,:),A(s,:));

end
```

This is fiddly

Compute stationary distribution

```
%%%%% compute stationary distribution

[eig_vectors,eig_values] = eig(Q');
[~,arg] = min(abs(diag(eig_values)-1));
unit_eig_vector = eig_vectors(:,arg);

mu = unit_eig_vector/sum(unit_eig_vector);
```

Be careful to check the orientation of the transition matrix

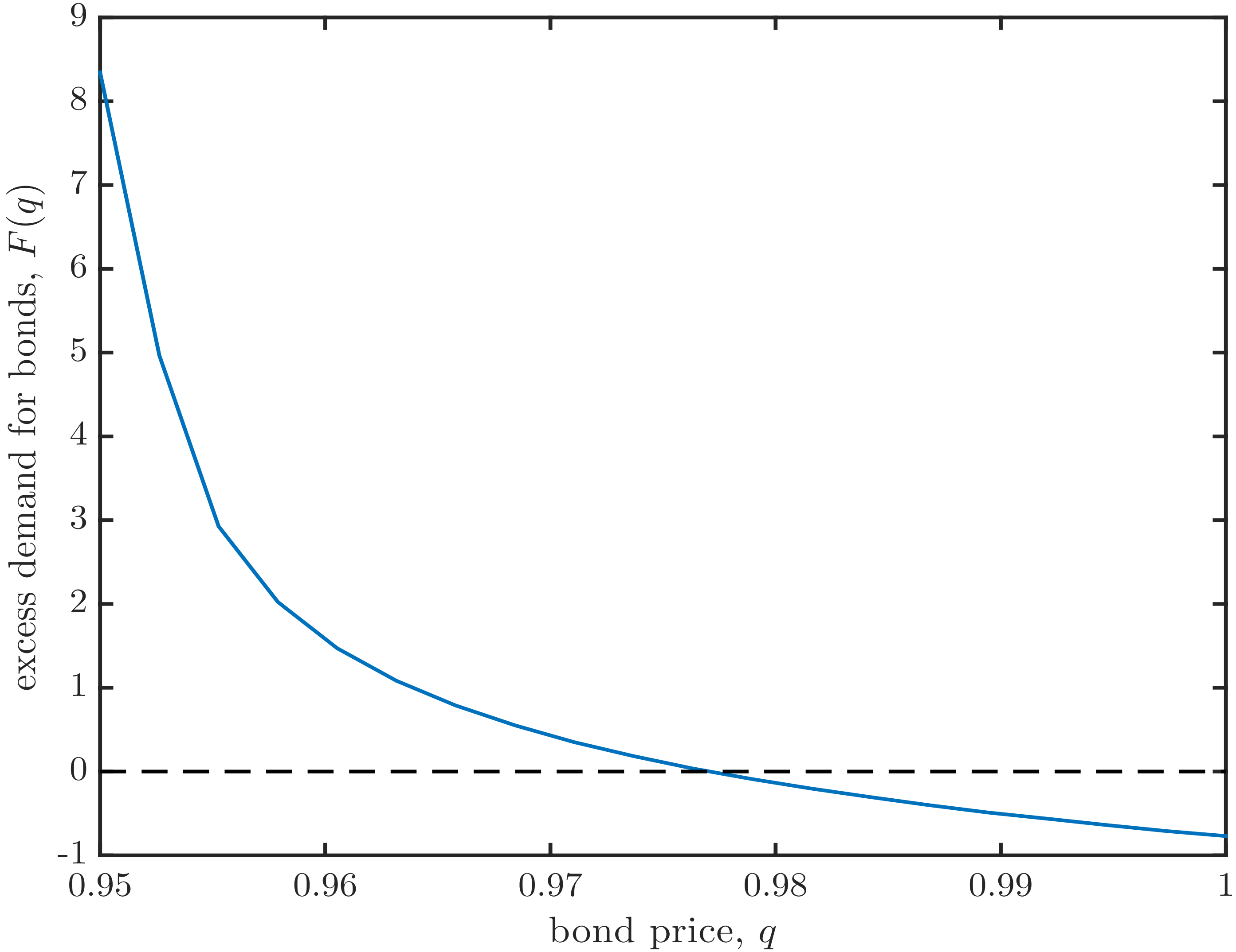
Check market clearing

