Lecture 9: Simulation and Inference

Jacob Adenbaum

University of Edinburgh

Spring 2024

Using our models

- There are three main reasons why a solved model is useful:
  1. **Understanding the mechanisms:** sometimes, plotting the solutions or simulating data can be illuminating about how a model works
  2. **Estimating Parameters:** We want to choose parameters of the model in a sensible way so that our model matches the data
  3. **Counterfactual Exercises:** What happens if we change something in the model? What does it predict? How does it change welfare? Other metrics we care about?
- It turns out that to do all of this, we need to be able to generate fake (simulated) data from our models

Section 1

Simulating Data from Models

Simulating Data: Deterministic Case

- Consider the non-stochastic neoclassical growth model we saw last week:

$$\begin{aligned} v(k) = \max_{c,k'} \quad & u(c) + \beta v(k') \\ \text{s.t.} \quad & c + k' \leq Ak^{\alpha} + (1-\delta)k \end{aligned} \tag{1}$$

- A solution to this model will give us some representation of $v(k)$ as well as policy functions $g_c(k)$ and $g_k(k)$, where

$$v(k) = u(g_c(k)) + \beta v(g_k(k))$$

- Suppose we start at an arbitrary k_0.

- Since the model is deterministic, generating "data" from the model is just a matter of stepping our value k_0 forward one period at a time using our policy functions

$$k_t = g_k(k_{t-1})$$
$$c_t = g_c(k_t)$$

Simulating Data: Deterministic Case

- Consider the non-stochastic neoclassical growth model we saw last week:

$$\begin{aligned} v(k) = \max_{c,k'} \quad & u(c) + \beta v(k') \\ \text{s.t.} \quad & c + k' \leq Ak^{\alpha} + (1-\delta)k \end{aligned} \tag{1}$$

- A solution to this model will give us some representation of $v(k)$ as well as policy functions $g_c(k)$ and $g_k(k)$, where

$$v(k) = u(g_c(k)) + \beta v(g_k(k))$$

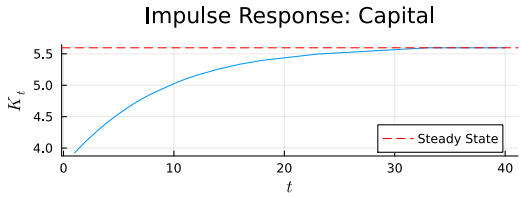
- Suppose we start at an arbitrary k_0.

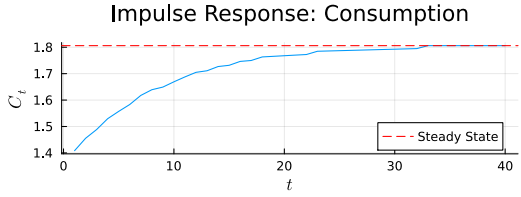
- Since the model is deterministic, generating "data" from the model is just a matter of stepping our value k_0 forward one period at a time using our policy functions

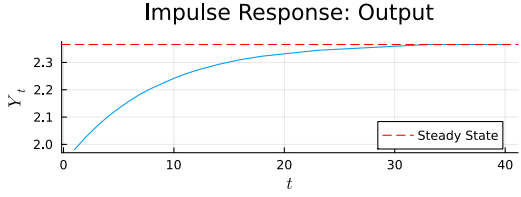
$$\begin{aligned} k_t &= g_k(k_{t-1}) \\ c_t &= g_c(k_t) \end{aligned}$$

Stepping the model forward

```
function simulate(p, policy, kgrid, ki0;
    T = 40
)
    @unpack A, α, δ = p
    C = zeros(T);        K = zeros(T)
    Ki= zeros(Int, T);   Y = zeros(T)

    Ki[1] = ki0
    for t in 1:T
        # Capital and Output
        K[t]   = kgrid[Ki[t]]
        Y[t]   = A * K[t]^α
        # Consumption today
        z      = Y[t] + (1-δ) * K[t]
        ki′    = policy[Ki[t]]
        k′     = kgrid[ki′]
        C[t]   = z - k′
        # Investment grid tomorrow
        if t < T
            Ki[t+1] = ki′
        end
    end
    return (; C, K, Ki, Y)
end
```







Non-deterministic case: Simulate data

- Suppose that we return to the case of our stochastic neoclassical growth model:

$$
\begin{aligned}
v(k, A) = \max_{c, k'} \quad & u(c) + \beta \mathbb{E}\left[v(k', A') | A\right] \\
\text{s.t.} \quad & c + k' \leq A k^{\alpha} + (1 - \delta) k \\
& \log(A') = \rho \log(A) + \epsilon \\
& \epsilon \sim N(0, \sigma)
\end{aligned}
\tag{2}
$$

- How can we simulate data from this model?
  1. Solve the model, and recover the policy rules $g_c(k, A)$ and $g_k(k, A)$
  2. Start from (k_0, A_0)
  3. Draw a random sequence of A_t that satisfy our process for A
  4. For each $t > 0$, set

$$
\begin{aligned}
k_t &= g_k(k_{t-1}, A_{t-1}) \\
c_t &= g_c(k_t, A_t)
\end{aligned}
$$

Non-deterministic case: Simulate data

- Suppose that we return to the case of our stochastic neoclassical growth model:

$$\begin{aligned} v(k, A) = \max_{c, k'} \quad & u(c) + \beta \mathbb{E}\left[v(k', A') | A\right] \\ \text{s.t.} \quad & c + k' \leq A k^{\alpha} + (1 - \delta) k \\ & \log(A') = \rho \log(A) + \epsilon \\ & \epsilon \sim N(0, \sigma) \end{aligned} \tag{2}$$

