Classical (not Simulated!) Method of Moment Estimation of Life-Cycle Models

Robert Kirkby

Victoria University of Wellington
robertdkirkby.com
vfitoolkit.com

April 16, 2024

Before we start

- I want to establish some intuition to which I will return halfway through.
- Say I want to run a regression.
- I run it twice, once with 100 observations, once with 1,000,000 observations.
- How do you expect the standard errors of the parameter estimates to differ in these two cases?

Before we start

- Intuitively: everyone expects standard errors to be smaller when we have more data!

Background (1/4):

- Generalized Method of Moments, GMM.
- Simulated Method of Moments, SMM (or MSM).
- SMM is a standard way to estimate life-cycle models.

Background (2/4): What is GMM?

$$\min_\theta \, (M^d - M^m(\theta))W(M^d - M^m(\theta))$$

where θ is parameter vector, M^d are moments of the data, M^m are moments of the model, W is a weighting matrix.

- GMM: Statistical estimator that minimizes distance between data moments and model moments.
- E.g., Fit a normal distribution by matching mean and variance to data. Moments are (trivial) analytical expression of parameters.
- Requires a way to calculate moments directly from model.

Background (3/4): What is SMM?

$$\min_{\theta}\ (M^d - M^m(sim(\theta)))W(M^d - M^m(sim(\theta)))$$

where θ is parameter vector, M^d are moments of the data, M^m are moments of the simulated model data, *sim* is simulated model data, W is a weighting matrix.

- SMM: Statistical estimator that minimizes distance between data moments and model moments.
- Solve model based on θ, simulate model data, calculate model moments from the simulated data.
- Important difference: simulation adds *additional* source of randomness to the estimator.

Background (4/4): Why use SMM?

- "For some problems, the [model moments] may be difficult to express analytically or to compute, but relatively easy to simulate"
  — McFadden (1989), which introduced SMM estimation.

- Existing literature: estimate life-cycle models with SMM estimation.

Three bullet points

- This whole presentation can be understood in three bullet points.
  Everything else will just be details.

Three bullet points

1. GMM estimation is more statistically efficient than SMM estimation.
2. GMM estimation of life-cycle models is possible.
3. GMM estimation of one-endogenous state life-cycle models is faster.

Three bullet points

1. GMM estimation is more statistically efficient than SMM estimation.
2. GMM estimation of life-cycle models is possible.
3. GMM estimation of one-endogenous state life-cycle models is faster.

Implication: GMM is better than SMM for estimating (one-endogenous state) life-cycle models.

Three bullet points

1. GMM estimation is more statistically efficient than SMM estimation.

- Standard result, everyone knows this.
- Covariance matrix of SMM $= (1 + 1/s)$ *times* Covariance matrix of GMM.
  s is ratio of number of simulated data observations to actual data observations.
  Since $1 + 1/s > 1$, SMM has is less statistically efficient.
  Precise formula later.

Three bullet points

2. GMM estimation of life-cycle models is possible.
   - Widely-known in Macroeconomics.
     E.g., you can find it in Ríos-Rull (1995). Widely-known is not the same as widely-used.
   - Seemingly not widely known (or at least thought relevant) in Structural Estimation.
     E.g., not mentioned in Handbook of Labor Economics chapter of Keane, Todd, and Wolpin (2011), nor in survey of Low and Meghir (2017).
   - More precisely, I mean we know you can iterate agent distribution, then calculate moments from the agent distribution.
   - That is, we can calculate moments *without* simulation.

Three bullet points

3. GMM estimation of one-endogenous state life-cycle models is faster.
   - Iteration is a faster way to compute moments than simulation.
   - This was not true 10 years ago.
   - Now, Tan improvement (Tan, 2020) makes it very fast.
   - True of many kinds of moments (Ocampo and Robinson, 2023).

Three bullet points

1. GMM estimation is more statistically efficient than SMM estimation.
2. GMM estimation of life-cycle models is possible.
3. GMM estimation of one-endogenous state life-cycle models is faster.

