# Numerical Methods
# Lecture 3: Perturbation Methods

Zachary R. Stangebye

University of Notre Dame

Fall 2017

# Traditional Methods

- Before large-scale computing resources available, economists made do with what they had
- Popular method: Linearizing/log-linearizing
- Idea
  1. Boil complicated non-linear relationships down to something tractable
  2. Use linear algebra (requires little numerical power)
- Log-linearizing
  1. Often yields easier solution than linearizing
  2. Gets rid of units; can talk about %-changes
  3. Not quite right...
     - Economists eventually started accepting the log-linearized model as the relevant model instead of the full, microfounded one

# Log-Linearizing

- ‘Oldest’ trick in the books
- A bit ad-hoc when it was first implemented; more sophisticated techniques developed
- Basic idea
  1. Take log of all equilibrium conditions
  2. Linearize (Taylor-expand) system around a point (steady state)
  3. Manipulate until all variables of interest are percentage deviations

# Why?

- Suppose we have a nonlinear relationship

$$f(x) = \frac{g(x)h(x)}{\eta(x)}$$

- Taking logs yields

$$\log f(x) = \log g(x) + \log h(x) - \log \eta(x)$$

- Piecewise (bit-by-bit) first-order approximation around $\bar{x}$

$$\log f(\bar{x}) + \frac{f'(\bar{x})}{f(\bar{x})}[x - \bar{x}] \approx \log g(\bar{x}) + \frac{g'(\bar{x})}{g(\bar{x})}[x - \bar{x}] + \dots$$

# Why?

- Notice that the constant terms all cancel. Divide remaining expressions by $\bar{x}$...

$$\frac{f'(\bar{x})}{f(\bar{x})}\hat{x} \approx \frac{g'(\bar{x})}{g(\bar{x})}\hat{x} + \frac{h'(\bar{x})}{h(\bar{x})}\hat{x} - \frac{\eta'(\bar{x})}{\eta(\bar{x})}\hat{x}$$

  where $\hat{x} = \frac{x-\bar{x}}{\bar{x}}$ i.e. percent deviation of $x$ from steady state
- Linear in $\hat{x}$. Impose equality to approximate the system
- Notice that $\hat{x}$ is approximated around zero by construction

# Additive/Multivariate Systems

- What if $y_t = g(x_t) + z_t$?

$$\implies \log y_t = \log(g(x_t) + z_t)$$

$$\log(\bar{y}) + \frac{1}{\bar{y}}[y_t - \bar{y}] = \log(g(\bar{x}) + \bar{z}) + \frac{g'(\bar{x})[x_t - \bar{x}] + [z_t - \bar{z}]}{g(\bar{x}) + \bar{z}}$$

- Exploit $\bar{y} = g(\bar{x}) + \bar{z}$

$$\frac{1}{\bar{y}}[y_t - \bar{y}] = \frac{g'(\bar{x})}{\bar{y}}[x_t - \bar{x}] + \frac{1}{\bar{y}}[z_t - \bar{z}]$$

- Cannot divide whole thing by $\bar{x}$. Multiply and divide each term by SS component

$$\hat{y}_t = \frac{\bar{x}g'(\bar{x})}{\bar{y}}\hat{x}_t + \frac{\bar{z}}{\bar{y}}\hat{z}_t$$

# Example: NCG Model, CRRA-utility

- Two-variable dynamical system

$$c_t^{-\sigma} = \beta(\alpha k_{t+1}^{\alpha-1} + 1 - \delta)c_{t+1}^{-\sigma}$$

$$c_t + k_{t+1} = k_t^{\alpha} + (1-\delta)k_t$$

- Example worked out on board in class
- Notice why it's ad hoc: Technically the log $\beta$ term should go away...Doesn't happen when we do it 'bit-by-bit'

$$\hat{k}_{t+1} = [\alpha \bar{k}^{\alpha-1} + 1 - \delta]\hat{k}_t - \frac{\bar{c}}{\bar{k}}\hat{c}_t$$

$$\hat{c}_{t+1} = \hat{c}_t + \frac{\alpha(\alpha-1)\bar{k}^{\alpha}}{\sigma[\alpha \bar{k}^{\alpha-1} + 1 - \delta]}\hat{k}_{t+1}$$