- How can we simulate data from this model?
  1. Solve the model, and recover the policy rules $g_c(k, A)$ and $g_k(k, A)$
  2. Start from (k_0, A_0)
  3. Draw a random sequence of A_t that satisfy our process for A
  4. For each $t > 0$, set

$$\begin{aligned} k_t &= g_k(k_{t-1}, A_{t-1}) \\ c_t &= g_c(k_t, A_t) \end{aligned}$$

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0,1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0,1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0,1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0,1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0,1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on $[0, 1]$
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on [0,1]
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- Suppose you have a cumulative distribution function $F_X : \mathcal{X} \subset \mathbb{R} \to [0, 1]$
- How can you generate data that is distributed according to F?
- Recall that if $X \sim F_X$ then $F_X(x) = Pr(X \leq x)$
- But note that $Z = F_X(x)$ is also a random variable. What is its distribution F_Z?
- Observe that since F_X is strictly increasing, it has an inverse F_X^{-1}. So,

$$F_Z(z) = Pr(Z \leq z) = Pr(F_X(X) \leq z) = Pr(X \leq F_X^{-1}(z)) = F_X(F_X^{-1}(z)) = z$$

- But this is *exactly* the cdf of the uniform distribution on [0,1]
- $F_X(X)$ is uniformly distributed!

Refresher: Drawing random variables

- This means that as long as we know how to take pseudo-random draws from a uniform distribution, we can draw random numbers from *any* distribution where we know the cdf

- Follow the steps:
  1. Draw $z \sim U[0, 1]$
  2. Find x such that $F_X(x) = z$

     This may require solving a root finding problem if you don't have a formula for F_X^{-1}
  3. Call x your random draw from F

- If there are no values of X that show up with probability 0, then F_X is strictly increasing and step 2 is always well-defined

- If there are values of X that occur with probability 0, then just pick x as the smallest x such that $F_X(x) = z$.

Refresher: Drawing random variables

- This means that as long as we know how to take pseudo-random draws from a uniform distribution, we can draw random numbers from *any* distribution where we know the cdf

- Follow the steps:
  1. Draw $z \sim U[0, 1]$
  2. Find x such that $F_X(x) = z$

     This may require solving a root finding problem if you don't have a formula for F_X^{-1}
  3. Call x your random draw from F

- If there are no values of X that show up with probability 0, then F_X is strictly increasing and step 2 is always well-defined

- If there are values of X that occur with probability 0, then just pick x as the smallest x such that $F_X(x) = z$.

Refresher: Drawing random variables

- This means that as long as we know how to take pseudo-random draws from a uniform distribution, we can draw random numbers from *any* distribution where we know the cdf

- Follow the steps:
  1. Draw $z \sim U[0, 1]$
  2. Find x such that $F_X(x) = z$

     This may require solving a root finding problem if you don't have a formula for F_X^{-1}
  3. Call x your random draw from F

- If there are no values of X that show up with probability 0, then F_X is strictly increasing and step 2 is always well-defined

- If there are values of X that occur with probability 0, then just pick x as the smallest x such that $F_X(x) = z$.

Refresher: Drawing random variables

Discrete variables

- This process is particularly straightforward if we have a discrete random variable

$$X \in \{X_1, X_2, \ldots, X_n\} \quad \text{where} \quad Pr(X = X_i) = p_i$$

  where $X_1 < X_2 < \cdots < X_n$

- Observe that we can write the cdf of X as a cumulative sum:

$$F_i := F(X_i) = Pr(X \leq X_i) = \sum_{j=1}^{i} Pr(X = X_j) = \sum_{j=1}^{i} p_j$$

- This means if we draw $z \sim U[0, 1]$, "inverting" F is just a matter of finding the smallest i such that $z \leq F_i$

- Since F_i are already a sorted vector, this is just searching through a sorted list (use `searchsortedfirst` in Julia)

Refresher: Drawing random variables

Discrete variables

- This process is particularly straightforward if we have a discrete random variable

$$X \in \{X_1, X_2, \ldots, X_n\} \quad \text{where} \quad Pr(X = X_i) = p_i$$

  where $X_1 < X_2 < \cdots < X_n$

- Observe that we can write the cdf of X as a cumulative sum:

$$F_i := F(X_i) = Pr(X \leq X_i) = \sum_{j=1}^{i} Pr(X = X_j) = \sum_{j=1}^{i} p_j$$

- This means if we draw $z \sim U[0, 1]$, "inverting" F is just a matter of finding the smallest i such that $z \leq F_i$

- Since F_i are already a sorted vector, this is just searching through a sorted list (use `searchsortedfirst` in Julia)

Refresher: Drawing random variables

Discrete variables

- This process is particularly straightforward if we have a discrete random variable

$$X \in \{X_1, X_2, \ldots, X_n\} \quad \text{where} \quad Pr(X = X_i) = p_i$$

  where $X_1 < X_2 < \cdots < X_n$

- Observe that we can write the cdf of X as a cumulative sum:

$$F_i := F(X_i) = Pr(X \leq X_i) = \sum_{j=1}^{i} Pr(X = X_j) = \sum_{j=1}^{i} p_j$$