   Implication: GMM is better than SMM for estimating (one-endogenous state) life-cycle models.

Example

- Estimate model of Gourinchas and Parker (2002).
- Life-cycle model with exogenous labor, essentially Carroll (1997) model.
- Target moments: mean consumption, conditional on age.
- Will explain steps in both GMM and SMM as we go.

Example: Setup

- Gourinchas and Parker (2002) uses SMM to estimate the life-cycle model,

$$
\begin{aligned}
V_j(a, z, e) \quad &= \quad \max_{a'} \upsilon_j \frac{c^{1-\rho}}{1-\rho} + \beta E[V_{j+1}(a', z', e')|z] \\
&\quad\quad \text{s.t. } c + a' = Ra + ze \\
&\quad\quad\quad a' \geq 0
\end{aligned}
$$

where a is assets, c is consumption, υ_j is a utility-shifter that depends on age j. Earnings follows an exogenous stochastic process comprised of a markov z and an i.i.d. e.

The terminal period value function is given by $V_{J+1} = \kappa \upsilon_{J+1} \frac{(X_{J+1}+H_{J+1})^{1-\rho}}{1-\rho}$, where $X_{J+1} = Ra_{J+1} + z_{J+1}$ and $H_{J+1} = hz_{J+1}$; by assumption $z_{J+1} = z_J$.

Model Detail

Example: Setup

- Agents in period $j = 1$ are distributed over assets as a log-normal distribution $log(a) \sim N(\omega_{26,mean}, \omega^2_{26,stddev})$.
- Some parameters are pre-calibrated: g_j, $\sigma_{z,n}$, σ_u, $\omega_{26,mean}$, and $\omega_{26,stddev}$.
  Essentially, exogenous shock parameters and initial age $j = 1$ distribution parameters.

Model Detail

Example: Setup

- The parameters β, ρ, h, and κ are estimated by classical method of moments (by SMM in Gourinchas and Parker (2002)).
  β is discount factor, ρ is CRRA utility parameter, h and κ parameterize V_{J+1}.
- The target data moments are the log of age-conditional mean of consumption (so $J = 40$ targets). The weighting matrix for the moments has the inverse of the empirical variance of each moment (calculated by bootstrap) on the diagonal, and zeros on the off-diagonal.

Example: Method of Moments

- The parameters β, ρ, h, and κ are estimated by classical method of moments (by SMM in Gourinchas and Parker (2002)).
- Classical Method of Moments,

$$\max_{\theta}[M_d - M_m(\theta)]' W [M_d - M_m(\theta)]$$

  W is weighting matrix

  M_d are data moments (in this example, mean consumption conditional on age, for 40 ages)

  $M_m(\theta)$ are model moments

- Simulated Method of Moments,

$$\max_{\theta}[M_d - M_{m,s}(\theta)]' W [M_d - M_{m,s}(\theta)]$$

  $M_{m,s}(\theta)$ are moments, calculated from model simulation

Example: Method of Moments

- Implementing code, for both GMM and SMM, outer part of code is an optimization.

Algorithm 1 Simulated/Classical Method of Moments Estimation: Optimization

1: Define M_d and W.
2: Guess initial θ.
3: Use optimization to solve $\min_\theta [M_d - M_{model}(\theta)]' W [M_d - M_{model}(\theta)]$

Unimportant detail: we use CMA-ES algorithm to solve the optimization.
M^d will be estimated from data. In principle can set any W, but is typically based on data.

Example: Method of Moments

- Difference between GMM and SMM is purely how to compute $M_{model}(\theta)$.

Example: Method of Moments

- SMM: simulate the life-cycle model, calculate moments from this.

