VFI Toolkit Workshop, pt4: OLG Models

vfitoolkit.com/2025-workshop-lse

Robert Kirkby

robertdkirkby.com
Victoria University of Wellington

VFI Toolkit

- We have seen Life-Cycle Models, which were partial equilibrium.
- Now for OLG models, which add general equilibrium.
- You can add general eqm to any of the models we have seen so far!

- Basic idea in the theory:
- General equilibrium involves choosing some parameters, like r...
- ...to sastify some equations, like $r = \alpha K^{\alpha-1} L^{1-\alpha}$.

VFI Toolkit

- Basic idea in VFI Toolkit:
- Give the names of the parameters to be determined in general eqm, e.g., $GEPriceParamNames = \{'r'\}$
- Setup *GeneralEqmEqns*, which must evaluate to zero in general eqm.
  E.g., instead of $r = \alpha K^{\alpha-1} L^{1-\alpha}$ we will write $r - \alpha K^{\alpha-1} L^{1-\alpha}$.
- We give the general eqm eqns names, e.g.,
  $GeneralEqmEqns.capitalmarket = @(r, K, L, alpha)\ r - \alpha K^{\alpha-1} L^{1-\alpha};$
- Inputs to *GeneralEqmEqns* can be anything in the parameter structure (*Params*) and any 'AggVars'.
  AggVars=the 'Mean' from AllStats ('AggVars' is a version of 'AllStats' that only computes the 'Mean')

- Solve stationary general eqm,
  [p_eqm,~,GEcondns]=HeteroAgentStationaryEqm_Case1_FHorz(...)

- Results:
- *p_eqm* contains GE param values, e.g., $p\_eqm.r = 0.05$;
- *GEcondns* contains values of GE conditions (should be all zeros).

- Internally VFI Toolkit uses optimization routines to choose *GEPriceParamNames* to minimize the sum-of-squares of the *GeneralEqmEqns*.

Basic OLG Model

- Let's solve a simple OLG model.
- Code: WorkshopOLGModel1.m (and WorkshopOLGModel1_ReturnFn.m)

Basic OLG Model

- Continum of mass one of households, solving 'household problem'.
- Continum of mass one of firms, solving 'firm problem'.
- Labor & Capital markets: labor & assets of household=labor & assets of firm.
- Assume perfectly competitive markets.

Basic OLG Model 1

- Household problem

$$
\begin{aligned}
V(a,z,j) \quad = \quad & \max_{h,c,aprime} \frac{c^{1-\sigma}}{1-\sigma} - \psi \frac{h^{1+\eta}}{1+\eta} + s_j \beta E[V(aprime, zprime, j+1)|z] \\
& \text{if } j < Jr: \; c + aprime = (1+r)a + w\kappa_j hexp(z) \\
& \text{if } j >= Jr: \; c + aprime = (1+r)a + pension \\
& 0 \leq h \leq 1, aprime \geq 0 \\
& z' = \rho_z z + \epsilon', \quad \epsilon \sim N(0, \sigma_{z,\epsilon})
\end{aligned}
$$

- Note, is just our WorkshopModel3 again!

Basic OLG Model 1

- Representative Firm problem

$$\max_{K^f,L^f} Y - rK^f - wL^f - \delta K^f$$
$$\text{s.t. } Y = (K^f)^\alpha (L^f)^{1-\alpha}$$

- Profit maximization (with price normalized to one), for firm with Cobb-Douglas production fn.
- Simplifies to two FOCs,

$$r = \alpha (K^f)^{\alpha-1}(L^f)^{1-\alpha} - \delta$$
$$w = (1-\alpha)(K^f)^{\alpha-1}(L^f)^{-\alpha}$$

- Trick: combine to get $w = (1-\alpha)\left(\frac{r+\delta}{\alpha}\right)^{\frac{\alpha}{\alpha-1}}$.
  This trick is standard and nothing to do with toolkit per se.

Our continuum of firms with constant-returns-to-scale production functions can just be replaced by a representative firm.

- Define aggregates, household labor supply and household assets

$$L^h = \int \kappa_j hexp(z) d\mu$$
$$K^h = \int a d\mu$$

- μ is the agent distribution of the households. Integral basically says 'add up across households'.

