Matthew Aaron Looney 09/24/2017

Resource Use - Cake Eating Problem

Model the cake eating problem using Deterministic Discrete Markov Decision Modeling.

We will use the Matlab function (ddpsolve) developed by Miranda and Fackler (2002) in their CompEcon Toolbox for Matlab.

Problem information:

Initial resource stock of cake = s^0.

Time periods $t = \{1,2,...,T\}$.

Utility function = $u(x) = x^{\alpha}, 0 < \alpha < 1$ where the function is increasing and strictly concave.

No borrowing allowed.

The social discount rate is given by β.

The goal is to maximize the present value of the total social welfare over T periods.

Results and Interpretation:

After setting up the problem in Matlab and running the program we observe the following results:

- When we systematically change the α parameter (holding β constant) which represents the Elasticity of Utility, from a low value (0.1) to a high value (0.9) we see a progressive move away from risk aversion toward more risk tolerant behavior. Or when societies preference is to consume more cake over the long run we would expect to see low values of α.

- When we systematically change our β parameter (holding α constant) which represents the discount factor, from a high value (0.9) to a low value (0.1) we see a progressive move away from long run preference to short run preference. In other words, when our β parameter is high, society places more value on consumption in the future and consumes less today.

Matlab Code Notes:

I have attached two scripts:

cakeEating_staticPlot.m

The first script generates a single plot.

The first script allows for changing values of alpha and beta to study the relationship between changes in discount factor and elasticity of utility on optimal consumption.

cakeEating_iterPlots.m

The second script generates two plots.

The first plot holds alpha constant at (0.5) and changes beta over the range (0.1 : 0.9).

The second plot holds beta constant at (0.5) and changes alpha over the range (0.1 : 0.9).

The plots show the systematic changes in the two parameters over time on optimal consumption preference for a given set of values of alpha and beta.

The plot is attached to the end of this report.

Matlab Code - Script 1:

```
% cakeEating_staticPlot.m
% need utilityFn.m file, a function to define the utility function

% References:

% Adda, J., & Cooper, R. W. (2003). Dynamic economics : quantitative
% methods and applications. (R. W. Cooper, Ed.). Cambridge, Mass. : MIT Press.

% Miranda, M. J., & Fackler, P. L. (2002). Applied computational economics
% and finance. (P. L. Fackler, Ed.). Cambridge, Mass. : MIT Press.

% Scripts adapted from Dr. Chenggang Wang and Ming Kang

% housekeeping
close all
clear
clc
fprintf('\nCake Eating MODEL\n')

% Enter model parameters
finite = 0;     % finite horizon 0/1
T = 10;         % Time horizon
```

```
% A discount factor, beta, which indicates the agents measure of time
% preference, with a higher beta meaning valuing future more and thus
% being more patient.

% Elasticity of Utility, alpha, a measure of risk preference. A lower
% value of alpha means a greater preference for low risk, or a higher
% preference for more consumption over the long run.

alpha = 0.8;      % elasticity of utility parameter
beta = 0.1;       % discount factor

tp = 0.5;     % time pause between plot iterations

% % Construct state and action spaces
N = 1000;              % grid size
k = linspace(0,1,N)';     % cake fractions

figure('units','normalized','outerposition',[0 0 0.75 1]) % make plot window open to X-fraction
of screen

% Construct reward function (f)
f = utilityFn(tril(k*ones(1, N)-ones(N, 1)*k'), alpha); % using function utilityFn.m

% Construct state transition function (g)
g = zeros(N ,N);
for s=1:N
   g(s, :) = min(s, (1:N));
end

% Pack model structure
clear model
model.reward = f;
model.transfunc = g;
model.discount = beta;
if (finite)
   model.horizon = T;
end

% Solve model using the function "ddpsolve"
[v, index] = ddpsolve(model);

% Compute optimal cake and consumption path
k1 = k(index);
c = k*ones(1, size(index, 2)) - k1;

c_star = zeros(T, 1);
k_star = zeros(T+1, 1);
k_star(1) = 1;

for t=1:T
   s = getindex(k_star(t), k);
   if (finite)
      k_star(t+1) = k1(s, t);
      c_star(t) = c(s, t);
```

```
    else
        k_star(t+1) = k1(s);
        c_star(t) = c(s);
    end
end

%figure(1);
subplot(1, 1, 1);
plot((1:T), c_star);
axis([1 T 0 1]);
title( {...
    '{\bf\fontsize{20} Optimal Consumption Path}';...
    '{\it \Delta\alpha and \Delta\beta to evaluate changes in consumption preferences}'...
    }, 'FontSize', 14, 'FontWeight', 'normal')
xlabel('time', 'FontSize', 18);
ylabel('c^*_t', 'FontSize', 18);
legend({['c^*_t  \alpha =' num2str(alpha), ', \beta =' num2str(beta)]}, 'FontSize', 18, 'Location', 
'North', 'Box','off');
```