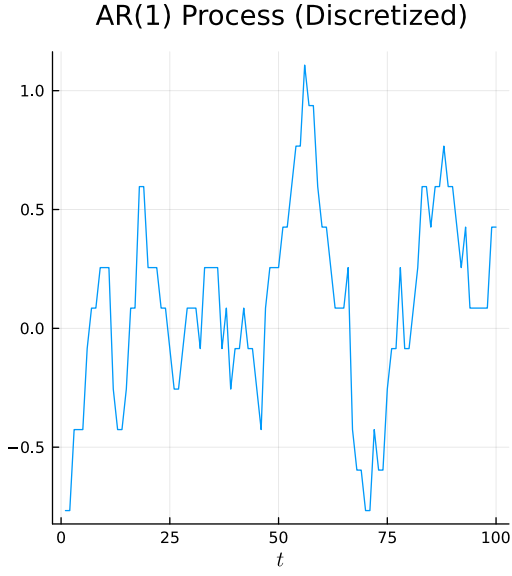
- This means if we draw $z \sim U[0, 1]$, "inverting" F is just a matter of finding the smallest i such that $z \leq F_i$

- Since F_i are already a sorted vector, this is just searching through a sorted list (use `searchsortedfirst` in Julia)

Refresher: Drawing random variables

Discretized AR(1)

```
## Simulate and plot and AR(1)
mc       = rouwenhorst(30, 0.9, 0.2)
grid     = mc.state_values
P        = mc.p'

function simulate_markov(grid, P; T = 100)
    n       = length(grid)
    i       = rand(1:n)
    X       = zeros(T)
    Pc      = cumsum(P, dims=1)
    for t in 1:T
        z       = rand()
        Fi      = @view Pc[:, i]
        i       = searchsortedfirst(Fi, z)
        X[t]    = grid[i]
    end
    return X
end
```



Stochastic Neoclassical Growth: Revisited

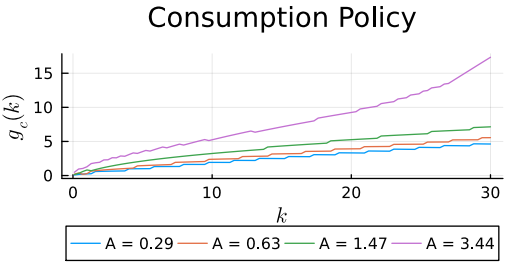
Outer Loop

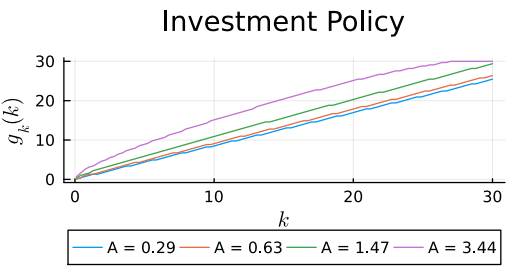
```
function solve_vfi_stochastic!(p, m; tol = 1e-12, maxiter = 1000)
    @unpack Na, ρ, σ                        = p
    @unpack kgrid, agrid, V0, V, P, policy = m
    V   .= V0
    iter = 0
    while true
        iter += 1
        # Step 1: Calculate Expectations
        EV = V * P

        # Step 2: Update Bellman Equation
        update_bellman_stochastic!(p, m, EV)

        # Step 3: Check for convergence
        ϵ = maximum(abs.(V - V0))
        ϵ < tol         && break
        iter >= maxiter && break
        V0 .= V
    end
    return (; V, policy, iter, kgrid, agrid, P)
end
```

Stochastic Neoclassical Growth: Revisited

Inner Loop

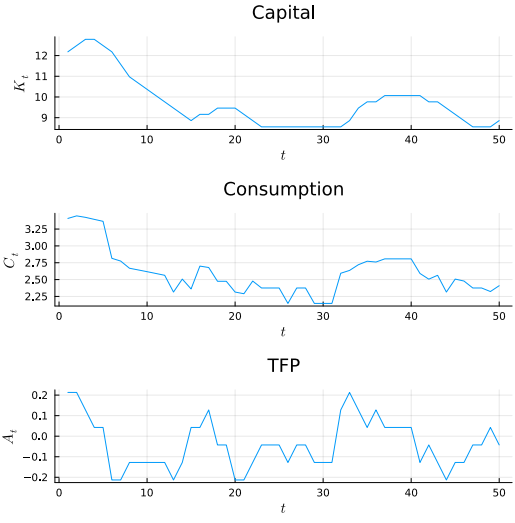
```
function update_bellman_stochastic!(p, m, EV)
    @unpack V, kgrid, agrid, policy = m
    @unpack α, δ, β, Na, Nk         = p
    for ai in 1:Na, ki in 1:Nk
        # Capital and productivity levels
        k = kgrid[ki]
        A = exp(agrid[ai])

        # Cash on hands
        z  = A * k^α + (1-δ) * k
        k′ = kgrid
        c  = z .- k′

        # Do a vectorized grid search
        vmax, pol = findmax(
            u.(c) .+ β .* EV[:, ai]
        )
        # Store the max values and policies
        V[ki, ai]      = vmax
        policy[ki, ai] = pol
    end
    return
end
```





How do we simulate?

```
function simulate_stochastic(p, m; T = 50)
    @unpack kgrid, agrid, P, policy = m
    @unpack α, δ = p
    # Simulate the markov chain and initializ
    A = simulate_markov(agrid, P,
        T = 100 + T)[101:end]
    C = zeros(T); K = zeros(T);
    Y = zeros(T); Ki = zeros(Int, T)
    Ki[1] = searchsortedfirst(kgrid, 12.0)
    # Step our policy function forward
    for t in 1:T
        K[t] = kgrid[Ki[t]]
        Y[t] = exp(A[t]) * K[t]^α
        z    = Y[t] + (1-δ) * K[t]
        Ai   = searchsortedfirst(agrid, A[t])
        ki′  = policy[Ki[t], Ai]
        k′   = kgrid[ki′]
        C[t] = z - k′
        if t < T
            Ki[t+1] = ki′
        end
    end
    return (; A, C, K, Y, Ki)
end
```



K_t is smoother than C_t which is smoother than A_t

Section 2

Estimating Models

How do we pick our parameters in OLS?