Basic OLG Model 1

- General eqm, is about $L^h = L^f$ and $K^h = K^f$. But we can substitute these into firm problem and instead get general eqm eqns,

$$r = \alpha (K^h)^{\alpha-1}(L^h)^{1-\alpha} - \delta$$

- and also the eqn we derived earlier $w = (1-\alpha)\left(\frac{r+\delta}{\alpha}\right)^{\frac{\alpha}{/}\alpha-1}$.

Basic OLG Model 1

- Definition of Stationary General Equilibrium,

Basic OLG Model 1

- Okay, let's write the code.
- Most is easy as is just copy-paste of WorkshopModel3.m where we already solved the household problem.
- One change here is *WorkshopModel3_ReturnFn*, which used to have w as input, but now remove this and instead use $w = (1-\alpha)\left(\frac{r+\delta}{\alpha}\right)^{\frac{\alpha}{\alpha-1}}$ inside the return fn.
  This is a standard trick and not about toolkit, we could just solve for r and w as general eqm parameters, and use both FOCs of firm problem as general eqm eqns, but would be slower. If you ever coded Aiyagari model you probably used this same trick so general eqm is only about r/K, and not also w/L.
  Also add alpha and delta as inputs to ReturnFn. Code also removes w from Params, and creates alpha and delta in Params.

- We need to add *FnsToEvaluate* for L^h (we already have one for K^h, although let's rename it to K).

```
%% Set up FnsToEvaluate
FnsToEvaluate.L=@(h,aprime,a,z,kappa_j) kappa_j*
    h*exp(z); % effective labor supply
FnsToEvaluate.K=@(h,aprime,a,z) a; % assets
```

Basic OLG Model 1

- Set up *GEPriceParamNames* as the names of parameters to find in general eqm, and *GeneralEqmEqns* as the equations that must evaluate to zero in general eqm.

```
%% General Eqm
GEPriceParamNames={'r'};
% note, Params.r we set earlier was an inital guess

GeneralEqmEqns.capitalmarket=@(r,alpha,delta,K,L) r-(alpha*(K^(alpha-1))*(L^(1-alpha))-delta);
% GeneralEqmEqn inputs must be either Params or AggVars (AggVars is like AllStats, but just the Mean)
% So here: r, alpha, delta will be taken from Params, and K,L will be taken from AggVars
```

Basic OLG Model 1

- Done setting up, now solve the Stationary General Equilibrium

```
%% Solve for stationary general eqm
heteroagentoptions.verbose=1; % just use
    defaults
[p_eqm,~,GEcondns]=
    HeteroAgentStationaryEqm_Case1_FHorz(
    jequaloneDist,AgeWeightParamNames, n_d, n_a,
    n_z, N_j, [], pi_z, d_grid, a_grid, z_grid,
    ReturnFn, FnsToEvaluate, GeneralEqmEqns,
    Params, DiscountFactorParamNames, [], [], [],
     GEPriceParamNames,heteroagentoptions,
    simoptions, vfoptions);
% Done, the general eqm prices are in p_eqm
% GEcondns tells us the values of the
    GeneralEqmEqns, should be near zero
```

- While solving, it repeatedly gives feedback

```
Current GE prices:
            r:    0.0533
Current aggregate variables:
            L:    0.6701
            K:    4.6979
Current GeneralEqmEqns:
            capitalmarket:   -0.0002
```

- Optimization: Given prices, solve V and Policy, solve StationaryDist, evaluate AggVars for the FnsToEvaluate, evaluate GeneralEqmEqns. Repeat with updated prices.
  Different optimization routines are just different ways to 'update prices'.

Basic OLG Model 1

- Analyzing the general eqm is easy.
- Just put general eqm price into Params: $Params.r = p\_eqm.r$
- Then just do V, $Policy$, $StationaryDist$, $AllStats$, $AgeConditionalStats$ like we did for life-cycle models.

- Done!
- Code: WorkshopOLGModel1.m (and WorkshopOLGModel1_ReturnFn.m)

OLG Models

- You can have n general eqm prices/parameters, for n general eqm eqns.
  They don't have to be 'prices', e.g., might choose gov. spending G to satisfy government budget constraint as one of our general eqm eqns.
- If you have a calibration target, like $K/Y = 3$, you can just set this up like another general eqm eqn.
  Only works if the calibration target can be expressed in terms of aggregates and parameters. (This is 'joint-optimization' of calibration and general eqm.)
- *heteroagentoptions.multiGEweights* can be used to give relative weights (default is equal weights).
  E.g. if you have lots of calibration targets you probably want to give them smaller weights so as not to interfere with the general eqm.

