Lecture Notes 8: Empirical Applications

Ivan Rudik

ECON 509: Computational Methods

April 1, 2016

In addition to computing solutions to analytical models, we can use numerical methods to estimate models with real world data. The most common approach is Maximum Likelihood Estimation (MLE). There are also estimation procedures, like Nested Fixed Point (NXFP) for estimating dynamic structural models (Rust, 1987). We will go over some basic approaches to estimating models with numerical methods

1 Maximum Likelihood

When entering the empirical world, there is a set of data generating processes (DGPs) that may have generated the data we are using. Suppose that our data consist of realizations of some random variable $Z \sim f_Z(Z;\theta^*)$ where θ^* is a vector of true parameters governing the DGP. The likelihood function is simply $\mathcal{L}(\theta) = f_Z(Z;\theta)$: the pdf evaluated at some vector θ. If there are multiple random variables, the likelihood is based on the joint pdf, $\mathcal{L}(\theta) = f_{Z_1,Z_2,...Z_n}(Z_1, Z_2, ..., Z_n;\theta)$. If these random variables, $Z_1, Z_2, ..., Z_n$ are IID, then the likelihood is simply the product of the marginal pdfs, $\mathcal{L}(\theta) = \prod_{i=1}^{n} f_{Z_i}(Z_i;\theta)$. For tractability, we prefer to work with the log likelihood, $\log\mathcal{L}(\theta) = \sum_{i=1}^{n} \log f_{Z_i}(Z_i;\theta)$.

The maximum likelihood estimator of θ^* is the value of θ that maximizes the likelihood or log likelihood function. In effect, it finds the model, within the class of models f_Z that best matches the observed data (selecting parameters that maximize the likelihood that the data were generated by this class of models).

We can think about a simple exam, suppose X is distributed normally with some unknown

mean and unit variance. The pdf of X is,

$$f_X(x;\mu) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{1}{2}(x-\mu)^2\right).$$

The likelihood is therefore,

$$\mathcal{L}(\mu) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{1}{2}(X-\mu)^2\right),$$

and the log likelihood is,

$$log\mathcal{L}(\mu) = -\frac{1}{2}log(2\pi) - \frac{1}{2}(X-\mu)^2.$$

What value of μ, our unknown parameter, maximizes our log likelihood? Simply $\hat{\mu}_{ML} = X$. If we have n random variables that are IID $N(\mu, 1)$ then our log likelihood is,

$$\mathcal{L}(\mu) = -n\frac{1}{2}log(2\pi) - \sum_{i=1}^{n}\frac{1}{2}(X_i-\mu)^2.$$

The argmax of the log likelihood is just, $\hat{\mu}_{ML} = \bar{X}$, the sample average of the $X_i's$.

Now we know how to maximize a likelihood, but why is this a logical strategy for estimating parameters of a model? Because we can appeal to large sample approximations. Suppose we have some random variable Y which is the ratio of the density function at some arbitrary θ to the density function at θ^*, the true theta, both evaluated at some X,

$$Y = f_X(X;\theta)/f_X(X;\theta^*)$$

. Suppose $g(\cdot)$ is the negative logarithm, $g(a) = -log(a), g'(a) = -1/a, g''(a) = 1/a^2 > 0$. $g(a)$ is clearly convex so Jensen's inequality states that,

$$E[g(Y)] \geq g(E[Y]).$$

This implies that,

$$E\left[-log\left(\frac{f_X(X;\theta)}{f_X(X;\theta^*)}\right)\right] \geq -log\left(E\left[\frac{f_X(X;\theta)}{f_X(X;\theta^*)}\right]\right).$$

What is the expectation over? The distribution of X: $f_X(x;\theta^*)$. This implies that the expectation on the right hand side is,

$$E\left[\frac{f_X(X;\theta)}{f_X(X;\theta^*)}\right] = \int \frac{f_X(X;\theta)}{f_X(X;\theta^*)} f_X(X;\theta^*)dx = \int f_X(X;\theta)dx = 1.$$