Matlab Code - Script 2:

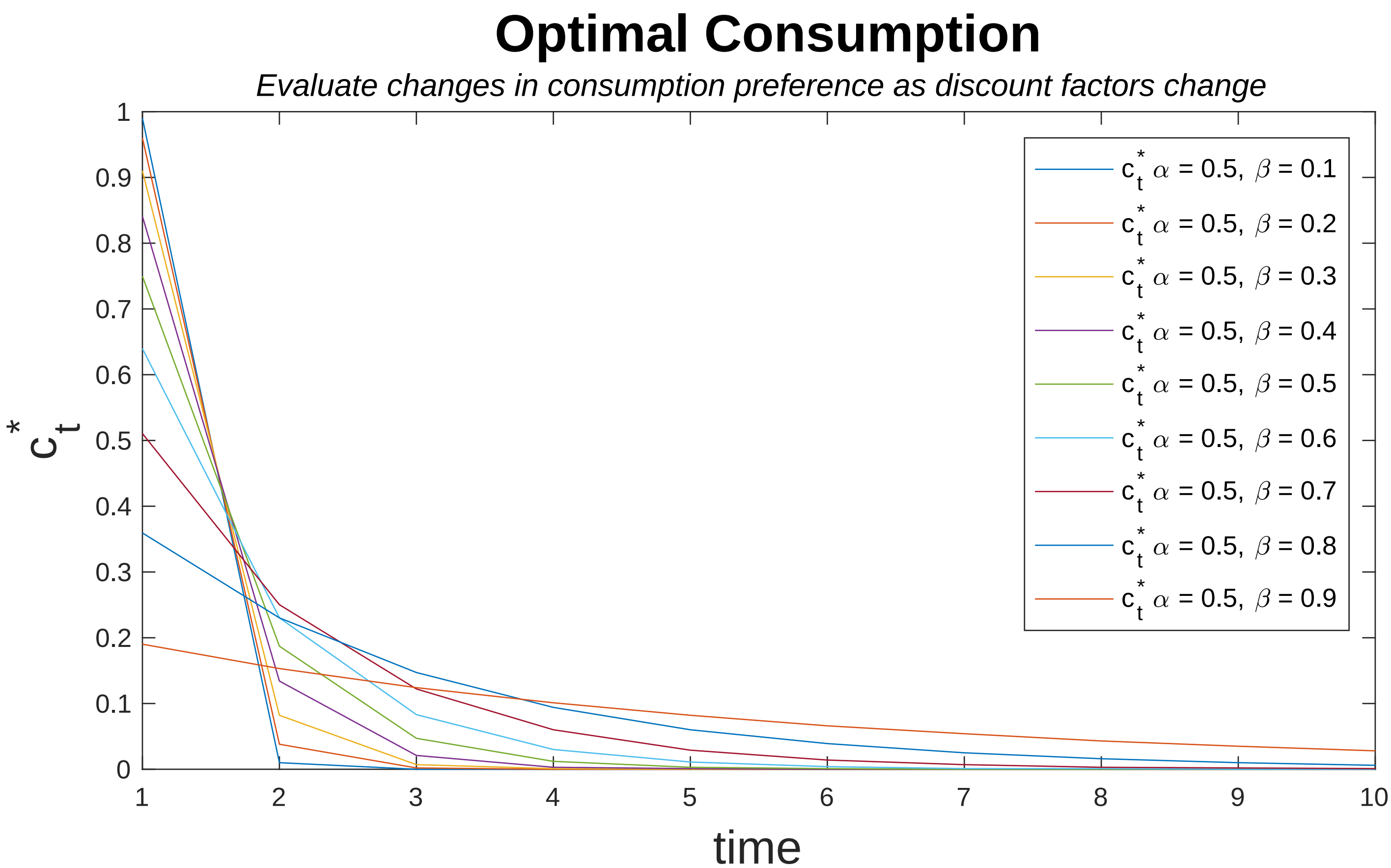
```
% cakeEating_iterPlots.m
% need utilityFn.m file, a function to define the utility function

% References:

% Adda, J., & Cooper, R. W. (2003). Dynamic economics : quantitative
% methods and applications. (R. W. Cooper, Ed.). Cambridge, Mass. : MIT Press.

% Miranda, M. J., & Fackler, P. L. (2002). Applied computational economics
% and finance. (P. L. Fackler, Ed.). Cambridge, Mass. : MIT Press.

% Scripts adapted from Dr. Chenggang Wang and Ming Kang

% housekeeping
close all
clear
clc
warning ('off')
optset('ddpsolve', 'prtiters', 0)
fprintf('\nCake Eating MODEL\n')
fprintf('\nddpsolve iteration output supressed, comment out optset to view\n')

% Enter model parameters
finite = 0;     % finite horizon 0/1
T = 10;         % Time horizon

% A discount factor, beta, which indicates the agents measure of time
% preference, with a higher beta meaning valuing future more and thus
```

```
% being more patient.

% Elasticity of Utility, alpha, a measure of risk preference. A lower
% value of alpha means a greater preference for low risk, or a higher
% preference for more consumption over the long run.