- When you took econometrics, you learned how to estimate:

$$y_i = X_i\beta + \epsilon_i$$

where $X_i \in \mathbb{R}^k$, and $\beta \in \mathbb{R}^k$

- Remember that OLS chooses $\widehat{\beta}$ to minimize the sum of squared residuals:

$$\widehat{\beta} \in \arg\min_{\beta} \sum_{i=1}^{n} (y_i - X_i\beta)^2$$

- It turns out, we can understand OLS as choosing $\widehat{\beta}$ in order to replicate a set of "moments" in our data

A "moment" here means any function mapping data (actual data or predicted data) to a real number. Averages, variances, covariances, etc... are all data moments

- This called the **method of moments** interpretation of OLS

How do we pick our parameters in OLS?

- When you took econometrics, you learned how to estimate:

$$y_i = X_i\beta + \epsilon_i$$

  where $X_i \in \mathbb{R}^k$, and $\beta \in \mathbb{R}^k$

- Remember that OLS chooses $\widehat{\beta}$ to minimize the sum of squared residuals:

$$\widehat{\beta} \in \arg\min_{\beta} \sum_{i=1}^{n} (y_i - X_i\beta)^2$$

- It turns out, we can understand OLS as choosing $\widehat{\beta}$ in order to replicate a set of "moments" in our data

  A "moment" here means any function mapping data (actual data or predicted data) to a real number. Averages, variances, covariances, etc... are all data moments

- This called the **method of moments** interpretation of OLS

Method of Moments: OLS

- To make this easier to see, let's stack up all our dependent observations y_i in a vector y, and all our regressors X_i in a matrix X.
- We can write $SSR(\beta)$ now as a matrix product:

$$SSR(\beta) = (y - X\beta)^T(y - X\beta)$$

- We can derive the OLS normal equations (first order conditions) by setting $\frac{\partial SSR}{\partial \beta} = 0$:

$$X^T(y - X\beta) = 0 \tag{3}$$

- If we substitute back in $\widehat{\epsilon} = y - X\beta$, we get the familiar moment conditions:

$$0 = X^T\widehat{\epsilon} = \sum_{i=1}^{n} X_i\widehat{\epsilon}_i \iff \text{Cov}(X_i, \widehat{\epsilon}_i) = 0 \iff \mathbb{E}\left[X_i\widehat{\epsilon}_i\right] = 0$$

Note: there are k different equations here, since $X_i \in \mathbb{R}^k$

- OLS chooses β so that the covariance between the predicted residuals and the regressors are zero.
- This is just a set of k different "moments"

Method of Moments: OLS

- To make this easier to see, let's stack up all our dependent observations y_i in a vector y, and all our regressors X_i in a matrix X.
- We can write $SSR(\beta)$ now as a matrix product:

$$SSR(\beta) = (y - X\beta)^T(y - X\beta)$$

- We can derive the OLS normal equations (first order conditions) by setting $\frac{\partial SSR}{\partial \beta} = 0$:

$$X^T(y - X\beta) = 0 \tag{3}$$

- If we substitute back in $\widehat{\epsilon} = y - X\beta$, we get the familiar moment conditions:

$$0 = X^T\widehat{\epsilon} = \sum_{i=1}^{n} X_i\widehat{\epsilon}_i \iff \text{Cov}(X_i, \widehat{\epsilon}_i) = 0 \iff \mathbb{E}\left[X_i\widehat{\epsilon}_i\right] = 0$$

Note: there are k different equations here, since $X_i \in \mathbb{R}^k$

- OLS chooses β so that the covariance between the predicted residuals and the regressors are zero.
- This is just a set of k different "moments"

Method of Moments: OLS

- To make this easier to see, let's stack up all our dependent observations y_i in a vector y, and all our regressors X_i in a matrix X.
- We can write $SSR(\beta)$ now as a matrix product:

$$SSR(\beta) = (y - X\beta)^T(y - X\beta)$$

- We can derive the OLS normal equations (first order conditions) by setting $\frac{\partial SSR}{\partial \beta} = 0$:

$$X^T(y - X\beta) = 0 \tag{3}$$

- If we substitute back in $\widehat{\epsilon} = y - X\beta$, we get the familiar moment conditions:

$$0 = X^T\widehat{\epsilon} = \sum_{i=1}^{n} X_i\widehat{\epsilon}_i \iff \text{Cov}(X_i, \widehat{\epsilon}_i) = 0 \iff \mathbb{E}\left[X_i\widehat{\epsilon}_i\right] = 0$$

Note: there are k different equations here, since $X_i \in \mathbb{R}^k$

- OLS chooses β so that the covariance between the predicted residuals and the regressors are zero.
- This is just a set of k different "moments"