# Stability I

- Want our approximation to be stationary around zero
- Express linear dynamical system as

$$\hat{x}_{t+1} = A\hat{x}_t$$

  for a matrix $A$
- Here, $\hat{x}_t = [k_t, c_t]'$ and

$$A = \begin{bmatrix} [\alpha\bar{k}^{\alpha-1} + 1 - \delta] & -\frac{\bar{c}}{\bar{k}} \\ \frac{\alpha(\alpha-1)\bar{k}^{\alpha}}{\sigma} & 1 - \frac{\alpha(\alpha-1)\bar{k}^{\alpha}}{\sigma[\alpha\bar{k}^{\alpha-1}+1-\delta]}\frac{\bar{c}}{\bar{k}} \end{bmatrix}$$

  Substituted in RC to get $\hat{k}_{t+1}$ out of Euler Equation

# Stability II

- Intuitively, it will be stable if we start from some $\hat{x}_t$ 'near' zero and it goes to zero over time i.e.

$$\lim_{n\to\infty} A^n \hat{x}_t = \mathbf{0}$$

- Think back to eigenvalues

$$Av = \gamma v$$

for some eigenvalue, $\gamma$, and eigenvector $v$

- If eigenvalues are distinct, then so are eigenvectors
  - $\implies$ Eigenvectors can span 2D-space

# Stability III

- Since eigenvectors span the space, we can write any $\hat{x}_t$ as a combination of the eigenvalues

$$\hat{x}_t = \alpha_{1,t}\nu_1 + \alpha_{2,t}\nu_2$$

for some constants $\alpha_{1,t}$ and $\alpha 2, t$

- Now

$$\begin{aligned} \lim_{n\to\infty} A^n \hat{x}_t &= \lim_{n\to\infty} A^n \left[\alpha_{1,t}\nu_1 + \alpha_{2,t}\nu_2\right] \\ &= \alpha_{1,t} \lim_{n\to\infty} A^n \nu_1 + \alpha_{2,t} \lim_{n\to\infty} A^n \nu_2 \\ &= \alpha_{1,t} \lim_{n\to\infty} \gamma_1^n \nu_1 + \alpha_{2,t} \lim_{n\to\infty} \gamma_2^n \nu_2 \end{aligned}$$

# Stability IV

- 3 possible cases
  1. $|\gamma_1|, |\gamma_2| < 1$ i.e. globally stable
  2. $|\gamma_1|, |\gamma_2| \geq 1$ i.e. globally unstable
  3. $|\gamma_i| < 1$ and $|\gamma_j| \geq 1$ i.e. saddle-path stable
- NCG model falls is saddle-path stable (one-dimensional stable manifold)
  - Given a $k_t$, only one $c_t$ will work
  - Will be a $c_t$ such that $\hat{x}_t$ implies $\alpha_{j,t} = 0$
    - i.e. in span of stable eigenvector (easy to compute)
- Sidenote: If $\gamma_i$ is complex, solution 'spirals' toward/away from steady state

# Adding Shocks

- The RBC model has a random component. RC deterministic, but EE given by

$$c_t^{-\sigma} = \beta E_t \left[ (\alpha e^{z_{t+1}} k_{t+1}^{\alpha-1} + 1 - \delta) c_{t+1}^{-\sigma} \right]$$

- Taking logs

$$-\sigma \log c_t = \log \beta + \log E_t \left[ (\alpha e^{z_{t+1}} k_{t+1}^{\alpha-1} + 1 - \delta) c_{t+1}^{-\sigma} \right]$$

but $\log(E[x]) \neq E[\log(x)]$...

- So people ignore this! (faux pas number two)
- Charge ahead and derive something like

$$-\sigma \hat{c}_t = -\sigma E_t \hat{c}_{t+1} + \gamma_1 \hat{k}_{t+1} + \gamma_2 E_t z_{t+1}$$

where $\gamma_1$ and $\gamma_2$ are known combinations of parameters

# LRE Models

- But even here is a problem:
  - Want something like

$$\begin{bmatrix} k_t \\ c_t \\ z_t \end{bmatrix} = A \begin{bmatrix} k_{t-1} \\ c_{t-1} \\ z_{t-1} \end{bmatrix} + B\epsilon_t$$