alpha = linspace(0.1, 0.9, 9);      % elasticity of utility parameter
beta = linspace(0.1, 0.9, 9);       % discount factor
tp = 0.5;     % time pause between plot iterations

% Construct state and action spaces
N = 1000;        % grid size
k = linspace(0, 1 ,N)';       % cake fractions

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% changing beta, holding alpha constant

figure('units','normalized','outerposition',[0 0 0.75 1]) % make plot window open to X-fraction
of screen
figure(1); hold on % allow plot build-up

for j = 1:numel(beta)

% Construct reward function (f)
f = utilityFn(tril(k*ones(1, N)-ones(N, 1)*k'), alpha(5)); % using function utilityFn.m

% Construct state transition function (g)
g = zeros(N, N);
for s=1:N
   g(s, :) = min(s, (1:N));
end

% Pack model structure
clear model
model.reward = f;
model.transfunc = g;
model.discount = beta(j);
if (finite)
   model.horizon = T;
end

% Solve model using the function "ddpsolve"
[v, index] = ddpsolve(model);

% Compute optimal cake and consumption path
k1 = k(index);
c = k*ones(1, size(index, 2)) - k1;

c_star = zeros(T, 1);
k_star = zeros(T+1, 1);
k_star(1) = 1;

for t=1:T
```