Method of Moments: OLS

- To make this easier to see, let's stack up all our dependent observations y_i in a vector y, and all our regressors X_i in a matrix X.
- We can write $SSR(\beta)$ now as a matrix product:

$$SSR(\beta) = (y - X\beta)^T(y - X\beta)$$

- We can derive the OLS normal equations (first order conditions) by setting $\frac{\partial SSR}{\partial \beta} = 0$:

$$X^T(y - X\beta) = 0 \tag{3}$$

- If we substitute back in $\widehat{\epsilon} = y - X\beta$, we get the familiar moment conditions:

$$0 = X^T\widehat{\epsilon} = \sum_{i=1}^{n} X_i\widehat{\epsilon}_i \iff \text{Cov}(X_i, \widehat{\epsilon}_i) = 0 \iff \mathbb{E}\left[X_i\widehat{\epsilon}_i\right] = 0$$

Note: there are k different equations here, since $X_i \in \mathbb{R}^k$

- OLS chooses β so that the covariance between the predicted residuals and the regressors are zero.
- This is just a set of k different "moments"

Interpreting OLS Method of Moments

- We can now write our OLS "moment function" $m : \mathbb{R}^k \to R^k$

$$m(\beta) = X^T(y - X\beta)$$

- Since there is a unique solution to $m(\beta) = 0$, we could imagine solving the following problem instead:

$$\widehat{\beta} = \arg\min_{\beta} m(\beta)^T m(\beta) \tag{4}$$

- This is called generalized method of moments: it generalizes OLS to a much broader class of statistical models
  - In general, you can write IV regressions as a method of moments problem (with a different moment condition)
  - You can use a moment function which is nonlinear
  - Crucially, you can use a moment function which involves simulated data from a structural model

Interpreting OLS Method of Moments

- We can now write our OLS "moment function" $m : \mathbb{R}^k \to R^k$

$$m(\beta) = X^T(y - X\beta)$$

- Since there is a unique solution to $m(\beta) = 0$, we could imagine solving the following problem instead:

$$\widehat{\beta} = \arg\min_{\beta} m(\beta)^T m(\beta) \tag{4}$$

- This is called generalized method of moments: it generalizes OLS to a much broader class of statistical models
  - In general, you can write IV regressions as a method of moments problem (with a different moment condition)
  - You can use a moment function which is nonlinear
  - Crucially, you can use a moment function which involves simulated data from a structural model

Stochastic Neoclassical Growth Model: Estimating ρ

- Return to our neoclassical growth model with stochastic productivity

$$\begin{aligned} v(k, A) = \max_{c, k'} \quad & u(c) + \beta \mathbb{E}\left[v(k', A') | A\right] \\ \text{s.t.} \quad & c + k' \leq A k^{\alpha} + (1 - \delta) k \\ & \log(A') = \rho \log(A) + \epsilon \\ & \epsilon \sim N(0, \sigma) \end{aligned} \tag{2}$$

- Let's say we want to choose a value of ρ, but we don't want to just make something up. We want to estimate it to match the data.

- We know, intuitively, that ρ controls the persistence of shocks to productivity
  - If we directly observed A_t, we could estimate it with a linear regression.
  - What if, instead, we only observe C_t and Y_t?
  - Since they both move in step with A_t, a higher ρ should give us a more persistent C_t and Y_t

Stochastic Neoclassical Growth Model: Estimating ρ

- Return to our neoclassical growth model with stochastic productivity

$$\begin{aligned} v(k, A) = \max_{c, k'} \quad & u(c) + \beta \mathbb{E}\left[v(k', A') | A\right] \\ \text{s.t.} \quad & c + k' \leq A k^{\alpha} + (1 - \delta) k \\ & \log(A') = \rho \log(A) + \epsilon \\ & \epsilon \sim N(0, \sigma) \end{aligned} \tag{2}$$

- Let's say we want to choose a value of ρ, but we don't want to just make something up. We want to estimate it to match the data.

- We know, intuitively, that ρ controls the persistence of shocks to productivity
  - If we directly observed A_t, we could estimate it with a linear regression.
  - What if, instead, we only observe C_t and Y_t?
  - Since they both move in step with A_t, a higher ρ should give us a more persistent C_t and Y_t

Stochastic Neoclassical Growth Model: Estimating ρ

- Return to our neoclassical growth model with stochastic productivity

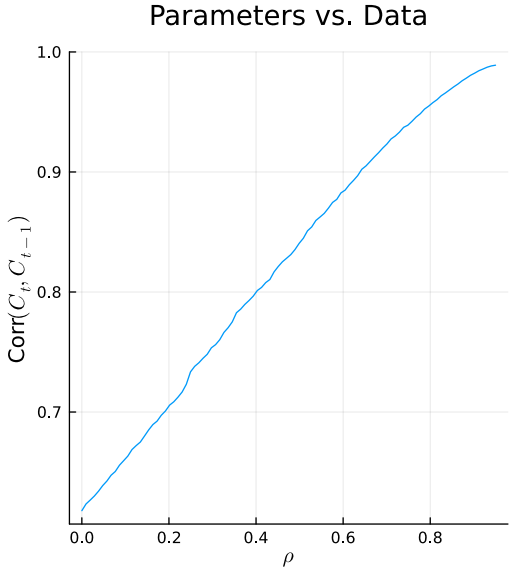
$$\begin{aligned} v(k, A) = \max_{c, k'} \quad & u(c) + \beta \mathbb{E}\left[v(k', A') | A\right] \\ \text{s.t.} \quad & c + k' \leq A k^{\alpha} + (1 - \delta) k \\ & \log(A') = \rho \log(A) + \epsilon \\ & \epsilon \sim N(0, \sigma) \end{aligned} \tag{2}$$

- Let's say we want to choose a value of ρ, but we don't want to just make something up. We want to estimate it to match the data.

