Growth - Projection Method -- polynomial approx

Basic method

In[917]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most
```

Out[917]=
```
{2020, 4, 29, 22, 27}
```

Choose production function

In[918]:=
```
f[x_] = A x^α; fp[x_] = A α x^(α-1);
```

Choose utility function

In[919]:=
```
ut[x_] = x^(1+γ)/(1+γ); utp[x_] = x^γ;
```

Choose parameters; set A so that steady state is k=1.

In[920]:=
```
α = .25; γ = -2.; β = .95;
A = (1/β-1)/α;
```

This notebook transforms the Euler equation to make it more linear. I did this in the JET paper.

Define inverse utility function

In[922]:=
```
utpinv[lam_] = lam^(1/ϒ);
```

Define operator

In[923]:=
```
Opf[x_] = c[x] - utpinv[
     β utp[c[x+f[x]-c[x]]] (1+fp[x+f[x]-c[x]])
                          ];
```

Define range

In[924]:=
```
capmin = .25; capmax = 1.75; capbar = (capmin + capmax)/2;
caprng = capmax - capmin;
```

In[926]:=
```
css = f[1]
```

Out[926]=
```
0.210526
```

Choose polynomial degree

In[927]:=
```
Degx = 3;
```

Define approximation

In[928]:=
```
c[x_] = Sum[a[i] x^(i-1), {i, 1, Degx+1}];
```

Compute collocation grid

```
In[929]:= sol = NSolve[ChebyshevT[Degx+1,x]==0,x];
          zeroes = Table[Last[First[sol[[i]]]],{i,1,Degx+1}];
          pts = N[Table[capbar + zeroes[[i]] caprng/2,{i,1,Degx+1}]]
Out[931]= {0.30709, 0.712987, 1.28701, 1.69291}
```

Compute initial guess

```
In[932]:= Solve[{a1 + a2 .5 == .6 f[.5],a1+a2==f[1]},{a1,a2}]
Out[932]= {{a1 → 0.00191067, a2 → 0.208616}}

In[933]:= {ainit[1], ainit[2]} = {a1, a2} /. %[[1]]
Out[933]= {0.00191067, 0.208616}

In[934]:= f[1]
Out[934]= 0.210526

In[935]:= f[1]
Out[935]= 0.210526

In[936]:= ainit[1] = 0; ainit[2] = f[1];
          Do[ainit[i] = 0, {i, 3, Degx + 1}];
```

Collect variables and initial guesses

```
In[938]:= varsin = Table[{a[i], ainit[i]}, {i, 1, Degx + 1}];
          vars = Table[a[i], {i, 1, Degx + 1}];
```

Compute collocation equations