Algorithm 2 Simulated Method of Moments Estimation: $M_{model}(\theta)$

Given θ.
Solve life-cycle model to get policy $g_j(a, z, e; \theta)$.
for $ii = 1 : sN$ **do**
    Simulate a J period life cycle
    Draw initial agent state (a_1, z_1, e_1) from initial agent distribution.
    Store $x_{i,1} = [a_1, z_1, e_1]$
    for $j = 2 : J$ **do**
        Draw (random) e_j from $\Pi_e(\cdot)$
        Draw (random) z_j from $\Pi_z(\cdot|z_{j-1})$
        $a_j = g_{j-1}(a_{j-1}, z_{j-1}, e_{j-1}; \theta)$
        Store $x_{i,j} = [a_j, z_j, e_j]$
    end for
    Store $x_i = [x_{i,1}, x_{i,2}, ..., x_{i,J}]$.
end for
Calculate moments $M_{model}(\theta) = \sum_{i=1}^{sN} f(x_i)$
return $M_{model}(\theta)$

Example: Method of Moments

- GMM: iterate agent distribution, calculate moments from this.

Example: Method of Moments

- Done first part. We have now estimated a parameter vector θ.
- For Gourinchas and Parker (2002) we get

Parameter	Value
β	0.9350
ρ	1.3186
h	0.3070
κ	38.785

- Next, standard errors.

GMM and SMM will lead to minorly different estimates of the parameter vector, but we do not look closely at this here.

Example: Method of Moments

- Theory for GMM and SMM is well established.
  Following is true if data is iid, time series (stochastic process), or markov.
- If the assumptions for GMM to be consistent and asymptotically normal are satisfied,

$$\sqrt{N}(\theta - \theta_0) \rightarrow N(0, \Sigma)$$

  where $\Sigma = (J'WJ)^{-1}J'W\Omega WJ(J'WJ)^{-1}$
  where W is the weighting matrix on the moments, J is the Jacobian matrix of derivatives of model moments with respect to θ ($J = \frac{\partial M_m}{\partial \theta}|_{\theta=\theta_0}$), and Ω is the covariance matrix of the data moments.
- If the assumptions for SMM to be consistent and asymptotically normal are satisfied,

$$\sqrt{N}(\theta - \theta_0) \rightarrow N(0, (1 + 1/s)\Sigma)$$

  where Σ is the same as for classical method of moments.
  s=# of obs in model simulation / # of data observations
- So variance of SMM is always $(1 + 1/s)$ times bigger.

Example: Method of Moments

- Difference between GMM and SMM is purely how to compute $M_{model}(\theta)$.
- SMM: *draw random* shocks to simulate model.
- This additional randomness leads to larger asymptotic variance.
  That SMM also leads to slightly different parameter vector is essentially just a symptom of this.
- We have now calculated the standard errors for our GP2002 GMM parameter estimates,

Parameter	Value	Std. Error
β	0.9350	0.0354
ρ	1.3186	0.7683
h	0.3070	3.9630
κ	38.785	84.546

Standard errors are essentially just Σ. We already have W, assume we have Ω (covar matrix of data moments), J is Jacobian matrix of derivatives of model moments with respect to estimated parameter vector and is computed by finite differences. Σ is jsut a combination of W, Ω and J.

Example: Method of Moments

- Story so far. Estimating GMM is possible, and more asymptotically statistically efficient.
- Remaining: GMM is also faster.
- Recall: GMM and SMM involve the same optimization, and same solving for policy function, so just compare 'iterate and calculate moments' versus 'simulate and calculate moments'.

Example: Method of Moments

- For Gourinchas and Parker (2002) we get the following runtimes.
- GMM runtime: 0.36 seconds
  0.29s to iterate agent dist, 0.07 to calculate moments
- SMM runtime: 0.52 seconds
  0.51s to simulate panel data, 0.02 to calculate moments
- SMM used 1000 individual simulations (40 periods each), which matches the 40,000 observations in data.
- GMM is faster, even with 'very small' simulation.
  Gourinchas and Parker (2002) used 20,000 simulations, Jorgensen (2023) used 500,000.