So for any value of θ, after taking a log we have that,

$$E\left[-log\left(\frac{f_X(X;\theta)}{f_X(X;\theta^*)}\right)\right] \geq 0$$

. This implies that,

$$E\left[logf_X(X;\theta^*)\right] \geq E\left[logf_X(X;\theta)\right], \ \ \forall\theta.$$

This tells us that the expected value of our log likelihood is maximized at the true value of θ, so we have a shot that our actual log likelihood is maximized near θ^*, so that $\hat{\theta}_{ML}$ is a good estimate of θ^*. We can illustrate this by considering an example where we have n Bernoulli trials with some unknown probability p^*. The joint density of these trials is,

$$f_{X_1,...,X_n} = (x_1, ..., x_n, ; p) = p^{\sum x_i}(1-p)^{N-\sum x_i}.$$

The log likelihood of this joint distribution is,

$$\mathcal{L}(p) = \sum X_i log(p) + (n - \sum X_i) log(1-p).$$

The MLE of this is, $\hat{p}_{ML} = \bar{X}$, the sample average. Now let us go to the expected log likelihood,

$$E\left[\mathcal{L}(p)\right] = E\left[\sum X_i log(p) + (n - \sum X_i) log(1-p)\right].$$

This reduces to,

$$E\left[\mathcal{L}(p)\right] = n \cdot p^* \cdot log(p) + n \cdot (1-p^*) \cdot log(1-p).$$

This function is maximized at $\hat{p} = p^*$, the true value. Therefore if we take the expected log likelihood (just multiply by $1/n$), we will have a sample average. Sample averages are subject to the law of large numbers so we should expect that the sample average log likelihood is close to the true expected log likelihood.

1.1 MLE in practice: Simulating data generating processes

How do we employ MLE in practice? We can learn how by simulating a data generating process (DGP) and then estimating its parameters.[1] Let us begin with a simple linear model,

$$Y = X\beta + \epsilon,$$

[1] This is a nice way to error check code for complex estimation problems. If you know how the data were generated, but your estimator doesn't reflect the parameters of the DGP, then your code must have an error or you got very unlucky with random draws.

or in other notation,

$$Y \sim N(X\beta, \sigma^2),$$

where Y is Nx1, X is NxK and β is Kx1, and $\epsilon \sim N(0, \sigma^2)$. We can simulate independent variables using any distribution we want, but let us begin with the multivariate normal,

```
N = 100;
X = mvnrnd([0 0 0], eye(3), N);
X = [ones(N, 1) X];
```

This simply draws N samples from our multivariate normal distribution of 3 mean zero, unit variance, independent normal distributions. We also want a constant term, so we must concatenate that in manually. Given these $X's$, we need a vector of true parameters and draws of random shocks to generate our dependent variable, Y,

```
true_betas = [0.1, 0.5, -0.3, 0.]';
epsilons = normrnd(0,1,[100,1]);
Y = X*true_betas + epsilons;
```

You can test that this worked by checking the mean of Y. It should be close to 0.1 (intercept term) since our X's were distributed normal with mean zero.

Now we wish to estimate the parameters of our model, the βs. How do we do this? We know that $Y \sim N(X\beta, \sigma^2)$, but we can rearrange this expression so that,

$$Y - X\beta \sim N(0, \sigma^2),$$

and we have a distribution that does not depend on X. Given the independence across observations, we have that the likelihood is,

$$\mathcal{L}(\beta, \sigma^2) = \prod_{i=1}^{N} \phi(Y_i - X_i\beta, \sigma^2 | \beta, \sigma^2),$$

where ϕ denotes the pdf of the normal distribution. Let $\rho = [\beta, \sigma^2]$. We will be estimating some $\hat{\rho}$ which is the argmax of the likelihood. We now need to estimate these parameters. To do so, we define a negative loglikelihood function to maximize,

```
function nloglikelihood = loglike(Y,X,rho)
    beta = rho(1:4);
    sigma2 = exp(rho(5));
    residual = Y - X*beta;
```

```
    nloglikelihood = normlike([0, sigma2], residual);
end
```

