Dave Backus @ NYU

Notes on Gertler's "OG" Model*

(Started: June 17, 2013; Revised: July 18, 2013)

Mark Gertler (Carnegie-Rochester, 1999) has a nice paper that simplifies the usual OG setup by giving people two types: working and retired. If working, there's a constant probability of becoming retired each period, and if retired, there's a constant probability of death. This retains the flavor of a life-cycle without having to keep track of people at lots of different ages.

1 Demography

At date t, we have N_t people working and R_t people retired. Working people have children at rate φ and retire at rate ω. Retired people die at rate μ. Therefore average time working is $1/\omega$ and average time retired is $1/\mu$ (they're geometric random variables).

The components of the population thus evolve according to

$$\begin{bmatrix} N_{t+1} \\ R_{t+1} \end{bmatrix} = \begin{bmatrix} 1+\varphi-\omega & 0 \\ \omega & 1-\mu \end{bmatrix} \begin{bmatrix} N_t \\ R_t \end{bmatrix}.$$

Let all these parameters be positive, with $\varphi > \omega$. Then the population has a long-term growth rate of $1+\varphi-\omega$, the maximal eigenvalue of the matrix. The stationary dependency ratio is therefore

$$R/N = \omega/(\varphi+\mu-\omega),$$

the associated eigenvector.

2 Preferences

Working-age agents supply one unit of labor inelastically. Agents of type

..

*Working notes, no guarantee of accuracy or sense.

A Code

A `gertler.m` (attached)

```
clear all;

% preference parameter
beta = .96;
sigma = 1;

% production
alpha = 1/3;
delta = 0.07;

% markov transition probabilities
phi   = 0.025;  % student -> working
omega = 0.025;  % working -> retired
mu    = 0.050;  % retired -> the unknown

dim = 901;

% Normalize population size
pop = 1;
% Ratio of retired to working
RN = omega/(phi+mu-omega);
% Number of people in working age
N = pop/(1+RN);

% Factor prices
R = 1.03;
w = (1-alpha)*((R-(1-delta))/alpha)^(alpha/(alpha-1));
% w computed according to CD production technology



% Construct consumption and utility matrices
a = linspace(.000001,25,dim);
for i = 1 : dim
    for j = 1 : dim
        cr(i,j) = R*a(i)  - a(j);
        if(cr(i,j) > 0)
            if sigma ==1
                ur(i,j) = log(cr(i,j));
            else
                ur(i,j) = (cr(i,j)^(1-sigma))/(1-sigma);
            end
        else
            ur(i,j) = -inf;
        end
    end
end

for i = 1 : dim
    for j = 1 : dim
        cn(i,j) = R*a(i) + w - a(j);
```

```
        if(cn(i,j) > 0)
            if sigma ==1
                un(i,j) = log(cn(i,j));
            else
                un(i,j) = (cn(i,j)^(1-sigma))/(1-sigma);
            end
        else
            un(i,j) = -inf;
        end
    end
end



%% Value function iterations
fprintf('Value fn iterations ..... \n \n')
tolcrit = 10E-6;
maxit = 10000;

% retired guys
vr  = zeros(dim,1);
dr  = zeros(dim,1);
Tvr = zeros(dim,1);
diff = 1;
iter = 0;
while (diff > tolcrit) & (iter < maxit)
    for i = 1 : dim
        [Tvr(i) dr(i)] = max(ur(i,:) + beta*(1-mu)*vr');
    end
    diff = norm(Tvr-vr);
    vr = Tvr;
    iter = iter + 1;
end
clear i diff iter Tvr;

% retired guys
vn  = zeros(dim,1);
dn  = zeros(dim,1);
Tvn = zeros(dim,1);
diff = 1;
iter = 0;
while (diff > tolcrit) & (iter < maxit)
    for i = 1 : dim
        [Tvn(i) dn(i)] = max(un(i,:) + beta*(omega*vr' + (1-omega)*vn'));
    end
    diff = norm(Tvn-vn);
    vn = Tvn;
    iter = iter + 1;
end
clear i diff iter Tvn;



% simulations
fprintf('Simulations ..... \n \n')
nsim = 1000;
respos = zeros(200,nsim);
```

```
resassets = zeros(200,nsim);
resavga = zeros(nsim,1);

% set seed for random number generator
rng(1234786)

for i = 1 : nsim
    alive   = true;
    retired = false;
    ctr = 1;
    % start with assets almost zero (position 1)
    respos(ctr,i) = 1;
    resassets(ctr,i) = a(respos(ctr,i));
    ctr = ctr + 1;
    while alive == true
        while retired == false
            respos(ctr,i) = dn(respos(ctr-1,i));
            resassets(ctr,i) = a(respos(ctr,i));
            ctr = ctr + 1;
            draw = rand;
            if draw <= omega
                retired = true;
            end
        end
        respos(ctr,i) = dr(respos(ctr-1,i));
        resassets(ctr,i) = a(respos(ctr,i));
        ctr = ctr + 1;
        draw = rand;
        if draw <= mu
            alive = false;
            resavga(i) = sum(resassets(1:(ctr-1),i))/(ctr-1);
        end
    end
end

fprintf('Aggregate asset holdings: %2.4f \n',mean(resavga))
```