- We know, intuitively, that ρ controls the persistence of shocks to productivity
  - If we directly observed A_t, we could estimate it with a linear regression.
  - What if, instead, we only observe C_t and Y_t?
  - Since they both move in step with A_t, a higher ρ should give us a more persistent C_t and Y_t

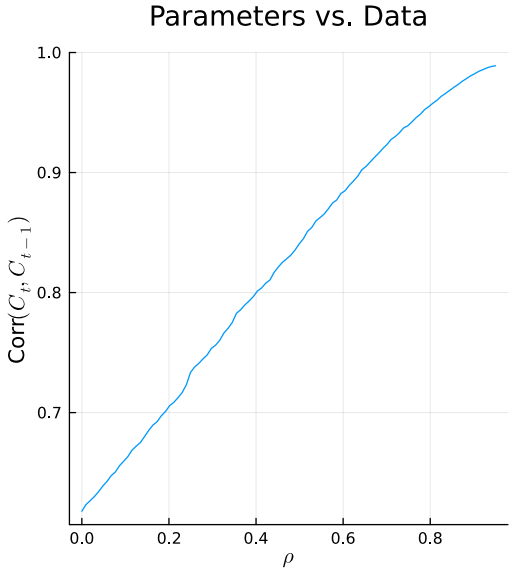
ρ controls Corr(C_t, C_{t-1}) too

```julia
function compute_moments(p; T = 1_000_000)
    @unpack Nk, Na = p
    mc = rouwenhorst(Na, p.ρ, p.σ)
    m  = (
        V       = zeros(Nk, Na),
        V0      = zeros(Nk, Na),
        policy  = zeros(Int, Nk, Na),
        kgrid   = LinRange(1e-1, 30, Nk),
        agrid   = mc.state_values,
        P       = mc.p'
    )
    solve_vfi_stochastic!(p, m)
    sim = simulate_stochastic(p, m; T)
    @unpack C = sim

    # Calculate autocorrelation of C
    ρC = cor(C[1:end-1], C[2:end])
    return ρC
end
```



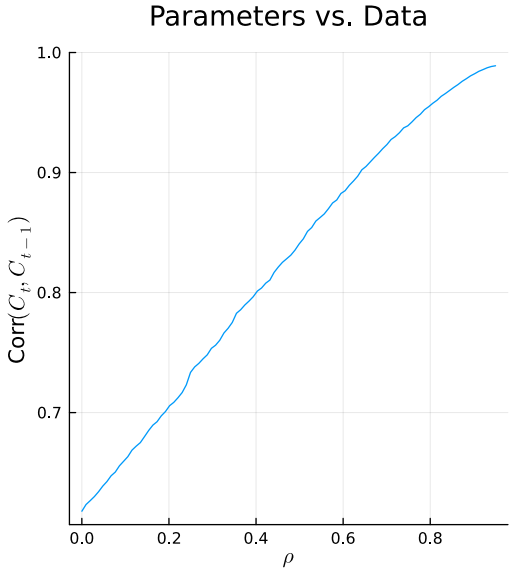
ρ controls $\text{Corr}(C_t, C_{t-1})$ too

- What does this tell us?
- Autocorrelation of consumption increases monotonically with ρ,
  - We should be able to back out a value of ρ for any "reasonable" value of autocorrelation in consumption that we observe in the data
- Reasonable means something very specific here:
  - with the other parameters we've set, we cannot get the model to generate autocorrelation in consumption below around 0.6
  - It's possible we could if we changed the curvature of the utility function (i.e. made the household less risk averse, so they smooth consumption less)



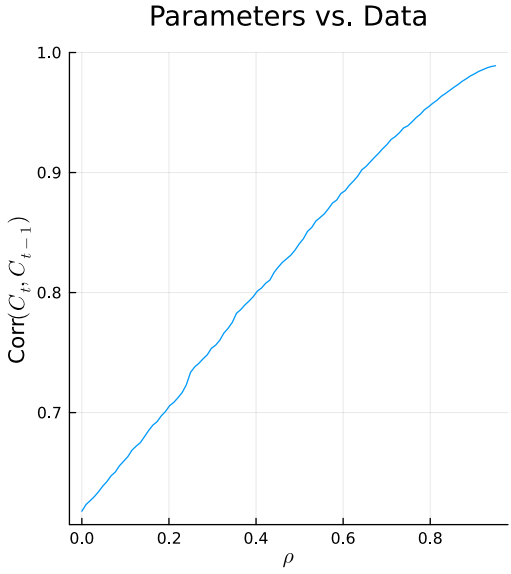
ρ controls $\text{Corr}(C_t, C_{t-1})$ too

- What does this tell us?
- Autocorrelation of consumption increases monotonically with ρ,
  - We should be able to back out a value of ρ for any "reasonable" value of autocorrelation in consumption that we observe in the data
- Reasonable means something very specific here:
  - with the other parameters we've set, we cannot get the model to generate autocorrelation in consumption below around 0.6
  - It's possible we could if we changed the curvature of the utility function (i.e. made the household less risk averse, so they smooth consumption less)