This function takes data Y and X, and an input of parameters *rho*, and returns the negative log likelihood. Notice that we take the exponential of the parameter that governs the residual variance. This is to ensure that guesses are strictly positive, we will have to undo this transformation later. Now that we have this function, we can send it to one of our maximization routines to search over parameters that minimize the negative log likelihood conditional on the data.

```
initial_rhos = [1 1 1 1 1]';
log_handle = @(rho) loglike(Y,X,rho);
mle = fminunc(log_handle, initial_rhos);
mle(5) = exp(mle(5));
```

This will yield an estimate of the true parameters. As we increase our sample size N, our estimates should get closer and closer to the true values via the law of large numbers argument.

We have estimated our parameters, but as economists we also care about how precise our estimates are: we need to get standard errors. We get standard errors for MLE by using a procedure called bootstrapping. Bootstrapping is a process where we can estimate properties of an estimator. In this case, we wish to estimate the standard errors of our estimates of ρ, $\hat{\rho}$. We measure these properties by drawing samples of data from some approximating distribution to the true distribution. In practice, we typically use the empirical distribution of the observed data. Under the assumption that the data in our sample are generated in an IID fashion, then we can resample our data with replacement to generate new datasets that would be representative of drawing from the actual DGP/population.

We then use these new datasets to re-perform the MLE procedure and get alternative estimates of ρ based on these alternative draws of data from our approximated DGP/population of data. If our original sample is a good approximation to the true DGP/population, then the sampling distribution of $\hat{\rho}$s we obtain should be a good approximation to the distribution of the $\hat{\rho}^*$ we would obtain if we resampled from the true DGP/population. If we take the standard deviation of the sampling distribution of $\hat{\rho}$, then we have an estimate of the standard deviation of $\hat{\rho}^*$ if we resampled the true distribution. If this standard deviation, which we call the standard error of our estimate $\hat{\rho}$, is large then using a sample of this size from the true population could give very different estimates depending on which data you happened to draw. Large standard errors imply less precision in our estimate. Bootstrapping is a nice non-parametric way to generate standard errors. In no point during this process did we assume any distribution on our data other than via re-estimating $\hat{\rho}$.

First we must construct a bootstrap function,

```
function samples = bootstrap_mle(N,X,Y,initial_rhos)
    options = optimset('display','off');
    sample_index = datasample(1:N,N);
    X_boot = X(sample_index,:);
    Y_boot = Y(sample_index,:);
    samples = fminunc(@(rho)loglike(Y_boot,X_boot,rho),initial_rhos,options);
    samples(5) = exp(samples(5));
end
```

This function takes in two sets of data, samples the sets of data with replacement, and then uses the re-sampled data to re-maximize the log likelihood. It then returns the optimal parameter values, $\hat{\rho^b}$ for samples $b = 1, ..., B$.

```
num_samples = 1000;
samples = zeros(num_samples,5);
for b = 1:num_samples
    samples(b,:) = bootstrap_mle(N,X,Y,initial_rhos);
end

% Calculate standard errors
bootstrapSE = std(samples,1);
```

With our bootstrap function, we then loop over how many times we want to bootstrap and get many different parameter estimates using different draws from our sample. Our bootstrapped standard errors are just the standard deviations of the parameter estimates. Typically we don't just stop at standard errors, we often want to test the null hypothesis that our parameters are zero. If we can safely assume the MLE is normally distributed, like it is here, then we can reject the hypothesis that the parameter is zero if it is at least 1.96 standard errors away from zero.