```
%%%%% check market clearing

z = sum(mu.*g);
```

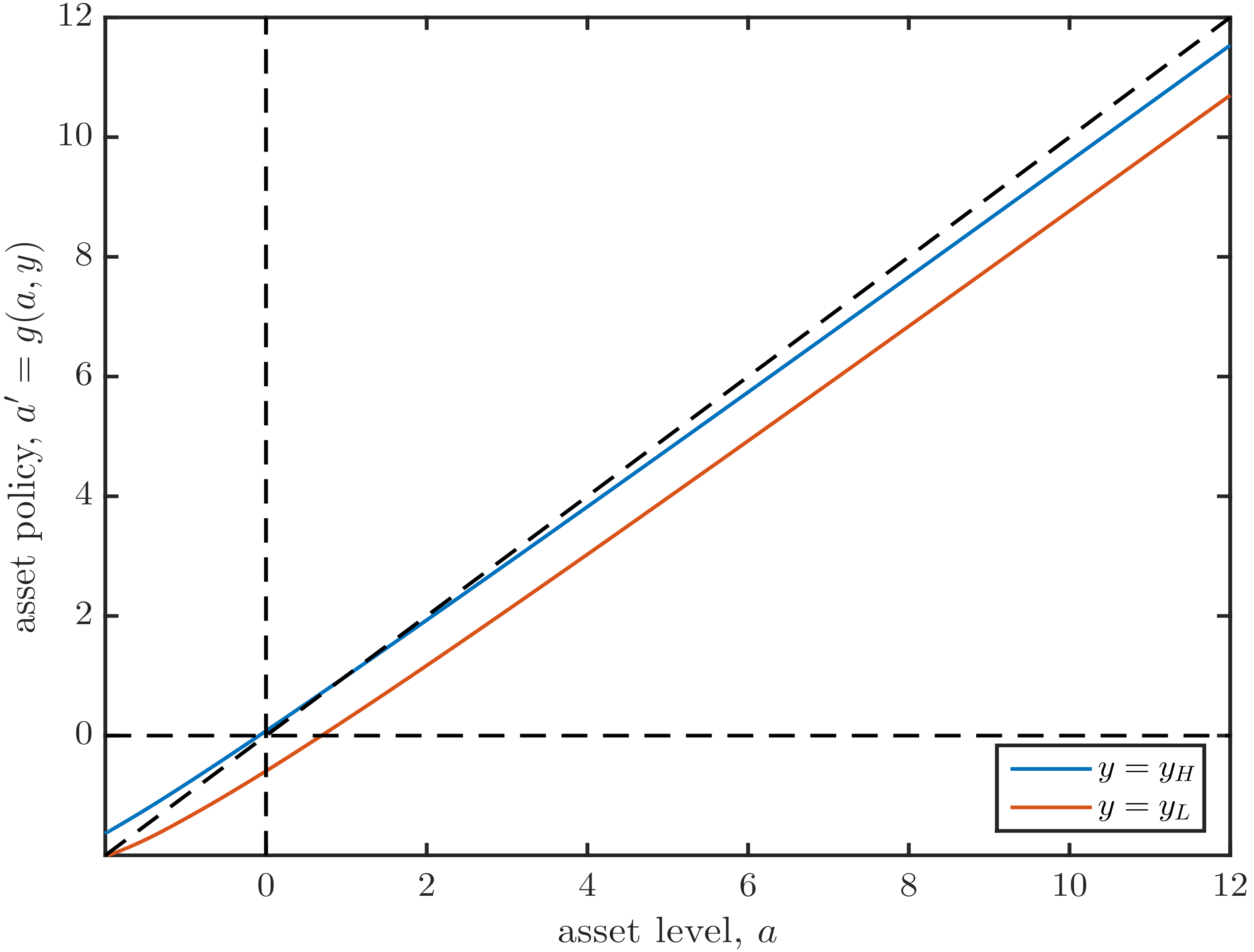
Find q that solves $F(q) = 0$

