VFI Toolkit Workshop, pt3C: Calibration and GMM Estimation

vfitoolkit.com/2025-workshop-lse

Robert Kirkby

robertdkirkby.com
Victoria University of Wellington

VFI Toolkit

- We have seen how to set up, solve and analyse Life-Cycle Models.
- In practice, it is very important what parameter values we use. Need a way to choose them.

  If you use arbitrary parameters, it is maths not economics.
- VFI Toolkit has tools to help: calibration and GMM estimation.

Life-Cycle Model

- So let's talk about..
- Calibration.
- GMM Estimation.

These will work in combination with everything we have seen so far.

Calibrating a Life-Cycle Model

- Basic Idea: Choose parameters, θ, so that some model statistics, M^m, match data statistics, M^d.

$$min_{\theta \in \Theta} \; d(M^m(\theta), M^d)$$

where $d()$ is a distance metric.

- We need to decide:
  1. Which parameters?
  2. Which model statistics?
  3. How to match?
- Let's do an example and discuss the options.

Calibrating a Life-Cycle Model

- Example: WorkshopModel17 (and WorkshopModel17_ReturnFn)
- Idea: solve Workshop Model 3, get life-cycle profile of mean assets, use this is as target to calibrate β and σ.
  Because we used model the generate 'data targets' we know what the solution should be.
  Obviously in practice your target will come from data.

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

Calibrating a Life-Cycle Model

1. Which parameters?

```
CalibParamNames={'beta','sigma'};
```

- Just use their names.
- Can include parameters that depend on age, and/or on permanent type.
- Can include parameters that parametrize the initial distribution, or parametrize the idiosyncratic shocks, or parametrize the permanent types.

Calibrating a Life-Cycle Model

2. Which model statistics?

```
1 TargetMoments.AgeConditionalStats.assets.Mean=
      AAA;
```

'AAA' is just in place of the actual numbers, you don't actually write AAA.

- Can target anything in 'AllStats' or in 'AgeConditionalStats'
  Including moments that depend on age.
  Notice that TargetMoments uses same naming as AllStats and AgeConditionalStats. So in practice, solve model, create AllStats and AgeConditionalStats, copy the naming into TargetMoments.
- Or you can target 'CustomModelStats'.

Will talk a bit later about which targets are 'better' or 'worse'. Here is just about which VFI Toolkit can handle.

Calibrating a Life-Cycle Model

3. How to match?

```
caliboptions=struct(); % use defaults
```

- Three aspects...
- Distance measure: sum of squares, sum of absolute difference, etc.

  caliboptions.metric $=$ *'sum_squared'* is the default.
- Weights: default is vector of ones.

  Set using *caliboptions.weights*
- Which optimization algorithm?

  Default is, *caliboptions.fminalgo* $= 8$, which uses Matlab *lsqnonlin*()

Calibrating a Life-Cycle Model

```
[CalibParams,calibsummary]=CalibrateLifeCycleModel(
    CalibParamNames,TargetMoments,...);
```

'...' are all inputs we have seen before (things we used in Workshop Model 3)

- And we're done!

One small change from Workshop Model 3: increased std dev of innovations to z, so as assets will be more sensitive to σ (otherwise struggles to estimate σ as model had very little risk, so risk aversion parameter didn't do much).

Calibrating a Life-Cycle Model

- Can also target 'conditionalrestriction' moments.
- There is also a 'CalibrateLifeCycleModel_PType()' command.
- Can target both 'conditional on ptype' and 'across all ptype' moments.

The calibrate and estimate commands automatically use *simoptions.whichstats* as we saw in Part 2.

GMM Estimating a Life-Cycle Model

- When all the model stats we target are moments, we use 'sum of squares' as the distance metric, and we have a weighting matrix W, this is GMM estimation.

$$\min_{\theta \in \Theta} (M^m(\theta) - M^d)' W (M^m(\theta) - M^d)$$

- W is a weighting matrix (if you have m moments, W is m-by-m).
- Also need Ω, the variance-covariance matrix of the data moments.