We can also do this without assuming a distribution by constructing non-parametric p-values. In this case, we take our bootstrapped samples and use them to create the distribution implied by the null hypothesis, that the βs are zero.

```
for i = 1:5
    null_dist(:,i) = null_dist(:,i) - mean(null_dist(:,i));
end
```

Now our null distribution (which we obtained via bootstrapping) that we are testing our true distribution against is mean zero. To perform a two-tailed hypothesis test that our parameter estimates are non-zero, we calculate the fraction of times that the absolute value of the MLE is less than the absolute value of the null hypothesis, i.e. when is the null distribution as extreme as our MLE?

```
for i = 1:4
    pvalues(i,1) = mean(abs(mle(i))<abs(null_dist(:,i)));
end
```

We can also check the precision of our estimates visually by plotting the estimates along with the standard errors and 95% confidence intervals.

```
% Plot estimates with 95% confidence intervals
errorbar(0:3,mle(1:4),conf_radius(1:4),'.');
hold on
plot(-1:.1:4,zeros(size(-1:.1:4)))

% Plot estimates with standard errors
errorbar(0:3,mle(1:4),bootstrapSE(1:4),'.');
hold on
plot(-1:.1:4,zeros(size(-1:.1:4)))
```

2 Dynamic Structural Estimation: Nested Fixed Point

What other types of empirical models do we need computational methods for? One is to do dynamic structural estimation of discrete choice models. This began with the seminal paper by Rust (1987) who develops the nested fixed point (NFXP) estimator. NFXP is a MLE algorithm for a dynamic model. These models are unlikely to have analytic solutions so they must be computed. The main idea behind NFXP is to do two things,

1. Solve an MLE problem
2. Solve a fixed point problem within the MLE problem

Lets learn the NFXP algorithm by studying Harold Zurcher of Rust (1987) fame. Harold is the maintenence manager of a bus company. His objective is to determine how long a bus should

operate before it is optimal to replace the engine. Failing on the road has significant costs: towing, time, customer satisfaction. Therefore Harold wants to take a precautionary replacement strategy rather than replace engines only when they break down. Therefore we want a model which,

1. Specifies the economic agent
2. Specifies the state variables, denoted (x, ϵ)
3. Specifies the state-dependent choice sets, $D(x)$
4. Specifies functional forms for utility, $u(x, d, \theta)$ and the transitions $p(x'|x, d, \theta)$

The bus dataset has monthly observations of 162 buses over a 6 month time frame. The data we have on each bus is the accumulated mileage x_t since the last engine replacement, and whether the engine was replaced, which is denoted by d_t. $d_t = 1$ indicates the engine was replaced in month t, $d_t = 0$ indicates the engine was not replaced in month t. The set of observations we have is $\{x_1, ..., x_T, d_1, ..., d_T\}$. Harold's choice set is $D(x_t) = \{0, 1\}$, to replace or not replace the bus engine. Harold's payoff is given by,

$$u(x, d, \theta) = \begin{cases} -RC - c(0, \theta) + \epsilon(1) \text{ if } d = 1 \\ -c(x, \theta) + \epsilon(0) \text{ if } d = 0 \end{cases}$$

RC is the total replacement costs of scrapping the old engine and installing a new one. RC is technically something we will be estimating (as econometricans we don't know this value and it's not obvious as a function of the data). Once the engine as been replaced, e.g. $x = 1$, mileage reverts to 0. θ is a vector of cost parameters that we will estimate using the data. ϵ is a vector of state variables that are observed by the agent, but not by the econometrican. The ϵs are error terms that account for how we cannot perfectly predict the agent's actions. The transition distribution for our mileage state is given by,

$$p(x_{t+1}|x_t, d_t, \theta) = \begin{cases} g(x_{t+1} - 0|\theta) \text{ if } d = 1 \\ g(x_{t+1} - x_t|\theta) \text{ if } d = 0 \end{cases}$$

where g is the exponential distribution. Therefore, mileage follows a random walk.

The unobserved (to the econometrician) state variables, $\epsilon_t(0)$ and $\epsilon_t(1)$ are assumed to be distributed type 1 extreme value. The mean is assumed to be 0 and variances are assumed to be $\pi/6$. What are these states? $\epsilon_t(0)$ can be interpreted as the (unobserved) maintenance and operating costs of the bus in period t. These unobserved state enter the model as stochastic shocks. Imagine what a large negative value of $\epsilon_t(0)$ implies. This means that operation costs were high in

period t, perhaps because of a critical part failure that makes continuing the use of the bus with the current engine extremely costly. If $\epsilon(0)$ were large and positive, then that may imply that the engine is performing excellently and does not need maintenance at all.