```
In[940]:= eqns = Table[Opf[pts[[i]]] == 0,{i,1,Degx+1}];
```

Solve

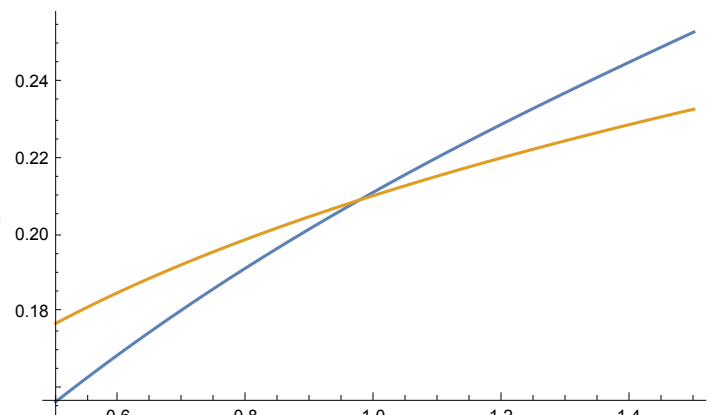
In[941]:=
```
sol = FindRoot[eqns,varsin]
```

Out[941]=
```
{a[1] → 0.0774433, a[2] → 0.18962, a[3] → -0.0700214, a[4] → 0.0144657}
```

Compute consumption function

In[942]:=
```
csol[x_] = Expand[c[x]/.sol];  oppol[x_] = Expand[Opf[x]/.sol];
Plot[{csol[x],f[x]},{x,.5,1.5}]
```

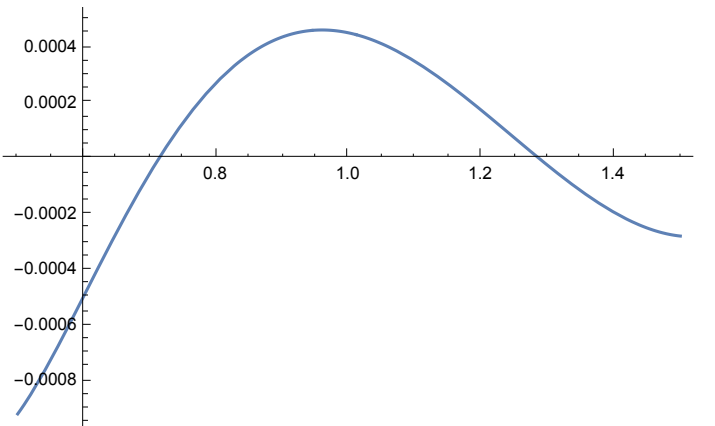
Out[943]=



In[944]:=
```
oppol[x_] = Expand[Opf[x] /. sol] / css;
```

In[945]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[945]=



Other examples

Define script project

In[946]:=
```
project := (f[x_] = A x^α; fp[x_] = A α x^(α-1);

ut[x_] = x^(1+γ)/(1+γ); utp[x_] = x^γ;

α = .25; γ = -2.; β = .95;
A = (1 / β - 1) / α;
utpinv[lam_] = lam^(1/γ);
Opf[x_] = c[x] - utpinv[
  β utp[c[x + f[x] - c[x]]] (1 + fp[x + f[x] - c[x]])
          ];
capmin = .25; capmax = 1.75; capbar = (capmin + capmax) / 2;
caprng = capmax - capmin;
css = f[1];
c[x_] = Sum[a[i] x^(i-1), {i, 1, Degx+1}];
sol = NSolve[ChebyshevT[Degx + 1, x] == 0, x];
zeroes = Table[Last[First[sol[[i]]]], {i, 1, Degx + 1}];
pts = N[Table[capbar + zeroes[[i]] caprng / 2, {i, 1, Degx + 1}]];
ainit[1] = 0; ainit[2] = f[1];
Do[ainit[i] = 0, {i, 3, Degx + 1}];
varsin = Table[{a[i], ainit[i]}, {i, 1, Degx + 1}];
vars = Table[a[i], {i, 1, Degx + 1}];
eqns = Table[Opf[pts[[i]]] == 0, {i, 1, Degx + 1}];
sol = FindRoot[eqns, varsin];
csol[x_] = Expand[c[x] /. sol]; oppol[x_] = Expand[Opf[x] /. sol];
oppol[x_] = Expand[Opf[x] /. sol] / css;)
```

Degree 3

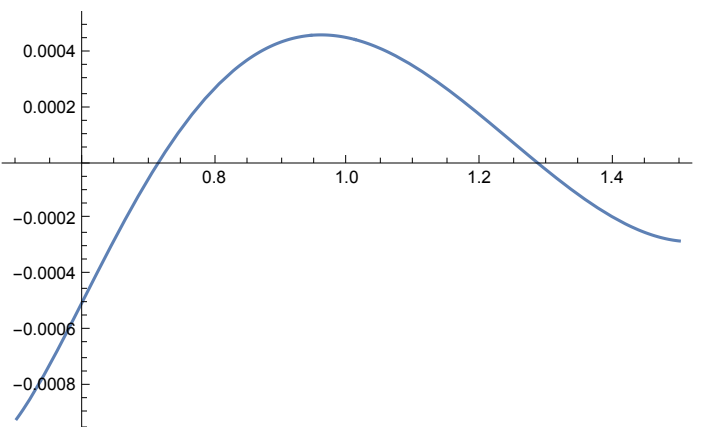
In[947]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most
```

