Solving Dynamic Discrete Choice Models: Integrated or Expected Value Function?

Patrick Kofod Mogensen

January 15, 2018

1 Dynamic Discrete Choice Models

Research in, and applications of, Dynamic Discrete Choice Models (DDCMs) has reached some maturity. The history of the research program is well-established, and several surveys exists, see for example [8] and [2]. In addition to surveys, the most relevant papers are [7, 5, 6] for information on the *expected value function,* and [1, 2] for the *integrated value function* formulation. We briefly define relevant objects of interest, as their definitions will be used below. The problem we're trying to solve is

$$V(s) = \max_{a_0,...a_T} E\left[\sum_{t=0}^{T} \left(\beta^t U(a_t, s_t))\right) |s_0\right]$$

although we will consider the infinite horizon (stationary) dynamic programming formulation

$$V(s) = \max_a \left(U(a,s) + \beta E_{s'} V(s')|s\right) \tag{1}$$

for all purposes below. The choice set, $\mathcal{A}$, is discrete and finite with J elements. The first problem we face is, that the value function depends on the shocks, and that these are typically continuous and unbounded - and unobserved. Then, the usual theorems do not apply.

Below, we will use the the fact that the state s consists of the *observed* states x and the shocks ϵ, such that $s = (x, \epsilon)$. We will also assume Conditional Independence (CI), that is (x, ϵ) follows a (controlled) Markov process according to the following transition density

$$p(x', \epsilon'|x, \epsilon, i) = q(\epsilon'|x')p(x'|x, i).$$

The shocks enter in a specific way

$$U(s, a) = u(x, a) + \epsilon,$$

that is they are additive.

1.1 Expected Value Function (EV)

The expected value formulation follows [5] and exploits the results the author presented on the the continuity, differentiability, and contraction properties of the expected value function in what [2] called Rust models. [6] showed that the results extend beyond the restrictions in the original formulation, but we will focus on *Rust models* below. The expected value function is *defined* as

$$EV_a(x) \equiv E_{s'}\left(V(s')|s, a\right) = E_{s'}\left(V(x', \epsilon')|s\right)$$

and leads to the following set of expected Bellman equations

$$EV_a(x) = E_{s'}\left(\max_j \left[U(j, x', \epsilon') + \beta EV_j(x')\right]|s, a\right), \quad \forall x \in \mathcal{X}, \forall a \in \mathcal{A}$$
$$EV_a(x) = \Gamma_a(EV, x).$$

It solves two problems of handling $V(s)$ directly: the expected Bellman operator is a contraction even if we have an unbounded (additive) component in U, and we do not have to find the solution at each ϵ, only each x. If we assume extreme value type I shocks, we can further simplify the expression:

$$\Gamma_a(EV, x) = E_{x'}\left(\log\left[\sum_{j\in\mathcal{A}}\exp\left\{u(x, j) + \beta EV(x', j)\right\}\right]|x, a\right).$$

We got rid of the numerical integration over shocks, and the maximization step was replaced with the log-sum-exp calculation. The problem is now almost a trivial exercise in root finding once we realize that Newton's method applied to $\Gamma_{EV}(EV) - EV = 0$ will give us convergence to some ε tolerance level in few iterations.

1.1.1 Discrete $\mathcal{X}$

If we assume that $\mathcal{X}$ is discrete and finite, we get

$$\Gamma(EV, a) = F(a)\left(\log\left[\sum_{j\in\mathcal{A}}\exp\left\{u(j) + \beta EV(j)\right\}\right]\right)$$

where $F(a)$ is the conditional transition *matrix* of dimension $|\mathcal{X}| \times |\mathcal{X}|$, log and exp are applied element-wise, and $EV(j)$, $u(j)$ are (stacked over x) vectors of length $|\mathcal{X}|$.

1.2 Integrated Value Function (W)

We saw that [5] took the joint expectation over s to get rid of two problems: unboundedness of U, and ϵ as an explicit state variable. Let us now consider an way of simplifying the problem. We consider the formulation in [1]. They consider the *integrated* value function as the object of analysis, defined as

$$W(x) \equiv E_{\epsilon'}\left[V(x, \epsilon')|\epsilon\right]$$

leading to the integrated Bellman equations

$$W(x) = E_{\epsilon'}\left[\max_j\left\{U(j, x, \epsilon') + \beta E_{x'|x,j}W(x')\right\}|\epsilon\right], \quad \forall x \in \mathcal{X}$$

$$W(x) = \Lambda(W, x)$$

which will solve the same problem as Γ. Again, we get the simplification using extreme value type I shocks that

$$\Lambda(W, x) = \log\left[\sum_{j\in\mathcal{A}}\exp\left\{u(x, j) + \beta E_{x'}(W(x'), |x, j)\right\}\right]$$

Since we know that Γ is continuous and once differentiable, then so will Λ be, as it is simply a log to the sum of exponentials of the sum of a continuous function (utility) and the EVs. They are also both contraction mappings with the same Lipschitz constant (β).