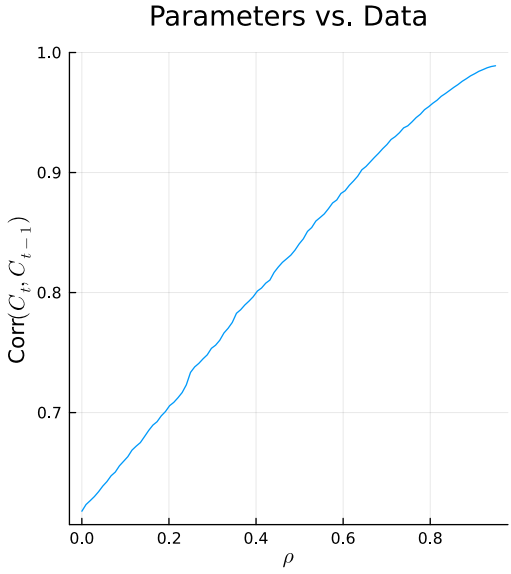
ρ controls Corr(C_t, C_{t-1}) too

- What does this tell us?
- Autocorrelation of consumption increases monotonically with ρ,
  - We should be able to back out a value of ρ for any "reasonable" value of autocorrelation in consumption that we observe in the data
- Reasonable means something very specific here:
  - with the other parameters we've set, we cannot get the model to generate autocorrelation in consumption below around 0.6
  - It's possible we could if we changed the curvature of the utility function (i.e, made the household less risk averse, so they smooth consumption less)



ρ controls Corr(C_t, C_{t-1}) too

- What does this tell us?
- Autocorrelation of consumption increases monotonically with ρ,
  - We should be able to back out a value of ρ for any "reasonable" value of autocorrelation in consumption that we observe in the data
- Reasonable means something very specific here:
  - with the other parameters we've set, we cannot get the model to generate autocorrelation in consumption below around 0.6
  - It's possible we could if we changed the curvature of the utility function (i.e, made the household less risk averse, so they smooth consumption less)



Estimating ρ

- Suppose you were told (or calculated yourself) that consumption has an autocorrelation coefficient of 0.9

  I made this number up – it's just an example

- How would we choose ρ so that our model matches the data?
- Option 1: treat this as a root finding problem
  - Works fine for just 1 parameters
  - Runs into issues for many parameters
  - Simulated moments are either very hard to differentiate, or not differentiable at all, so you will find it hard to calculate a gradient/hessian

Estimating ρ

- Suppose you were told (or calculated yourself) that consumption has an autocorrelation coefficient of 0.9

  I made this number up – it's just an example

- How would we choose ρ so that our model matches the data?
- Option 1: treat this as a root finding problem
  - Works fine for just 1 parameters
  - Runs into issues for many parameters
  - Simulated moments are either very hard to differentiate, or not differentiable at all, so you will find it hard to calculate a gradient/hessian

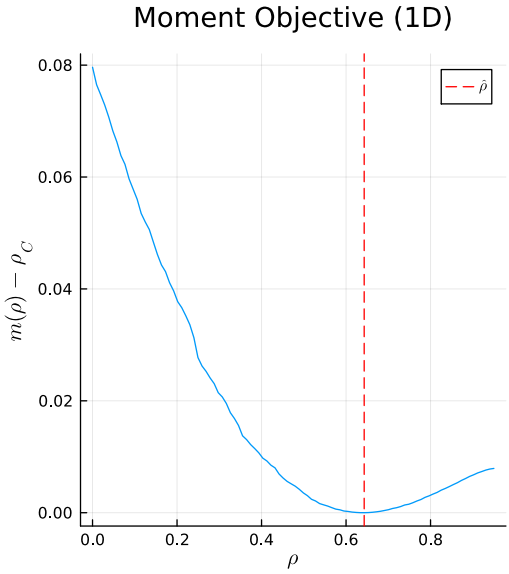
Estimating ρ

- Option 2: treat this as an optimization problem

$$\widehat{\rho} = \arg\min \rho \ \ (m(\rho) - \rho_C)^2 \qquad (5)$$

  where $m(\rho)$ is the model simulated ρ_C (autocorrelation of consumption)

- This gives the same answer in the 1D case
- Generalizes better if you want to estimate many parameters at once



Simulated Method of Moments

- If we generalize this to several different moments from our data/model, we arrive at the algorithm called **Simulated Method of Moments**
- Suppose our model has a vector of k parameters $\theta \in \mathbb{R}^k$, which we want to estimate
- Suppose further, we have a set of $n \geq k$ moments $\overline{m} \in \mathbb{R}^n$ (calculated in the data) that are "informative" about the underlying parameters
- Let $m : \mathbb{R}^k \to \mathbb{R}^k$ be the function that
  1. Takes the parameter θ, and solves our underlying model
  2. Simulates the model, and
  3. Calculates the model analogue of the moments in the data
- We choose our estimate $\widehat{\theta}$ to solve

$$\widehat{\theta} \in \arg\max_{\theta} \left(m(\theta) - \overline{m}\right)^T W \left(m(\theta) - \overline{m}\right) \tag{6}$$

where W is a weighting matrix that controls how important each moment is.

For this course, you will generally just set $W = I$, and then this is just the sum of squared deviations of model moments from data moments

Simulated Method of Moments

- If we generalize this to several different moments from our data/model, we arrive at the algorithm called **Simulated Method of Moments**
- Suppose our model has a vector of k parameters $\theta \in \mathbb{R}^k$, which we want to estimate
- Suppose further, we have a set of $n \geq k$ moments $\overline{m} \in \mathbb{R}^n$ (calculated in the data) that are "informative" about the underlying parameters
- Let $m : \mathbb{R}^k \to \mathbb{R}^k$ be the function that
  1. Takes the parameter θ, and solves our underlying model
  2. Simulates the model, and
  3. Calculates the model analogue of the moments in the data
- We choose our estimate $\widehat{\theta}$ to solve

$$\widehat{\theta} \in \arg\max_{\theta} \left(m(\theta) - \overline{m}\right)^T W \left(m(\theta) - \overline{m}\right) \tag{6}$$

where W is a weighting matrix that controls how important each moment is.