Basic OLG Model 2

- Let's do an example. OLG with government.
- We will add pensions, and have a tax that pays for them.
  General eqm eqn to balance the pension spending with the tax revenue.
- We will add progressive taxes, and use them to pay for (unmodelled) 'general government spending'.
  General eqm eqn to balance the general gov. spending with the revenue from progressive taxes.
- We will add a target for K/Y, and choose β to hit this target.
  General eqm eqn for this calibration target. And we will give it a lower weight.
- Code: WorkshopOLGModel2.m (and WorkshopOLGModel2_ReturnFn.m)
  We should use same trick for w, but I am lazy and formulas are all much easier to write if I have w, so add a General eqm eqn for the labor market too.

Some countries have separate budgets for pensions from rest of government, some have a single budget. Here we have separate budgets.

Basic OLG Model 2

- Household problem

$$
\begin{aligned}
&V(a,z,j) = \max_{h,c,aprime} \frac{c^{1-\sigma}}{1-\sigma} - \psi \frac{h^{1+\eta}}{1+\eta} + s_j \beta E[V(aprime, zprime, j+1)|z\\
&\text{if } j < Jr: \;\; earnings = w \kappa_j h exp(z)\\
&\text{if } j < Jr: \;\; income = r * a + earnings\\
&\text{if } j < Jr: \;\; earningstax = \tau_e \, earnings\\
&\text{if } j < Jr: \;\; incometax = income * (1 - \tau_{i1} income^{\tau_{i2}})\\
&\text{if } j < Jr: \;\; c + aprime = a + income - earningstax - incometax\\
&\text{if } j < Jr: \;\; income = r * a\\
&\text{if } j < Jr: \;\; incometax = income * (1 - \tau_{i1} income^{\tau_{i2}})\\
&\text{if } j >= Jr: \;\; c + aprime = a + income - incometax + pension\\
&0 \leq h \leq 1, aprime \geq 0\\
&z' = \rho_z z + \epsilon', \quad \epsilon \sim N(0, \sigma_{z,\epsilon})
\end{aligned}
$$

- Progressive income tax. Flat earnings tax. Pension.

Basic OLG Model 2

- Main changes are more FnsToEvaluate

```
FnsToEvaluate.L=@(h,aprime,a,z,kappa_j) kappa_j*h*exp(z); % effective labor supply
FnsToEvaluate.K=@(h,aprime,a,z) a; % assets
FnsToEvaluate.earningstaxrevenue=@(h,aprime,a,z,w,kappa_j,tau_e) tau_e*w*kappa_j*h*exp(z); % tau_e*earnings
FnsToEvaluate.incometaxrevenue=@(h,aprime,a,z,r,w,kappa_j,tau_i1,tau_i2) (r*a+w*kappa_j*h*exp(z))*(1-tau_i1*(r*a+w*kappa_j*h*exp(z))^tau_i2); % income*(1-tau_i1*income^tau_i2)
FnsToEvaluate.pensionspending=@(h,aprime,a,z,pension,agej,Jr) pension*(agej>=Jr); % pension is received by retirees
```

Basic OLG Model 2

- And more GeneralEqmEqns.

```
%% General Eqm
GEPriceParamNames={'w','r','pension','G','beta'};

GeneralEqmEqns.labormarket=@(w,alpha,K,L) w-(1-alpha)*(K^alpha)*(L^(-alpha));
GeneralEqmEqns.capitalmarket=@(r,alpha,delta,K,L) r-(alpha*(K^(alpha-1))*(L^(1-alpha))-delta);
GeneralEqmEqns.pensionbalance=@(earningstaxrevenue,pensionspending) earningstaxrevenue-pensionspending;
GeneralEqmEqns.govbudgetbalance=@(G, incometaxrevenue) G-incometaxrevenue;
GeneralEqmEqns.KdivYtarget=@(alpha,K,L,KdivYcalibtarget) K/((K^(alpha))*(L^(1-alpha)))-KdivYcalibtarget;
```

Basic OLG Model 2

- Done.
- You can watch it converge, which is helpful for seeing what is going wrong it if does not solve.
  E.g., you can see which GE condition is failing to get to zero. Or that some AggVar goes negative. Useful for debugging.
- Always check *GEcondns* to make sure that it did solve correctly.
  Solving one GE you can just watch it converge on screen, but sometime, e.g., you want to write a code to solve lots of models, in which case make sure you save GEcondns so you can check it.
- A little later we will see options to be 'faster' or 'slower but more robust'.