```
%%%%% find q that zeros out market-clearing condition

qmin  =  beta+eps;
qmax  =  1    -eps;

%fmin = findq(qmin,parameters,max_iter,penalty,tol);
%fmax = findq(qmax,parameters,max_iter,penalty,tol);

optset('bisect','tol',tol) ;
tic
q = bisect('findq',qmin,qmax,parameters,max_iter,penalty,tol);
toc
```

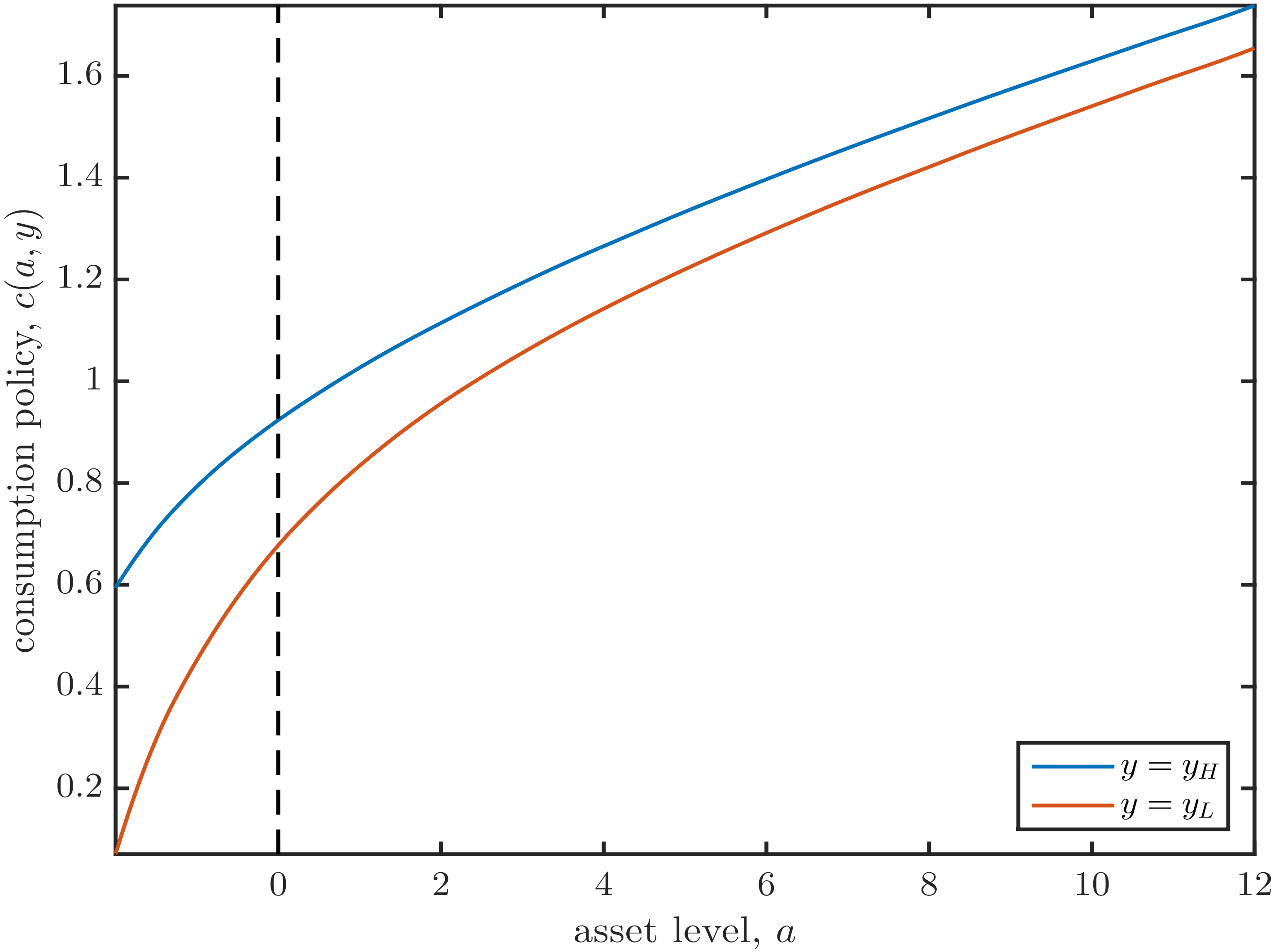
Excess demand $F(q)$



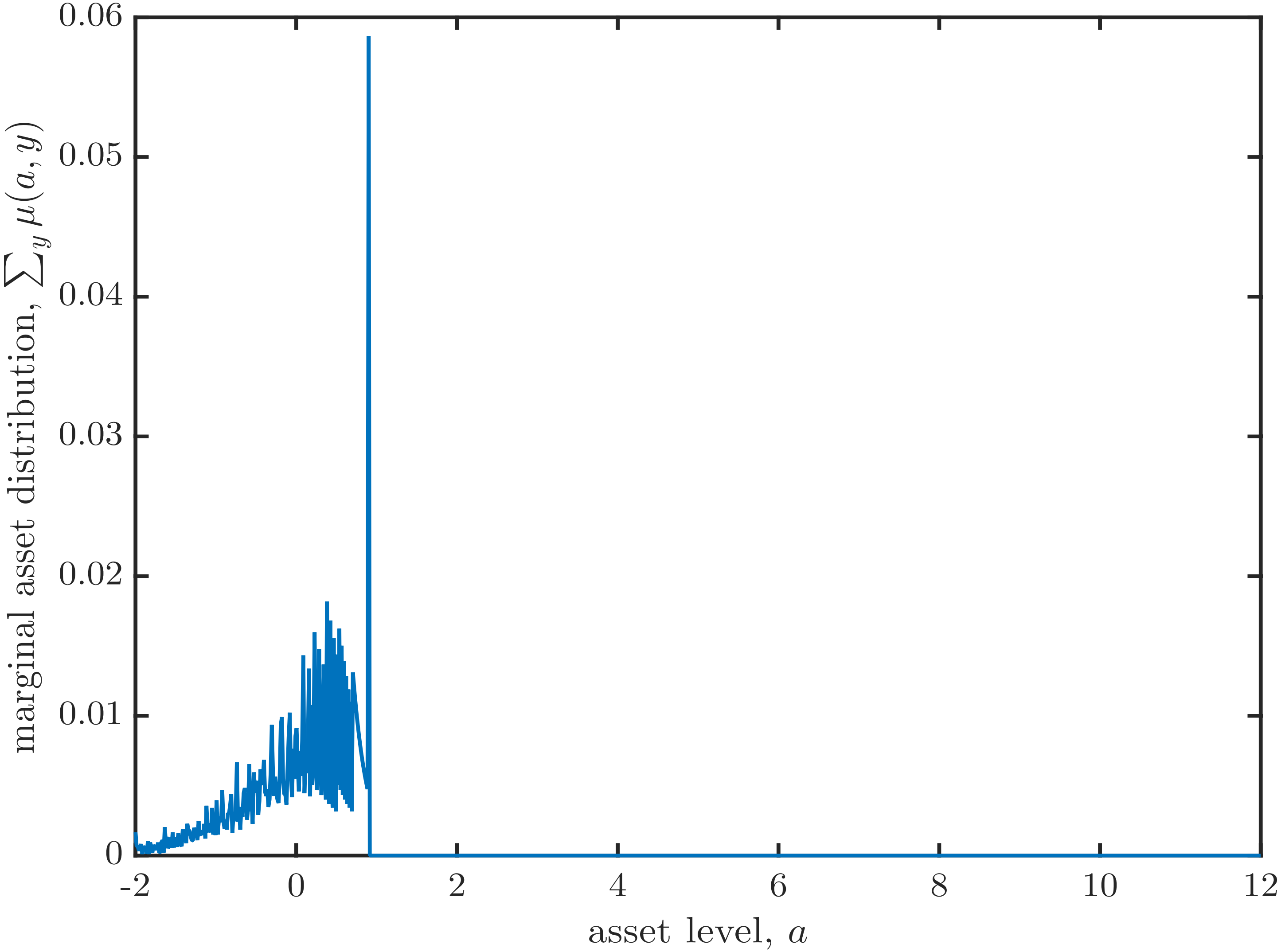
Asset policy $a' = g(a, y)$



Consumption policy $c(a, y)$



Marginal asset distribution $\sum_y \mu(a, y)$



Tauchen/Hussey (1991) approximation

- Can use quadrature to obtain discrete Markov chain approximation to process with continuous support