Note: Value function iteration took 4.06s.

Three bullet points

1. GMM estimation is more statistically efficient than SMM estimation.
2. GMM estimation of life-cycle models is possible.
3. GMM estimation of one-endogenous state life-cycle models is faster.

   Implication: GMM is better than SMM for estimating (one-endogenous state) life-cycle models.

GMM Extensions: Efficient GMM (1/2)

- Efficient GMM uses $W = \Omega^{-1}$.
- Standard GMM: $\sqrt(N)(\theta - \theta_0) \to N(0, \Sigma)$
  where $\Sigma = (J'WJ)^{-1}J'W\Omega WJ(J'WJ)^{-1}$
- Efficient GMM: Σ simplifies to $(J'\Omega^{-1}J)^{-1}$
- Obvious idea, use $W = \hat{\Omega}^{-1}$.
  E.g., estimate covar matrix of model moments by bootstrap to get $\hat{\Omega}$.

GMM Extensions: Efficient GMM (2/2)

- Obvious idea, use $W = \hat{\Omega}^{-1}$.
  E.g., estimate covar matrix of model moments by bootstrap to get $\hat{\Omega}$.
- Problem: in small samples $\hat{\Omega}$ is correlated with M^d
  Intuition: imagine the mean of a log-normal distribution; random draws that lead to high mean come from fat tail and also lead to large variance of the mean (Altonji and Segal, 1995).
- Correlation between W and M^d will cause biased estimates.

I am not aware of any evidence on what constitutes a 'small sample' for life-cycle models that would lead to this bias being quantitatively relevant.

GMM Extensions: Two-Iteration Efficient GMM

- Two-iteration efficient GMM aims to avoid the just described problem.
- Iteration 1: Use any W to GMM estimate $\hat{\theta}_1$.
  Consistent, just not efficient.
- Iteration 1.5: (Bootstrap) Simulate model under $\hat{\theta}_1$, compute $\widehat{\Omega(\hat{\theta}_1)}$.
  Standard advice is always use 'large' simulations.
- Iteration 2: Use $W = \widehat{\Omega(\hat{\theta}_1)}^{-1}$ to GMM estimate $\hat{\theta}_2$.

Flashback

- Intuitively: everyone expects standard errors to be smaller when we have more data!

GMM Extensions: Two-Iteration Efficient GMM, WTF?

- Intuitively: everyone expects standard errors to be smaller when we have more data!
- Two-Iteration GMM: standard errors are based on number of simulated observations, and typical advice is that these have nothing to do with number of data observations!!!
- Two-Iteration GMM: Standard errors fail the most basic intuition!

Two-Iteration GMM can be rescued if people insist on simulations using same number of observations as data. As they are currently done they deliberately/misleadingly give smaller standard errors (as bigger simulations). Even using same number of observations as data they eliminate all possibility of measurement error and other noise in data so probably still understate the uncertainty, but at least roughly right.

GMM Extensions: Bootstrap Standard Errors

- One way to get standard errors, calculate $\Sigma = (J'WJ)^{-1}J'W\Omega WJ(J'WJ)^{-1}$.
- Alternative: Bootstrap standard errors.
- Bootstrap 1: Bootstrap data to get B estimates of data moments, then use GMM to estimate parameter vector θ_b for each $b = 1, ..., B$.
- Bootstrap 2: After GMM estimation, re-estimate B times, using a different random number sequence for your SMM estimation.
  Normally, 'After SMM', but no reason we cannot do GMM as the first and main estimate, then use SMM for the bootstrap of standard errors.
  Obviously, my same concern about how many simulations you use here applies as for two-iteration GMM.
- In both, get B different estimates of θ, and these are the distribution of the parameter estimates.
  Confidence intervals, standard errors, etc., can then be trivally calculated from these.

GMM estimation of life-cycle models in VFI Toolkit

- Enough about GMM estimation of life-cycle models.
- What can VFI Toolkit do?