Basic OLG Model 3

- *heteroagentoptions.GEptype* can be used to name general eqm eqns to evaluate conditional on permanent type.
- Let's do an example. OLG with permanent types for five 'earnings quintiles'.
- We will include (accidental) bequests. Using a 'warm glow of bequests'.
  Warm glow is implemented via the ReturnFn
- Bequest received will be 'conditional on permanent type'.
  Bequests left=Bequests received, will be a general equilibrium. And we will solve it conditional on permanent type.
  Note: All the other general eqm eqns will be solved as standard (not conditioning on permanent type).
- As always, for permanent types use the *_PType* commands.
- Code: WorkshopOLGModel3.m (and WorkshopOLGModel3_ReturnFn.m)

- Household problem

$$
\begin{aligned}
V(a,z,j;i) \quad = \quad & \max_{h,c,aprime} \frac{c^{1-\sigma}}{1-\sigma} - \psi \frac{h^{1+\eta}}{1+\eta} + s_j \beta E[V(aprime, zprime, j+1; i) \\
& \text{if } j < Jr: \; c + aprime = (1+r)a + w\kappa_j\gamma_i hexp(z) \\
& \qquad\qquad +Beq * (Jbeq1 \leq agej \leq Jbeq2) \\
& \text{if } j >= Jr: \; c + aprime = (1+r)a + pension \\
& 0 \leq h \leq 1, aprime \geq 0 \\
& z' = \rho_z z + \epsilon', \quad \epsilon \sim N(0, \sigma_{z,\epsilon})
\end{aligned}
$$

- Fixed-effect in earings, γ_i. Bequests received, Beq_i.
- So permanent types. And bequests differ by permanent type.
- Bequests go to middle-aged (ages 40 to 60).

OLG Models: Further Options

- By default, it is using $heteroagentoptions.fminalgo = 1$, which is Matlab's $fminsearch()$.
- You can set different optimization routines (like *lsqnonlin*, $CMA - ES$, and *shooting*) just by changing $heteroagentoptions.fminalgo$ (=8,4,5 respectively).
  lsqnonlin requires Matlab Optimization Toolbox and implements Levenberg-Marquardt, it can be very fast. $CMA - ES$ is Covariance-Matrix Addaptation Evolutionary Strategy, it is slow but very robust. *shooting* is very very fast, but requires substantially more user input about how to update. See Appendix to Intro to OLG Models.
- Control solution tolerance using $heteroagentoptions$.
  heteroagentoptions.toleranceGEprices=10^(-4); % Accuracy of general eqm prices
  heteroagentoptions.toleranceGEcondns=10^(-4); % Accuracy of general eqm eqns
- You can use two algorithms, one after the other, with different tolerance. Can help get good mix of speed (first algo) and robust accuracy (second algo).
  E.g., set heteroagentoptions.fminalgo=[8,1] and heteroagentoptions.toleranceGEcondns=[10^(-3),10^(-5)]
- $lsqnonlin()$, $heteroagentoptions.fminalgo = 4$ is currently my preferred option.
  Is fast, although can be a bit of a rough solution, so I often then clean it up with $fminsearch()$ by setting $heteroagentoptions.fminalgo = [4, 1]$.

OLG Models: Further Options

- Evaluate (rather than solve for) GeneralEqmEqns
- Set *heteroagentoptions.maxiter* $= 0$ and then call *HeteroAgentStationaryEqm_Case1_FHorz()*
- Will be based on current contents of Params (for the values of the *GEPriceParamNames*).

OLG Models

- Anything we saw with Life-Cycle Models, can be done with OLG models!
- So Epstein-Zin preferences, Human Capital, etc.

- Everything we have seen so far in this workshop is covered in the examples of the Intro to OLG Models.
  Intro to OLG Models

References I

Robert Kirkby. VFI toolkit, v2. *Zenodo*, 2022. doi: https://doi.org/10.5281/zenodo.8136790.