- Density for $x_{t+1} = x'$ conditional on $x_t = x$

$$p(x' \,|\, x)$$

- Discretize support of x to n quadrature nodes x_i and replace $p(x' \,|\, x)$ by $n \times n$ matrix of transition probabilities

$$p_{ij} = \frac{p(x_j \,|\, x_i)\frac{w_j}{\omega(x_j)}}{\sum_{j'=1}^{n} p(x_{j'} \,|\, x_i)\frac{w_{j'}}{\omega(x_{j'})}}, \qquad i, j = 1, ..., n$$

where w_i are quadrature weights for x_i and $\omega(x)$ is a 'regularity function' that controls the quality of the approximation to higher moments

Tauchen/Hussey (1991) example

- Suppose we want to approximate AR(1) with Markov chain

$$p(x' \mid x) = \frac{1}{\sigma}\phi\left(\frac{x' - (1-\rho)\bar{x} - \rho x}{\sigma}\right)$$

- Lookup quadrature nodes x_i, weights w_i for normal $N(\mu, \hat{\sigma}^2)$. Set regularity function to

$$\omega(x) = \frac{1}{\hat{\sigma}}\phi\left(\frac{x - \bar{x}}{\hat{\sigma}}\right)$$

- Tauchen/Hussey (1991) advocate $\hat{\sigma} = \sigma$ (innovation std dev). But Floden (2008) advocates that for highly persistent processes

$$\hat{\sigma} = \theta\sigma + (1-\theta)\bar{\sigma}, \qquad \theta = 1/2 + \rho/4$$

($\rho \approx 1 \Rightarrow$ more weight in tails, better match conditional variance)

Tauchen-Hussey example

Uses Matlab files in “*tauchen_hussey_example.zip*” in LMS