1.2.1 Discrete $\mathcal{X}$

Again, discrete and finite $\mathcal{X}$ simplifies matters, and we get

$$\Lambda(W) = \log\left[\sum_{j\in\mathcal{A}}\exp\left\{u(j) + \beta F(a)W\right\}\right].$$

Dimensions are as before, and all terms are stacked over x's.

2 Solution methods

Since the comparison here is motivated by the solution part of the nested fixed point algorithm in [7], we'll focus on two methods: value function iterations and Newton steps for fixed point problems. We consider discrete state spaces, although the lessons apply to the continuous state case as well.

First consider the cost of **value function iterations** (VFIs). It is the same for EV and V. Consider again the expressions from above

$$\Gamma(EV, a) = F(a) \left(\log \left[\sum_{j \in \mathcal{A}} \exp \{u(j) + \beta EV(j)\} \right] \right) \quad \forall a \in \mathcal{A}$$

$$\Lambda(W) = \log \left[\sum_{j \in \mathcal{A}} \exp \{u(j) + \beta F(a)W\} \right].$$

as long as we only calculate the log-sum-exp expression in Γ once per major VFI iteration, they both require J matrix-vector multiplications, J scalar-vector products, J vector-vector additions, J exp calls over vectors and one log call over a vector. This means that really is no reason to choose one over the other.

For **Newton's method** the computational costs differ. Define the conditional choice probabilities as

$$P(a, x) = Pr \left(a = \arg \max_{j \in \mathcal{A}} \{u(j, x) + \epsilon(j) + \beta E_{x'}(W(x')|x, j)\} \right)$$

$$= \frac{\exp \left(u(a, x) + \beta E_{x'}(W(x'|x, a)) \right)}{\sum_{j \in \mathcal{A}} \exp \left(u(j, x) + \beta E_{x'}(W(x)|x, j) \right)}$$

where the last line follows from the conditional independence and i.i.d. extreme value type I assumptions from above. Then, stack the Bellman equations

$$W = \Gamma(W),$$
$$EV = \{\Gamma(EV, j)\}_{j \in \mathcal{A}}.$$

Rearranging, we get

$$W - \Gamma(W) = (I - \Gamma)(W) = 0,$$
$$EV - \Lambda(EV = (I - \Lambda)(EV) = 0,$$

where I's denote the multiplicative identity, and 0 is the zero element in the Banach spaces where our value functions live. When we say we are using Newton's method, it means applying Newton's method to these equations by use of an inverse function theorem

$$W_{k+1} \leftarrow W_k + (I - \Lambda')^{-1}(I - \Lambda)W_k$$
$$EV_{k+1} \leftarrow EV_k + (I - \Gamma')^{-1}(I - \Gamma)EV_k$$

Looking at the discrete versions of the formulæ, and calculating the partial derivative state-by-state, we see it is β times the controlled transition. That is, for W we get

$$\Lambda' = \left(\sum_{i=1}^{J} \beta P(i) * F(i) \right)$$

For the EV, the derivative has more elements, and can be calculated as (defining P as the stacked conditional choice probabilities $\{P(j)\}_{j \in \mathcal{A}}$:

$$\Gamma' = \beta P *^{\top} \begin{pmatrix} F(1) & F(1) & \cdots & F(1) \\ F(2) & F(2) & \cdots & F(2) \\ \vdots & \vdots & \ddots & \vdots \\ F(J) & F(J) & \cdots & F(J) \end{pmatrix}$$

k	W	EV
1	2.244×10^1	2.244×10^1
2	1.666×10^1	1.666×10^1
3	1.292×10^1	1.292×10^1
4	1.091×10^1	1.091×10^1
5	5.535×10^0	5.531×10^0
6	2.883×10^0	2.879×10^0
7	3.635×10^{-1}	0.363×10^{-1}
8	6.553×10^{-3}	0.007×10^{-3}
9	1.378×10^{-6}	1.362×10^{-6}
10	7.105×10^{-14}	6.928×10^{-14}