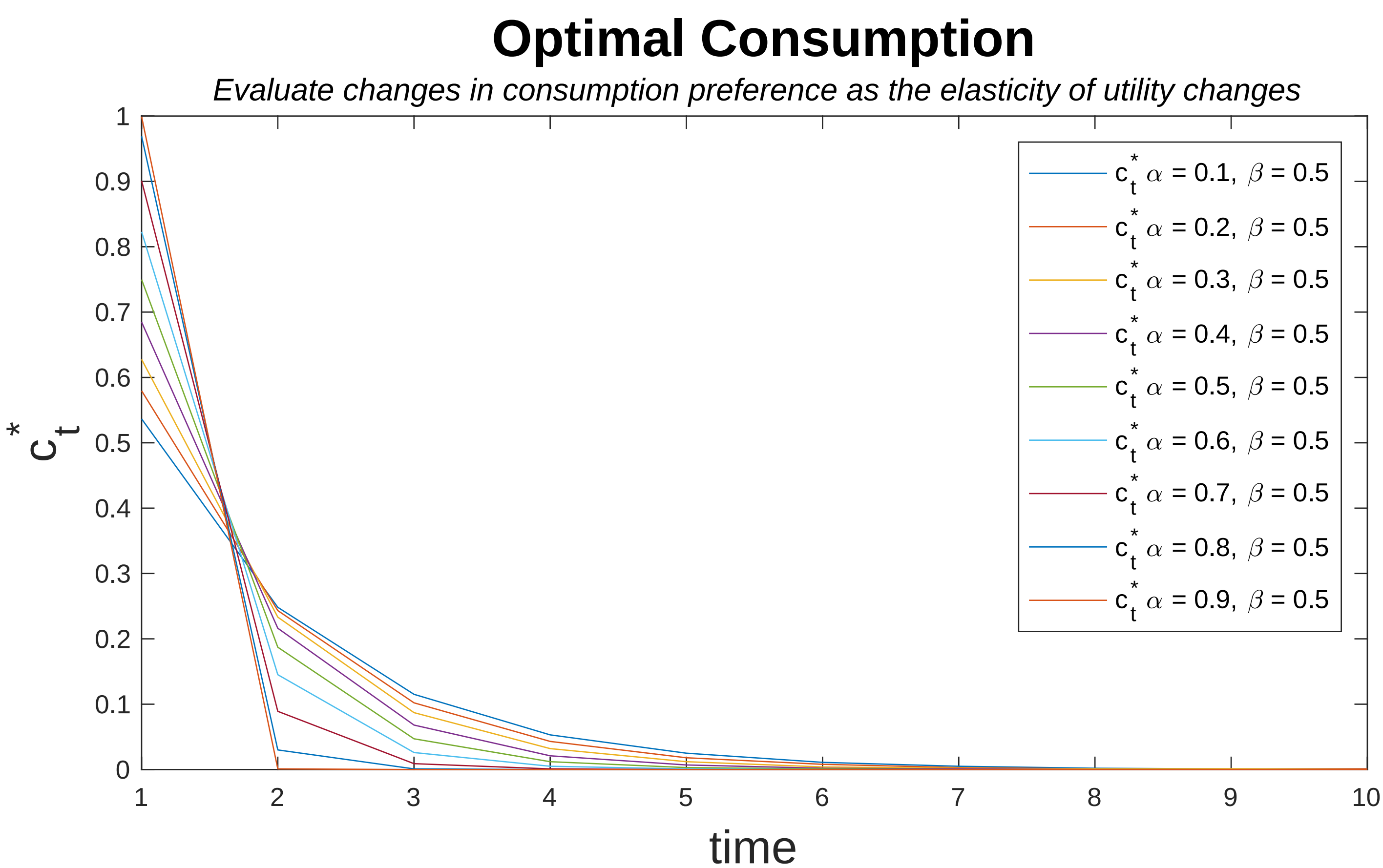
```
    s = getindex(k_star(t), k);
    if (finite)
        k_star(t+1) = k1(s, t);
        c_star(t) = c(s, t);
    else
        k_star(t+1) = k1(s);
        c_star(t) = c(s);
    end
end

%Plot of consumption path vs. change in discount rate

%figure(1);
subplot(2, 1, 1);
plot((1:T), c_star);
axis([1 T 0 1]);
title( {...
    '{\bf\fontsize{20} Optimal Consumption}';...
    '{\it Evaluate changes in consumption preference as discount factors change}'...
    }, 'FontSize', 12, 'FontWeight', 'normal')
xlabel('time', 'FontSize', 18);
ylabel('c^*_t', 'FontSize', 18);
legend({...
    'c^*_t \alpha = 0.5, \beta = 0.1',...
    'c^*_t \alpha = 0.5, \beta = 0.2',...
    'c^*_t \alpha = 0.5, \beta = 0.3',...
    'c^*_t \alpha = 0.5, \beta = 0.4',...
    'c^*_t \alpha = 0.5, \beta = 0.5',...
    'c^*_t \alpha = 0.5, \beta = 0.6',...
    'c^*_t \alpha = 0.5, \beta = 0.7',...
    'c^*_t \alpha = 0.5, \beta = 0.8',...
    'c^*_t \alpha = 0.5, \beta = 0.9'},...
    'FontSize', 10,'Location','Northeast','Box','on');
hold on;
pause(tp)

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% changing Alpha, holding beta constant at 0.5

for i = 1:numel(alpha)

% Construct reward function
f = utilityFn(tril(k*ones(1, N)-ones(N, 1)*k'), alpha(i)); % using function utilityFn.m

% Construct transition function
g = zeros(N, N);

for s=1:N
    g(s, :) = min(s, (1:N));
end
```

```
% Pack model structure
clear model
model.reward = f;
model.transfunc = g;
model.discount = beta(5);
if (finite)
   model.horizon = T;
end

% Solve model
[v, index] = ddpsolve(model);

% Compute optimal cake and consumption path
k1 = k(index);
c = k*ones(1, size(index, 2)) - k1;

c_star = zeros(T,1);
k_star = zeros(T+1, 1);
k_star(1) = 1;

for t=1:T
   s = getindex(k_star(t), k);
   if (finite)
      k_star(t+1) = k1(s, t);
      c_star(t) = c(s, t);
   else
      k_star(t+1) = k1(s);
      c_star(t) = c(s);
   end
end