GMM estimation of life-cycle models in VFI Toolkit

- What can VFI Toolkit do?
- Essentially, any life-cycle model toolkit solves can potentially be estimated with GMM.
  Obviously, depends on runtimes.
- You tell it which moments and their values.
  Likely about 2-10 lines of trivial code.
- It does everything else.
  One line of code to run GMM.
- Internally, minimize the GMM objective function: for each parameter vector do value fn, agent dist, calc moments.
- So anything VFI Toolkit solves using 'ValueFnIter_Case1_FHorz' can be estimated, as long as runtime is usable.

GMM estimation of life-cycle models in VFI Toolkit

- In terms of runtime, this currently roughly means GMM estimation of any one-endogenous state life-cycle model.
- Includes everything in Intro to Life-Cycle Models
  - Markov and iid shocks.
  - Permanent types.
  - Semi-exogenous shocks (used for fertility and search-labor).
  - Human capital.
    'experienceasset', 'experienceassetu'
  - Portfolio-choice.
    'riskyasset'
  - Epstein-Zin preferences, Quasi-hyperbolic discounting, Gul-Pesendorfer preferences.

GMM estimation of life-cycle models in VFI Toolkit

- In terms of target moments, anything computed by 'AllStats' or 'LifeCycleProfiles'.
- So for any function of state/policy/parameters (e.g. wealth, consumption, etc.)
  - Mean, Median, Variance.
  - Lorenz curve, Gini.
  - Quantiles.
  - Aggregate, or age-conditional.
  - And more...

Intend to add correlations to possible targets at a future date.

GMM estimation of life-cycle models in VFI Toolkit

- Additional outputs:
  - Sensitivity of the parameter estimates to the estimation moments (Andrews, Gentzkow, and Shapiro, 2017).
  - (Optional) Sensitivity of the parameter estimates to pre-calibrated parameters (Jorgensen, 2023).

GMM estimation of life-cycle models in VFI Toolkit

- New contribution: before estimation, reports derivatives of 'all' moments with respect to parameters to be estimated.
- Intuition: sensitivity of parameter estimates to estimation moments is essentially increasing the magnitude of derivative, decreasing in variance of moment.
- So we might be interested in which moments have largest derivatives.
- E.g., Cagetti (2003) estimates same model as Gourinchas and Parker (2002) (which is the model of Carroll (1997)). Targets age-conditional median wealth, instead of age-conditional mean consumption.
- Can we tell beforehand, which might be 'better'?
  Which moments will give lower parameter uncertainty?

GMM estimation of life-cycle models in VFI Toolkit

- VFI Toolkit computes 'all' moments anyway, so a one-line command reports derivatives and ranks them by magnitude.
  One line: 'EstimateLifeCycleModel_MomentDerivatives'
- Do this based on calibration of model in Carroll (1997) for consumption and wealth.
- Find that magnitude of derivatives w.r.t. β are ~ 3 times larger for Wealth than for Consumption.
  Suggests –if standard error of wealth$<$ 3 times standard error of consumption– we will get smaller parameter uncertainty targeting wealth.
- Compare findings of Cagetti (2003) and Gourinchas and Parker (2002) for β, those of Cagetti (2003) have slightly smaller standard errors.
  Caveat: They get quite different estimates of ρ, and have some other minor differences.
- Is intended as rough guide to think about what data to collect to estimate a model.
- Also have to think about uncertainty of moments.
- Also, computed derivative at calibrated parameter values, but standard errors based on derivatives at estimated parameter values. These may differ.

Closing

- GMM estimation of life-cycle models is possible.
- GMM estimation of life-cycle models is more 'statistically efficient' than SMM.
- For one endogenous-state models, GMM also has faster code run-time than SMM.
  I expect this will soon include two-endogenous state models, if not already.
- Use GMM to estimate your life-cycle model! It is better than SMM in every aspect (statistically and computationally).