Table 1: Convergence of Newton steps in Rust's model ($|\mathcal{X}| = 1000$) measured in change in EV for both cases.

where $*^\top$ is the Hadamard product between the vector P and the matrix, but applied down through the rows instead of across the columns as is usually done, such that

$$p *^\top M = (p * M^\top)^\top$$

for a vector p and matrix M. Here we have the problem of using the EV formulation: if we want to use Newton's method (and we really do), then this is now a $J|\mathcal{X}| \times J|\mathcal{X}|$ problem, or J^2 as large as the integrated value function problem. Note, that in regenerative models it is possible to use a trick in the style of the original [7]. However, it does require a little care when specifying the transitions, and indeed there is a mistake in the original application, see the appendix for the details.

2.1 Mathematical Programming with Equilibrium Constraints

The solution of the types of model we've considered here, is typically used in NFXP to estimate parameters in a given model based on some data we observe. This is quite naturally formulated as a constrained maximum likelihood problem as mentioned in [9], but NFXP essentially substitutes the constraints into the likelihood function since the model can be efficiently solved using the methods presented here. The MPEC (Mathematical Programming with Equilibrium Constraints) method in [10] basically keeps the restrictions on the feasible EV's (the ones that solve $EV = \Gamma(EV)$) in the formulation as equality constraints. They then solve the constrained maximum likelihood problem using an interior point solver. Since each point in the state space has a constraint for each choice, $EV_j(x_k) = \Gamma(EV, j, x_k)$, the considerations mentioned above for Newton's method also applies here. A much more concise specification is available, by simply considering constraints of the form $W(x_k) = \Lambda(W, x_k)$ instead.

3 Numerical results

We present some numerical evidence to support the claims above, and to assess the practical importance of them. First, convergence is the same in the two approaches. Consider Table 1 for a sequence of the ∞-norm of the differences between the current and previous iterates. For a given k, we see the similar improvements. What about differences in run-times? In Table 2, we see the time to perform a Newton step for the EV formulation relative to the W formulation. We solve Rust's model, and what we call Hitsch's model. The latter is a storable goods demand model. For a similar peer reviewed model see [3]. This simpler version is due to a presentation by Günter J. Hitsch. [1] The W formulation is faster as expected. We also report the time to calculate the Newton step and add it to the existing iterate. The "total" time also includes calculating choice probabilities, the derivative of the Bellman operators, and so on.

---

[1]See the presentation titled "Single Agent Dynamics: Dynamic Discrete Choice" by Günter J. Hitsch, The University of Chicago Booth School of Business, 2013.

Rust's model ($J = 2$)						
$\|\mathcal{X}\|$	10	100	200	300	500	800
Total relative time (EV/W)	3.94	2.80	3.64	3.93	4.26	7.16
Only $x_{k-1} + (I - M)\backslash(x_k - x_{k-1})$	3.90	3.56	4.78	4.94	5.47	7.24

Hitsch's model ($J = 3$)						
$\|\mathcal{X}\|$	12	102	202	302	502	802
Total relative time (EV/W)	4.45	5.07	7.26	10.90	9.60	13.49
Only $x_{k-1} + (I - M)\backslash(x_k - x_{k-1})$	3.93	8.40	13.10	16.70	15.84	16.56

Table 2: Relative run-times of Newton-steps. Times are given as EV formulation relative to W formulation, both total time and the time it takes to calculate the step and take it.

4 Conclusion

[7, 5] introduced simplifying assumptions that facilitated solution and estimation of empirically models of dynamic, discrete decision making. The theory and analysis was based on the expected value function. As shown, this is *not* efficient if Newton's method is applied to solve the system. The same information contained in the $EV(a)$'s can be summarized by the lower dimensional W. If successive approximations are used, there is no advantage, but if higher order methods are applied, it is inefficient to use EV over W. This also applies to the MPEC approach in [10], which will have more constraints than necessary. The performance gap increase as the number of choices increases, so especially multinomial problem is many discrete choices should be solved with W instead of EV.