% Plot of consumption path vs. change in elasticity of utility
%textLeg(i,:) = ['c^*_t \alpha =' num2str(alpha(i)), ', \beta =' num2str(beta(5))];
%textLeg = [{'c^*_t \alpha = 0.1, \beta = 0.5', 'c^*_t \alpha = 0.2, \beta = 0.5'}];

%figure(2);
subplot(2, 1, 2);
plot((1:T), c_star);
axis([1 T 0 1]);
title( {...
   '{\bf\fontsize{20} Optimal Consumption}';...
   '{\it Evaluate changes in consumption preference as the elasticity of utility changes}'...
   }, 'FontSize', 12, 'FontWeight', 'normal')
xlabel('time', 'FontSize', 18);
ylabel('c^*_t', 'FontSize', 18);
%legend({textLeg(1,1), textLeg(1,2)}, 'FontSize', 10,'Location','North','Box','off');
legend({...
   'c^*_t \alpha = 0.1, \beta = 0.5',...
   'c^*_t \alpha = 0.2, \beta = 0.5'...
   'c^*_t \alpha = 0.3, \beta = 0.5'...
   'c^*_t \alpha = 0.4, \beta = 0.5'...
   'c^*_t \alpha = 0.5, \beta = 0.5'...
   'c^*_t \alpha = 0.6, \beta = 0.5'...
   'c^*_t \alpha = 0.7, \beta = 0.5'...
```

```
    'c^*_t \alpha = 0.8, \beta = 0.5'...
    'c^*_t \alpha = 0.9, \beta = 0.5'},...
    'FontSize', 10,'Location','Northeast','Box','on');
%legend({['c^*_t \alpha =' num2str(alpha), ', \beta =' num2str(beta) '\newline']}, 'FontSize', 10,
'Location', 'North', 'Box','off');
hold on;
pause(tp)

end

%print('Looney_cake_plot', '-dpdf', '-bestfit') % export plot to file
```



Optimal Consumption
Evaluate changes in consumption preference as discount factors change
c_t^*
time
c_t^* α = 0.5, β = 0.1
c_t^* α = 0.5, β = 0.2
c_t^* α = 0.5, β = 0.3
c_t^* α = 0.5, β = 0.4
c_t^* α = 0.5, β = 0.5
c_t^* α = 0.5, β = 0.6
c_t^* α = 0.5, β = 0.7
c_t^* α = 0.5, β = 0.8
c_t^* α = 0.5, β = 0.9



Optimal Consumption
Evaluate changes in consumption preference as the elasticity of utility changes
c_t^*
time
c_t^* α = 0.1, β = 0.5
c_t^* α = 0.2, β = 0.5
c_t^* α = 0.3, β = 0.5
c_t^* α = 0.4, β = 0.5
c_t^* α = 0.5, β = 0.5
c_t^* α = 0.6, β = 0.5
c_t^* α = 0.7, β = 0.5
c_t^* α = 0.8, β = 0.5
c_t^* α = 0.9, β = 0.5