For this course, you will generally just set $W = I$, and then this is just the sum of squared deviations of model moments from data moments

Simulated Method of Moments

- If we generalize this to several different moments from our data/model, we arrive at the algorithm called **Simulated Method of Moments**
- Suppose our model has a vector of k parameters $\theta \in \mathbb{R}^k$, which we want to estimate
- Suppose further, we have a set of $n \geq k$ moments $\overline{m} \in \mathbb{R}^n$ (calculated in the data) that are "informative" about the underlying parameters
- Let $m : \mathbb{R}^k \to \mathbb{R}^k$ be the function that
  1. Takes the parameter θ, and solves our underlying model
  2. Simulates the model, and
  3. Calculates the model analogue of the moments in the data
- We choose our estimate $\widehat{\theta}$ to solve

$$\widehat{\theta} \in \arg\max_{\theta} \left(m(\theta) - \overline{m}\right)^T W \left(m(\theta) - \overline{m}\right) \tag{6}$$

where W is a weighting matrix that controls how important each moment is.

For this course, you will generally just set $W = I$, and then this is just the sum of squared deviations of model moments from data moments

Simulated Method of Moments: Very computationally intensive

- In general, you have to use a derivative free optimization algorithm (like Nelder-Mead) to solve this problem (if you have even 4 or 5 parameters, this means 100s or 1000s of function evaluations)
- Even worse, there are local minima everywhere.

  Often need to restart optimization from many different starting values and repeat to be sure you've found global min
- With model solves inside every function evaluation, this could easily run for hours/days/weeks, depending on how large the problem is

  Obviously, I won't ask you to solve a problem that has to run for days...
- You really want to try your best to get blazing fast performance when solving your model

Simulated Method of Moments: Very computationally intensive

- In general, you have to use a derivative free optimization algorithm (like Nelder-Mead) to solve this problem (if you have even 4 or 5 parameters, this means 100s or 1000s of function evaluations)
- Even worse, there are local minima everywhere.

  Often need to restart optimization from many different starting values and repeat to be sure you've found global min
- With model solves inside every function evaluation, this could easily run for hours/days/weeks, depending on how large the problem is

  Obviously, I won't ask you to solve a problem that has to run for days...
- You really want to try your best to get blazing fast performance when solving your model

Simulated Method of Moments: Very computationally intensive

- In general, you have to use a derivative free optimization algorithm (like Nelder-Mead) to solve this problem (if you have even 4 or 5 parameters, this means 100s or 1000s of function evaluations)
- Even worse, there are local minima everywhere.

  Often need to restart optimization from many different starting values and repeat to be sure you've found global min
- With model solves inside every function evaluation, this could easily run for hours/days/weeks, depending on how large the problem is

  Obviously, I won't ask you to solve a problem that has to run for days...
- You really want to try your best to get blazing fast performance when solving your model

Simulated Method of Moments: Very powerful

- All you need are "informative" moments
  - If $f(\theta)$ is your objective, then you need $Df(\theta^\star)$ to have full rank (be invertible) at the true parameters
- Any statistic can be "informative" if it captures the right thing (although choosing moments is a bit of an art)

  You can use aggregate statistics, which is great for working with confidential microdata
- In principle you can use regression coefficients that are badly identified, so long as the identification problem you're worried about in the data is also present in the model

  Example: labor search models with unobserved heterogeneity in human capital. There is selection on who is hired out of unemployment (since employers see your skill)

  As long as you model the unobserved heterogeneity in human capital, you can make use of a regression of earnings just out of unemployment on duration of unemployment

Simulated Method of Moments: Very powerful

- All you need are "informative" moments
  - If $f(\theta)$ is your objective, then you need $Df(\theta^\star)$ to have full rank (be invertible) at the true parameters
- Any statistic can be "informative" if it captures the right thing (although choosing moments is a bit of an art)

  You can use aggregate statistics, which is great for working with confidential microdata
- In principle you can use regression coefficients that are badly identified, so long as the identification problem you're worried about in the data is also present in the model

  Example: labor search models with unobserved heterogeneity in human capital. There is selection on who is hired out of unemployment (since employers see your skill)

  As long as you model the unobserved heterogeneity in human capital, you can make use of a regression of earnings just out of unemployment on duration of unemployment

Simulated Method of Moments: Very powerful

- All you need are "informative" moments
  - If $f(\theta)$ is your objective, then you need $Df(\theta^\star)$ to have full rank (be invertible) at the true parameters
- Any statistic can be "informative" if it captures the right thing (although choosing moments is a bit of an art)

  You can use aggregate statistics, which is great for working with confidential microdata
- In principle you can use regression coefficients that are badly identified, so long as the identification problem you're worried about in the data is also present in the model

  Example: labor search models with unobserved heterogeneity in human capital. There is selection on who is hired out of unemployment (since employers see your skill)

  As long as you model the unobserved heterogeneity in human capital, you can make use of a regression of earnings just out of unemployment on duration of unemployment

Summary

- Saw today that you can simulate your model fairly easily once you've solved it

  Just need to repeatedly apply the policy functions you've solved for

- Simulating the model can be extremely helpful for estimating your model to match the data
  - Simulated method of moments is a very powerful tool for estimating structural models
  - Only requires that parameters be "informative" about something in the data to estimate them
  - It's generally quite tricky to pick sensible moments for estimation (something of an art)
- Next week, we'll return to see how we can use these models to run policy counterfactuals