1 Model

Writing the model in scaled form (note I'm dropping the tilde's – all appropriate variables implicitly have them)

$$j(k,x,v) = \max_{\chi \equiv k'g'} \left[(1-\beta)\left\{ y(k) + (1-\delta)k - \chi \right\}^{\rho} + \beta E_t \left\{ g'^{\alpha} J'(\chi/g', x', v')^{\alpha} \right\}^{\frac{\rho}{\alpha}} \right]^{1/\rho}$$

$$\begin{aligned} s.t. \quad & x' = Ax + Bv^{1/2}\epsilon_1' \\ & v' = (1-\varphi_v)\bar{v} + \varphi v + \tau \epsilon_2' \end{aligned}$$

Then the FONC wrt to χ is

$$\frac{\partial j_t}{\partial \chi_t} = \frac{1}{\rho} j_t^{1-\rho} \left((1-\beta)\rho c_t^{\rho-1} \frac{\partial c_t}{\partial \chi_t} + \beta \frac{\partial E_t \left[g_{t+1}^{\alpha}(x_{t+1}, v_{t+1}) J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]^{\frac{\rho}{\alpha}}}{\partial \chi_t} \right)$$

Then

$$\frac{\partial c_t}{\partial \chi_t} = -1$$

$$\frac{\partial E_t \left[g_{t+1}^{\alpha}(x_{t+1}, v_{t+1}) J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]^{\frac{\rho}{\alpha}}}{\partial \chi_t}$$

$$= \frac{\rho}{\alpha} E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]^{\frac{\rho-\alpha}{\alpha}} \frac{\partial E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]}{\partial \chi_t}$$

$$\frac{\partial E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]}{\partial \chi_t}$$

$$= E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} \alpha J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha-1} \underbrace{\frac{\partial J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})}{\partial k_{t+1}} \frac{1}{g_{t+1}}}_{\frac{\partial J_{t+1}}{\partial \chi_t}} \right]$$

Putting this together reveals

$$\begin{aligned} & (1-\beta)c_t^{\rho-1} \\ & = \beta E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha} \right]^{\frac{\rho-\alpha}{\alpha}} \\ & E_t \left[g_{t+1}(x_{t+1}, v_{t+1})^{\alpha} J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})^{\alpha-1} \frac{\partial J_{t+1}(\chi_t/g_{t+1}, x_{t+1}, v_{t+1})}{\partial k_{t+1}} \frac{1}{g_{t+1}} \right] \end{aligned}$$

Now let a $\cdot'$ stand for $\cdot_{t+1}$ and re-write this condition as:

$$(1-\beta)c(k,k')^{\rho-1} = \beta\mu\left(g'J'(\chi/g',x',v')\right)^{\rho-\alpha} E_t\left[g'^{\alpha-1}J'(\chi_t/g',x',v')^{\alpha-1}\frac{\partial J'(\chi_t/g',x',v')}{\partial k'}\right]. \tag{1}$$

To deal with that last derivative in there we need an envelope condition. This condition is

$$\frac{\partial J(k,x,v)}{\partial k} = \frac{1}{\rho}J(k,x,v)^{1-\rho}\rho(1-\beta)c^{\rho-1}(y^{1-\nu}\omega k+1-\delta). \tag{2}$$
$$= J(k,x,v)^{1-\rho}(1-\beta)c^{\rho-1}(y(k)^{1-\nu}\omega k+1-\delta) \tag{3}$$

Iterating (2) forward one period and combining with (1) yields (note I write just μ instead of $\mu(g'J'(\chi/g',x',v'))$, divide through by the $(1-\beta)$ on both sides, and combine terms where possible)

$$c(k,\chi/g')^{\rho-1} = \beta\mu^{\rho-\alpha}E_t\left[g'^{\alpha-1}J'(\chi_t/g',x',v')^{\alpha-\rho}c'(\chi/g',k'')^{\rho-1}(y'(\chi/g')^{1-\nu}\omega\chi/g'+1-\delta)\right]. \tag{4}$$