  More precisely, need $\hat{\Omega}$, an estimate of Ω

- Two important differences from calibration:
  - Choice of W determines statistical efficiency of the estimator.
    Can use any matrix W that is postive semi-definite. 'efficient GMM' is $W = \Omega^{-1}$.
  - After we get parameter estimate $\hat{\theta}$, we will calculate the standard deviation/confidence intervals for $\hat{\theta}$.

The above formulation of GMM assumes that the moments are 'separable'. Otherwise we would normally write $\min_{\theta \in \Theta} \; g(\theta, data)' W g(\theta, data)$, where $g()$ is a moment condition.

GMM Estimating a Life-Cycle Model

- For GMM, you first calculate M^d and $\hat{\Omega}$ from the data.
- Since since is unrelated to VFI Toolkit, I will not discuss it here.

  The 'life cycle model' is irrelvant to how you do this. I suggest you first look at how to GMM estimate the two parameters of a normal dist to understand concepts in a simple setting, then build from there.
- Once you have this, and you pick a W, VFI Toolkit does all the rest.

GMM Estimating a Life-Cycle Model

- Example: WorkshopModel18 (and reuse WorkshopModel17_ReturnFn)
- Idea: solve Workshop Model 3, get life-cycle profile of mean assets, use this is as target to GMM estimate β and σ. Use $W = \mathbb{I}$, and make up $\hat{\Omega}$.

  Because we used model the generate 'data targets' we know what the solution should be.
  Obviously in practice your target will come from data.

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

GMM Estimating a Life-Cycle Model

- Most is same as for calibration (except different names).
- Need to define weighting matrix W,

```
WeightingMatrix=eye(length(TargetMoments.
    AgeConditionalStats.assets.Mean));
```

Identity matrix (not a good choice in practice as scale dependent)

- And Covar-Matrix of data moments, $\hat{\Omega}$

```
CovarMatrixDataMoments=diag([linspace
    (0.01,10,20),linspace(10,1,25),linspace
    (1,0.01,N_j-45)]);
```

Covar-Matrix of data moments is here just made up numbers, it should be estimated from data.

- And done...

```
[EstimParams,EstimParamsConfInts,estsummary]=
    EstimateLifeCycleModel_MethodOfMoments(...);
```

GMM Estimating a Life-Cycle Model

- GMM Results
- Reports estimated parameters, and their 90% confidence intervals.
- Reports *estsummary* containing:
  - Standard deviation of estimated parameters.
  - Other confidence intervals: 80%, 95%, etc.
  - Identification: if model is (locally) identified.
  - Sensitivity (of estimated parameters to the target moments).
  - Various intermediate results (W, J, etc.)
- GMM options:
  - Sensitivity of estimated parameters to the pre-calibrated parameters.
  - Target log of moments.
  - Constrain parameters (to be positive, to be 0 to 1, to be A to B).

GMM Estimating a Life-Cycle Model

- The estimated parameter values are:
  $argmin_{\theta \in \Theta} (M^m(\theta) - M^d)' W (M^m(\theta) - M^d)$
- How are the standard deviation and confidence intervals determined?
- Standard deviation of $\hat{\theta}$ is given by 'sqrt of diagonals' of
  $\widehat{\sigma(\hat{\theta})^2} = (J^T W J)^{-1} J^T W \hat{\Omega} W^T J (J^T W J)$.
- Confidence intervals are then based on asymptotic theory which tells us that $\sqrt{N}(\hat{\theta} - \theta_0) \to_d N(0, \sigma(\theta)^2)$. So just confidence intervals of a normal distribution.