Now let us consider the $\epsilon_t(1)$s. If these are large and negative, then the cost of replacing an engine is unexpectedly high to the econometrician. Perhaps because all mechanics and service bays are busy, or there are no replacement engines that are currently available. A large positive value would imply that replacement costs are low, perhaps because the mechanics aren't busy or there are cheap replacement engines on the market. We cannot identify these costs without additional information (recall we only have mileage and engine replacement data).

We can now formulate Harold's Bellman equation,

$$V_\theta(x,\epsilon) = \max_{d\in D(x)} \left[u(x,d,\theta) + \beta \int V_\theta(x',\epsilon')\pi(dx', d\epsilon'|x,\epsilon,\theta) \right] \tag{1}$$

where $\pi(\cdot)$ denotes the joint transition density for x and ϵ.[2] We can define the *expected value function* as

$$EV_\theta(x,d) = \int V_\theta(x',\epsilon')\pi(dx', d\epsilon'|x,\epsilon,d,\theta), \tag{2}$$

which is a function of current states. Now we can write the Bellman more compactly as,

$$V_\theta(x,\epsilon) = \max_{d\in D(x)} \left[u(x,d,\theta) + \beta EV_\theta(x,d) \right]. \tag{3}$$

Given the data and this Bellman, we need a way to estimate the parameters, θ and RC. We will do this via maximum likelihood. With MLE, we need the pdf $L(x_1,...,x_T,d_1,...,d_T|\theta)$ to compute the $\hat{\theta}$ which maximizes the likelihood function of our sample. The question is, how do we get the pdf of our data? To do this we need to solve the Bellman in equation (1). A key assumption we need to make is that of *conditional independence*:

$$\pi(x_{t+1},\epsilon_{t+1}|x_t,\epsilon_t,d_t,\theta) = p(x_{t+1}|x_t,d_t,\theta)q(\epsilon_t|x_t,\theta). \tag{4}$$

We are assuming that our type 1 extreme value errors are independent of the mileage transition. Given this assumption, Rust (1987) shows that we can write the likelihood of our data as,

$$L(\theta) = \prod_{t=2}^{T} P(d_t|x_t,\theta)p(x_t|x_{t-1},d_{t-1},\theta), \tag{5}$$

[2] We will return to this later.

where $p(\cdot)$ is just the transition density of mileage, and $P(\cdot)$ is called the *conditional choice probability*,

$$P(d|x,\theta) = \frac{exp\left\{u(x,d,\theta) + \beta EV_\theta(x,d)\right\}}{\sum_{d' \in D(x)} exp\left\{u(x,d',\theta) + \beta EV_\theta(x,d')\right\}}. \tag{6}$$

This is just the standard multinomial logit formula! We can re-define EV_θ as the fixed point to a contraction mapping, $T_\theta(EV_\theta) = EV_\theta$ by seeing that,

$$\begin{aligned}
V_\theta(x',\epsilon') &= \max_{d \in D(x)} \left[u(x',d',\theta) + \beta \int V_\theta(x'',\epsilon'')\pi(dx',d\epsilon'|x,\epsilon,\theta) \right] \\
\Rightarrow EV(x,d) &= \int\int V_\theta(x',\epsilon')p(dx'|x,\epsilon,d,\theta)q(d\epsilon'|x',\theta) \\
&= \int\int \max_{d' \in D(x')} \left[u(x',d',\theta) + \beta EV_\theta(x',d')\right] p(dx'|x,\epsilon,d,\theta)q(d\epsilon'|x',\theta) \\
EV_\theta(x,d) = T_\theta(EV_\theta)(x,d) &= \int log\left(\sum_{d' \in D(x')} exp\left[u(x',d',\theta) + \beta EV_\theta(x',d')\right] \right) p(dx'|x,\epsilon,d,\theta)
\end{aligned} \tag{7}$$

The last line uses a closed-form expression for the expectation of a maximization over extreme-value variables.