We will work with the log of (4):

$$(\rho-1)\log c(k,\chi/g') = $$
$$\log\beta+(\rho-\alpha)\log\mu+\log E_t\left[g'^{\alpha-1}J'(\chi_t/g',x',v')^{\alpha-\rho}c'(\chi/g',k'')^{\rho-1}(y'(\chi/g')^{1-\nu}\omega\chi/g'+1-\delta)\right]. \tag{5}$$

Simplified version without showing what things are functions of

$$(\rho-1)\log c = \log\beta+(\rho-\alpha)\log\mu+\log E_t\left[g'^{\alpha-1}J'^{\alpha-\rho}c'^{\rho-1}(y'^{1-\nu}\omega\chi/g'+1-\delta)\right].$$

2 ECM

We now describe a version of the envelope condition method that can be applied to our model. First we derive some equations. Starting from (2), we can solve for c:

$$c = \left(\frac{\frac{\partial J}{\partial k}}{J^{1-\rho}(1-\beta)(y^{1-\nu}\omega k+1-\delta)}\right)^{\frac{1}{\rho-1}}. \tag{6}$$

We can then use the budget constraint to solve for k':

$$k' = \frac{1}{g'}\left(y + (1-\delta)k - \left(\frac{\frac{\partial J}{\partial k}}{J^{1-\rho}(1-\beta)(y^{1-\nu}\omega k + 1 - \delta)}\right)^{\frac{1}{\rho-1}}\right) \tag{7}$$

Finally, we can combine (1) with (2) to derive an update rule for $\frac{\partial J}{\partial k}$:

$$\frac{\partial J}{\partial k} = \beta J^{1-\rho}\mu^{\rho-\alpha}(y^{1-\nu}\omega k + 1 - \delta)E_t\left[g'^{\alpha-1}J'^{\alpha-1}\frac{\partial J'}{\partial k'}\right] \tag{8}$$

An algorithm to apply the method is as follows

1. Initialization:
    - Choose parameter values
    - Choose uni-variate grids for k, x, and v
    - Expand the grids into a $M := n_k * n_x * n_v \times 3$ matrix Γ
    - Choose an integration method and construct a $N \times 2$ matrix of integration nodes ϵ as well as a N element probability vector Π.
    - Let x_m and v_m represent the second and third columns of the mth row of Γ. Then let ϵ_{1n} and ϵ_{2n} be the first and second columns of the nth row of ϵ. Then, for $m = 1 \ldots M$ and $n = 1 \ldots N$, compute $x'_{mn} = Ax_m + Bv_m^{1/2}\epsilon_{1n}$ and $v'_{mn} = (1-\varphi)\bar{v} + \varphi v_m + \tau\epsilon_{2,n}$.
    - For $m = 1 \ldots M$ and $n = 1 \ldots N$ compute $g'_{mn} = \bar{g}exp(x'_{mn})$.
    - Choose a functional form for representing the derivative of the value function $J_k(k, x, v; b)$ and make an initial guess for the coefficient vector β
2. At iteration i for $m = 1 \ldots M$
    (a) Use (6) to solve for c_m
    (b) Use c_m and (7) to solve for k'_{mn} for each $n = 1 \ldots m$.
    (c) Use (8) to solve for the derivative of J on grid. Call the left-hand-side d_m
3. Run a regression and convex combination to update the coefficient vector b
    - This means compute $\hat{b} = \hat{V}(k_m, x_m, v_m; b)\backslash d_m$, where $\hat{V}(k_m, x_m, v_m; b)$ is the $M \times nf$ basis matrix for the chosen form of approximating function that has nf terms.
    - With $\hat{b}$ in hand we then form a convex combination between $\hat{b}$ and the existing b to form the new coefficient vector: $b = (1-\xi)\hat{b} + \xi b$

4. Check convergence by computing $err = \frac{1}{M}\sum_{m=1}^{M}\left|\frac{c_m^{(i+1)}-c_m^{(i)}}{c_m^{(i)}}\right|$, where $c_m^{(i)}$ is the c_m computed in step 2.a on the ith iteration of the algorithm. If $err <$ some predefined tolerance, stop. Otherwise start over at step 2 with the updated coefficient vector.