Out[947]=
```
{2020, 4, 29, 22, 27}
```

In[948]:=
```
Degx = 3;
```

In[970]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[970]=



Degree 5

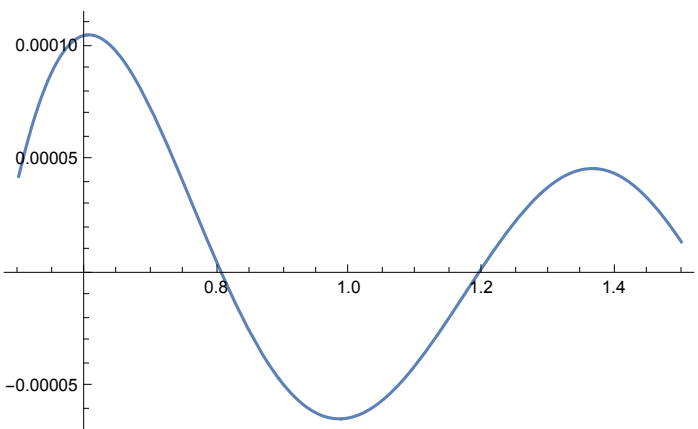
In[971]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most
```

Out[971]=
```
{2020, 4, 29, 22, 27}
```

In[972]:=
```
Degx = 5;
```

In[994]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[994]=



Degree 7

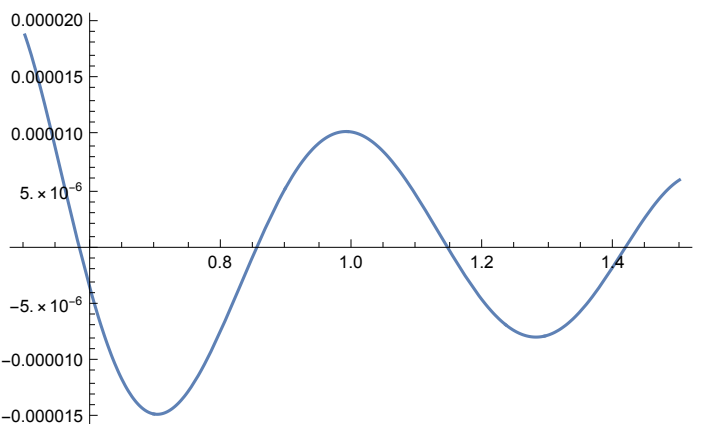
In[995]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most
```

Out[995]=
```
{2020, 4, 29, 22, 27}
```

In[996]:=
```
Degx = 7;
```

In[1018]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[1018]=



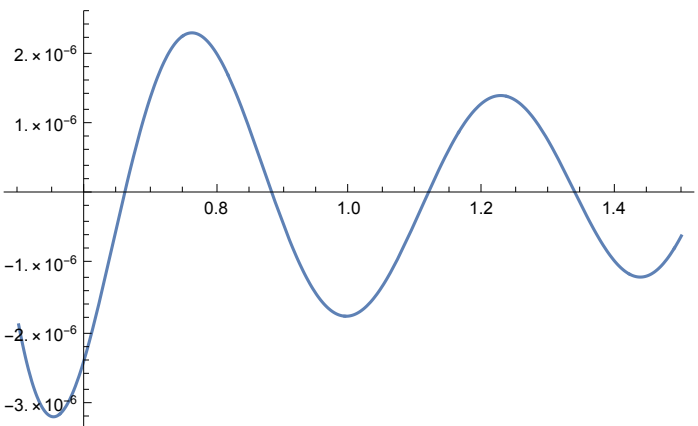
Degree 9

In[1019]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most;
```

In[1020]:=
```
Degx = 9;
```

In[1042]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[1042]=



Degree 11

In[1043]:=
```
x = 0; Remove["Global`*"]; DateList[Date[]] // Most;
```

In[1044]:=
```
Degx = 11;
```

In[1066]:=
```
Plot[oppol[x],{x,.5,1.5},PlotRange->All]
```

Out[1066]=