Now we wish to write out the form of EV_θ and the conditional choice probability. Notice that $EV_\theta(x,1) = EV_\theta(0,0)$. If you replace the engine at any arbitrary level of mileage, your expected continuation value is equal to that of the expected continuation value of having an engine with 0 miles. This means that we only need to account for one expected value, $EV_\theta(x,0)$. Denote this as just $EV_\theta(x)$. With this we can show that,

$$EV_\theta(x) = \int_0^\infty log\left[exp\left\{-c(x+y,\theta) + \beta EV_\theta(x+y)\right\} + exp\left\{-RC - c(0,\theta) + \beta EV_\theta(0)\right\}\right] g(dy|\theta) \tag{8}$$

where y is the increment to the bus' mileage. We can also reduce the conditional choice probability of keeping the current engine to the following,

$$P(0|x,\theta) = \frac{1}{1 + exp\left\{c(x,\theta) - RC - \beta EV_\theta(x) - c(0,\theta) + \beta EV_\theta(0)\right\}} \tag{9}$$

2.1 The NFXP algorithm

Now we have fully defined our problem. How do we employ the NFXP algorithm? It is really two processes: an outer MLE that searches for the argmax to the likelihood $L(\theta)$ and an inner

fixed point that computes EV_θ conditional on the current estimate of θ, $\hat{\theta}$. This corresponds to the following maximization problem,

$$\max_{\theta, EV} L(\theta, EV) \text{ subject to } EV = T_\theta(EV) \tag{10}$$

Since the fixed point is a function of θ, EV will be a function of θ. Thus we can re-write the constrained problem as an unconstrained problem,

$$\max_{\theta} L(\theta, EV_\theta) \tag{11}$$

This implies that we need to calculate what EV_θ is before we can perform the maximization.

3 The Nested Fixed Point Algorithm

The NFXP algorithm consists of two, nested loops. In the outer loop we search for the parameter values, $\hat{\theta}$ that maximize our likelihood. In the inner loop, we compute a value function $V(x; \hat{\theta})$ conditional on some $\hat{\theta}$.

3.1 Outer Loop

Recall our likelihood is,

$$L(\theta) = \prod_{t=2}^{T} P(d_t|x_t, \theta) p(x_t|x_{t-1}, d_{t-1}, \theta), \tag{12}$$

therefore our log likelihood is,

$$logL(\theta) = \sum_{t=2}^{T} log\left(P(d_t|x_t, \theta)\right) + log\left(p(x_t|x_{t-1}, d_{t-1}, \theta)\right) \tag{13}$$

We can estimate θ with MLE. We can first estimate the components of θ that enter the transition probability since they are additively separable from the rest of the log likelihood. When this step is performed, the mileage is typically assumed to be discrete and the data are binned.

To estimate the rest of the parameters we assume the errors are IID Type 1 extreme value and maximize the log likelihood with respect to the cost parameters and replacement cost. In this case, we are just maximizing the first term which is our log conditional choice probability.

4 Inner Loop

We now have a set of parameters θ_t at iteration t of the algorithm. How do we compute the value function? We do this by iterating on the Bellman equation like we learned before. It's a contraction mapping so we know that we will find the fixed point. The clever aspect of this algorithm is that we will iterate over EV_θ and not the value function itself. By taking this route, we no longer need to calculate the value function at specific values of the error terms since they have been averaged out. Therefore, we have two fewer states!

Specifically, we iterate over equation 8. Let τ indicate iterations of this procedure. We also must discretize x to a grid. Begin at $\tau = 0$. We guess some initial EV_θ^0, and use equation 8 to compute $EV_t^1 heta$, where now since x is discretized we will have some probability of transitioning into different mileage bins between the grid points. We compute EV_θ^1 by interpolating between the grid points. Continue until some convergence criterion is met.

Once we have our fixed point, $\tilde{EV}_\theta$ conditional on being at outer loop iteration t, this yields us a different EV_θ for iteration $t+1$ in our outer loop that enters our conditional choice probability. We then repeat the entire process until the parameter estimates converge.

References

Rust, John (1987) "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher," *Econometrica*, Vol. 55, No. 5, pp. 999–1033.