Reports all of these, but with emphasis on confidience intervals rather than standard deviations as in structural estimation you typically don't care about 'beta being statistically significantly different from zero'. And because confindence intervals are better practice and encourage you to avoid the 'star wars'. (It's my toolkit and I'll do what I want too ;)

What this means in practice for the internal codes is: 1. solve for the minimization problem to get $\hat{\theta}$, 2. calculate Jacobian matrix (J) of derivatives of moments w.r.t. θ, 3. evaluate $(J^T W J)^{-1} J^T W \hat{\Omega} W^T J (J^T W J)$ (note W and $\hat{\Omega}$ were both inputs), 4. calculate standard deviations and confidence intervals.

GMM Estimating a Life-Cycle Model

- Informal intuition on what determines standard deviation of estimated parameters:
- Estimated parameter will have small standard deviation if
  - Data moments have small standard deviation
    There is not much uncertainty to begin with.
  - 'Derivative of target moment with respect to parameter' is large
    If small changes in the parameter cause large changes in the target moment, then only small changes in the parameter are consistent with the plausible range for the target moment.

This is purely the intuition, we saw the exact equation on previous slide. Note that weights also matter, and that the targets are a vector and the derivatives are a (Jacobian) matrix.

GMM Estimating a Life-Cycle Model

- GMM is explained in Intro to Life-Cycle Models, Appendix C.
- GMM estimation examples in Life-Cycle Models 45-50.

  Covers: GMM estimation, permanent types, constraining parameters, parametrizing initial distribution of agents and idiosyncratic shock processes, how to do data work, how to choose W, how to use permanent types and unobserved heterogeneity, and more.
- This is GMM, not SMM (not simulated method of moments). See: Kirkby - 'Classical (not Simulated!) Method of Moment Estimation of Life-Cycle Models' for explanation.

Life-Cycle Model: Algorithms for estimation/calibration

- Optimization methods: set using *caliboptions.fminalgo* (and *estimoptions.fminalgo*)
- Default is *caliboptions.fminalgo* = 8, which uses Matlab *lsqnonlin*() (non-linear least squares, Levenberg-Marquart algorithm).
- Default is fast and good, but only medium presicion (hard to get accurate to more than 1e-3 or so).

  Also requires Matlab Optimization Toolbox.
- *caliboptions.fminalgo* = 4 is CMA-ES (Covariance Matrix Adaptation - Evolutionary Strategy). Slow, but very robust and accurate.
- Sometimes you might want to use one fast algo to get rough solution, then clean up with another.
- Or use a robust one the first time you run it while starting a new project to get decent initial guess. And from then on you comment that out and use it as an initial guess in a faster algo while you work with and refine/improve the model.
- Because of how VFI Toolkit operates, multi-grid optimization (first estimate with a small grid, then use this as initial guess to estimate with a big grid) is easy to implement.

Life-Cycle Models

- Calibration vs GMM Estimation.
- A lot of calibration is done as 'we use sigma=3 because James and Sarah (1990) did'.
- When doing the kind of calibration that 'CalibrateLifeCycleModel' command does, the difference from GMM is that you don't need 'Omega' (Covariance matrix of data moments) and that you don't get confidence intervals.
- Internally, the two share various codes (e.g., the code to solve the model and evaluate the objective fn).
- As a result, a lot of the options for how to set up are the same (just I call them *caliboptions* and *estimoptions*).

Life-Cycle Models

- Informal advice.
- I once heard/read Jesus Fernandez-Villaverde say he only estimates models he can solve in under 4 minutes.
  He was talking about all the Bayesian estimation of Rep agent DSGE he did with Juan Rubio Ramirez back in the day.
- If you can solve model in sub-2 minutes, good odds it is sub-24hrs to estimate.
- If you can solve model in 10 minutes, is likely to be all week to estimate.
- You should check model runtimes before deciding estimation is something you want to try. No rules, depends how much you like waiting one day/one week/one month.

Note: Something is going to go wrong, so you are not just going to do one estimation, you are going to do a few. That said, the first estimation is always the hardest, as after that you can use it as initial guess for all future estimations.

Life-Cycle Model

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

- Intro to Life-Cycle Models: pdf of 50 example Life-Cycle models, adding features one at a time. Covers everything we did here, plus much more.

References I

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