```
%%%%% AR1 process

phi    = 0.95; %% AR1 coefficient
sigeps = 0.10; %% innovation std deviation

% long run moments
mu     = 0;
sigma  = sigeps/sqrt(1-phi^2);
```

Tauchen-Hussey example

```
%%%% discrete-state approximation to AR1

N      = 33;  %% number of nodes
floden = 1;   %% indicator for Floden correction

[nodes,weights,P] = get_tauchen_hussey(mu,sigeps,phi,N,floden);
```

Tauchen-Hussey example

Inside the function file

```
%%%%% INDICATOR FOR FLODEN CORRECTION
if floden==1,

w       = 0.5 + phi/4;
sigx    = sigeps/sqrt(1-phi^2); %% unconditional std dev

flodensigma  = w*sigeps + (1-w)*sigx;

else

flodensigma  = sigeps;

end

%%%%% LOOKUP QUADRATURE NODES AND WEIGHTS
[nodes,weights] = qnwnorm(N,mu,flodensigma^2);
```

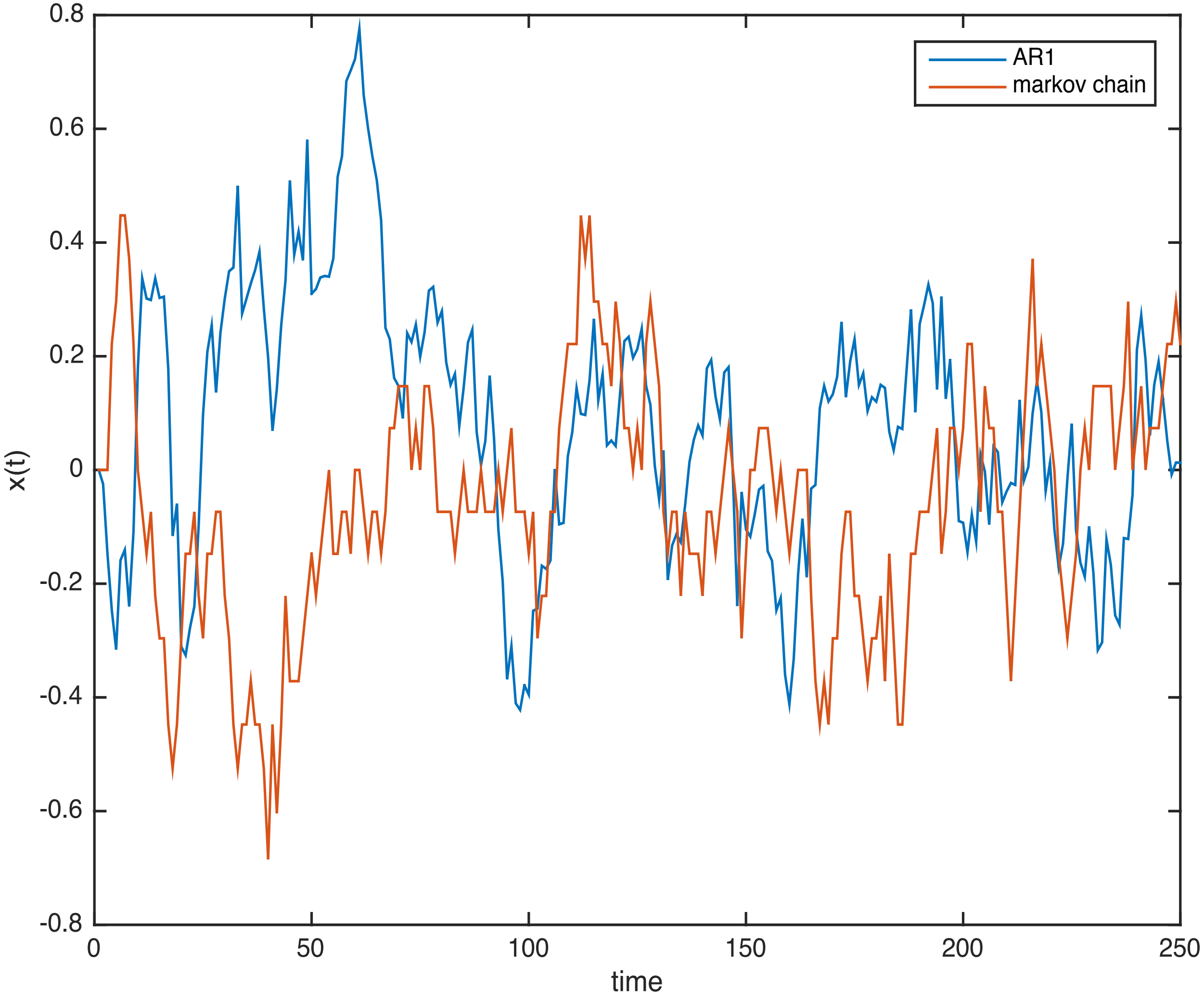
Tauchen-Hussey example

```
%%%%% CONSTRUCT TRANSITION MATRIX
%p[ij] = f[ij]*quadrature_weight(j)/regularity_function(j)

for i=1:N,
    for j=1:N,

%%%%% conditional mean
mean    = (1-phi)*mu + phi*nodes(i);

%%%%% given we are at node(i), what is likelihood of node(j)?
F(i,j)  = normpdf(nodes(j),mean,sigeps);

%%%%% multiply by quadrature weights
P(i,j)  = F(i,j)*weights(j);

%%%%% divide by regularity_function
regularity_function(j) = normpdf(nodes(j),mu,flodensigma);

P(i,j) = P(i,j)/regularity_function(j);
```

Tauchen-Hussey example

```
%%%%% normalize so rows sum to 1
for i=1:N,

    P(i,:) = P(i,:) / sum(P(i,:),2);

end
```

Markov chain vs. AR1 with same moments



Next class

- Aiyagari (1994)
  - production economy with capital and labor (heterogeneous agents version of stochastic growth model)
  - but still no aggregate risk