A Appendix: Rust's Fréchet derivative

In [7] we have a binary model, so the expected value function can be separated into $EV_1(x)$ and $EV_2(x)$ in the notation of this note. In the original paper, a convenient observation is made:

$$EV_2(x) = EV_2(0) = EV_1(0)$$

that is, once Harold Zurcher replaces the engine it is as if the bus is brand new and replacement was not chosen. This basically means that we can fully represent the solution of the model using $EV_1(x)$, and then the Newton steps simplify to (writing $EV \equiv EV_1$)

$$EV_{k+1} \leftarrow EV_k + (I - \beta(P(1) *^\top F(1) + P(2) *^\top F(2))^{-1}(I - \Gamma)EV_k$$

This is a substantial dimensionality reduction, and eases the computational effort as a result. If there are $|\mathcal{X}|$ discrete states, then Γ' is only of size $|\mathcal{X}| \times |\mathcal{X}|$ instead of $2|\mathcal{X}| \times 2|\mathcal{X}|$. This is computationally very convenient, and it contributes to the fast run times in [4].

However, there is a slight logical disconnect between the way EV is reduced to effectively be of size $|\mathcal{X}|$ rather than $2|\mathcal{X}|$, and the way the Fréchet derivative of the Bellman operator is derived to be Rust (2000, p. 25):

$$P(1) *^\top F(1) + \left[P(2) *^\top \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \ldots & 0 \end{pmatrix} \right]$$

where (choosing a small p vector for demonstration purposes)

$$F(1) = \begin{pmatrix} p_1 & p_2 & p_3 & 0 & 0 \\ 0 & p_1 & p_2 & p_3 & 0 \\ 0 & 0 & p_1 & p_2 & p_3 \\ 0 & 0 & 0 & p_1 & p_2 + p_3 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The interpretation here is, that if you replace, you will reset your odometer, but you will have to keep the bus in the garage for a month. The problem is, that if that is the modeling choice, and the controlled process is as above, then generally $EV_1(x)$ is *not* $EV_2(0)$, as is clearly seen:

$$EV_1(x) = p_1 W(x) + p_2 W(x + \Delta x) + p_3 W(x + 2\Delta x)$$
$$EV_2(x) = 1 * W(0)$$
$$= EV_2(0)$$

where Δx is the change in x from one bin to the following. Now, it *is* true that EV_2 is constant, but as $W(x)$ is clearly going to be decreasing due to rising maintenance costs, we have that $EV_1(x) < EV_2(x)$ as long as $p_1 < 1$. This is unfortunate, as this increases the cost of the Newton steps substantially, if we want to keep the interpretation of the transitions. The fix is easy, just specify $F(2)$ as

$$\begin{pmatrix} p_1 & p_2 & p_3 & 0 & \ldots & 0 \\ p_1 & p_2 & p_3 & 0 & \ldots & 0 \\ p_1 & p_2 & p_3 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_1 & p_2 & p_3 & 0 & \ldots & 0 \end{pmatrix}$$

Then it is actually true, that, if we enter a period where the engine was replaced last period, it is the same situation as entering a period with a bus that did not have an engine replacement last period, but didn't transition from the first to the second state either.

References

[1] Victor Aguirregabiria and Pedro Mira. Swapping the nested fixed point algorithm: A class of estimators for discrete markov decision models. *Econometrica*, 70(4):1519–1543, 2002.

[2] Victor Aguirregabiria and Pedro Mira. Dynamic discrete choice structural models: A survey. *Journal of Econometrics*, 156(1):38–67, 2010.

[3] Igal Hendel and Aviv Nevo. Measuring the implications of sales and consumer inventory behavior. *Econometrica*, 74(6):1637–1673, 2006.

[4] Fedor Iskhakov, Jinhyuk Lee, John Rust, Bertel Schjerning, and Kyoungwon Seo. Comment on "constrained optimization approaches to estimation of structural models". *Econometrica*, 84(1):365–370, 2016.

[5] Rust John. Maximum likelihood estimation of discrete control processes. *SIAM Journal on Control and Optimization*, 26(5):1006–1024, 1988.

[6] Andriy Norets. Continuity and differentiability of expected value functions in dynamic discrete choice models. *Quantitative economics*, 1(2):305–322, 2010.

[7] John Rust. Optimal replacement of gmc bus engines: An empirical model of harold zurcher. *Econometrica: Journal of the Econometric Society*, pages 999–1033, 1987.

[8] John Rust. Structural estimation of markov decision processes. *Handbook of econometrics*, 4:3081–3143, 1994.

[9] John Rust. Nested fixed point algorithm documentation manual. *Unpublished Manuscript. Version*, 6, 2000.

[10] Che-Lin Su and Kenneth L. Judd. Constrained optimization approaches to estimation of structural models. *Econometrica*, 80(5):2213–2230, 2012.