References I

Joseph Altonji and Lewis Segal. Small-sample bias in gmm estimation of covariance structures. *Journal of Business and Economics Statistics*, 14(3), 1995. doi: https://doi.org/10.1080/07350015.1996.10524661.

Isaiah Andrews, Matthew Gentzkow, and Jesse Shapiro. Measuring the sensitivity of parameter estimates to estimation moments. *Quarterly Journalk of Economics*, 132(4), 2017. doi: https://doi.org/10.1093/qje/qjx023.

Marco Cagetti. Wealth accumulation over the life cycle and precautionary savings. *Journal of Business and Economic Statistics*, 21(3), 2003. doi: https://doi.org/10.1198/073500103288619007.

Christopher Carroll. Buffer-stock saving and the life cycle/permanent income hypothesis. *The Quarterly Journal of Economics*, 112(1):1–55, 1997.

Pierre-Olivier Gourinchas and Jonathan Parker. Consumption over the life cycle. *Econometrica*, 70(1), 2002. doi: https://doi.org/10.1111/1468-0262.00269.

References II

Thomas Jorgensen. Sensitivity to calibrated parameters. *Review of Economics and Statistics*, 105(2), 2023. doi: https://doi.org/10.1162/rest_a_01054.

Michael Keane, Petra Todd, and Kenneth Wolpin. The structural estimation of behavioral models: Discrete choice dynamic programming methods and applications. In D. Card and O. Ashenfelter, editors, *Handbook of Labor Economics*, chapter 4. Elesevier, 2011.

Hamish Low and Costas Meghir. The use of structural models in econometrics. *Journal of Economic Perspectives*, 31(2), 2017. doi: https://doi.org/10.1257/jep.31.2.33.

McFadden. A method of simulated moments for estimation of discrete response models without numerical integration. *Econometrica*, 57(5), 1989. doi: https://doi.org/10.2307/1913621.

Sergio Ocampo and Baxter Robinson. Computing longitudinal moments for heterogeneous agent models. *Computational Economics*, 2023. doi: https://doi.org/10.1007/s10614-023-10493-1.

References III

José Víctor Ríos-Rull. Models with heterogenous agents. In T. Cooley, editor, *Frontiers in Business Cycle Research*, chapter 4. Princeton University Press, 1995. doi: https://doi.org/10.2307/j.ctv14163jx.10.

Eugene Tan. A fast and low computational memory algorithm for non-stochastic simulations in heterogeneous agent models. *Economics Letters*, 193, 2020. doi: https://doi.org/10.1016/j.econlet.2020.109285.

Gourinchas & Parker (2002), Details: Shocks

The (non-stationary) markov shocks follow a non-stationary unit-root with drift,

$$log(z_j) = g_j + log(z_{j-1}) + n$$
$$n \sim N(0, \sigma_{z,n})$$

The i.i.d shocks are assumed to follow

$$e \begin{cases} \text{with probability } p: & e = 0 \\ \text{with probability } 1-p: & u = log(e), u \sim N(0, \sigma_u) \end{cases}$$

Gourinchas and Parker (2002) do not explicitly include $a' \geq 0$ in their setup. But they follow Carroll (1997) who combines an assumption that agents must die with non-negative assets (that $a_{J+1} \geq 0$) with an income process that has a positive probability of taking a zero value in each period, and together these two assumptions imply that $a' \geq 0$ in every period.
The article of Gourinchas and Parker (2002) does not discuss how the initial values of z and e are set. The code of Jorgensen (2023) sets 'period 0' to the mean value of $log(Y_1)$, and then does one single draw of the permanent shock innovations to get the period 1 permanent shock distribution; and uses the standard transitory shock distribution. Then sets the agent distribution to the cross-product of the asset, permanent shock, and transitory shock distributions.
Model with unit-root shocks should be renormalized to eliminate them from state-space (so they are just a stochastic discount factor) when solving value function problem. I am lazy and just discretize them instead, so we are actually solving something more computationally difficult than necessary.

Back