  - How to get rid of $E_t[c_{t+1}]$ terms?
- These models are often called **Linear Rational Expectations Models**
  - Sims (2002) proposed most common solution

# LRE Solution: Applicability

- Designed for models of form

$$\Gamma_0 y_t = \Gamma_1 y_{t-1} + C + \Psi z_t + \Pi \eta_t$$

- Some elements of $y_t$ are expectations of future elements

$$x_t = E_{t-1}[x_t] + \eta_t$$

1. States (including expected states): $y_t = [x_t, E_t x_{t+1}]'$
2. Vector of constants: $C$
3. Exogenous shock processes: $z_t$
4. Expectations errors: $\eta_t$ (not exogenous; part of solution)

# Procedure

- Assume either
  1. $\Gamma_0 = I_n$
  2. $\Gamma_0$ non-singular; pre-multiply everything through by it
- Work with system (with possible redefinition of RHS matrices)

$$y_t = \Gamma_1 y_{t-1} + C + \Psi z_t + \Pi \eta_t$$

- **Step 1:** Jordan Decomposition of Square Matrix

$$\Gamma_1 = P \Lambda P^{-1}$$

  1. $P$: Matrix of (generalized) eigenvectors of $\Gamma_1$
  2. $\Lambda$: Diagonal matrix of eigenvalues
     - Place 2 identical eigenvalues next to each other
     - $\Lambda_{i,i+1} = 1$ if $\Lambda_{i,i} = \Lambda_{i+1,i+1}$

# Procedure

- **Step 2:** Pre-multiply by $P^{-1}$
  - Define $w_t = P^{-1} y_t$

$$w_t = \Lambda w_{t-1} + P^{-1} C + P^{-1}(\Psi z_t + \Pi \eta_t)$$

- This system breaks into completely independent 'blocks' by eigenvalue, with a typical block

$$w_{j,t} = \begin{bmatrix} \lambda_j & 1 & 0 & \dots & 0 \\ 0 & \lambda_j & 1 & \dots & 0 \\ 0 & 0 & \dots & \dots & 0 \\ \dots & \dots & \dots & \lambda_j & 1 \\ 0 & \dots & 0 & 0 & \lambda_j \end{bmatrix} w_{j,t-1} + P^{j\cdot} C + P^{j\cdot}(\Psi z_t + \Pi \eta_t)$$

where $P^{j\cdot}$ is matrix of corresponding rows of $P^{-1}$

# Procedure

- Steady state solution will impose $z_t$ and $\eta_t$ equal zero

$$w_{j,t} = [I - \Lambda_j]^{-1} P^{j\cdot} C$$

- Notice a block-system is stationary iff $|\lambda_j| < 1$
- **Step 3:** Separate blocks into two groups: $S$ (stable) and $U$ (unstable)
  - In any solution, it must be the case that

$$w_{U,t} = [I - \Lambda_U]^{-1} P^{U\cdot} C$$

    for *all* $t$ (otherwise it would explode)
  - Implies shocks must satisfy (for any $t$)

$$P^{U\cdot}(\Psi z_t + \Pi \eta_t) = 0$$

# Procedure

- For a solution to exist, it must be that

$$\mathrm{span}(P^{U\cdot}\Psi) \subset \mathrm{span}(P^{U\cdot}\Pi)$$

  i.e. expectations shocks can always arise to offset exogenous shocks in unstable manifold (keeping it in steady state)
- For a solution to be unique, it must be that

$$\mathrm{span}(\Pi'(P^{S\cdot})') \subset \mathrm{span}(\Pi'(P^{U\cdot})')$$

  i.e. there must be only *one* such set of expectations shocks
- **Step 5:** If these conditions hold, then a matrix $\Phi$ exists such that

$$P^{S\cdot}\Pi\eta = \Phi P^{U\cdot}\Pi\eta$$

# Procedure

- To solve for $\Phi$...
- Run a 'regression' of $P^{U\cdot}\Pi$ on $P^{S\cdot}\Pi$
    - Residuals should be zero since it's in the span
- $y = X\beta \implies \beta = (X'X)^{-1}X'y$
    1. $\beta = \Phi'$
    2. $X = \Pi'(P^{U\cdot})'$
    3. $y = \Pi'(P^{S\cdot})'$

$$\Phi' = [P^{U\cdot}\Pi\Pi'(P^{U\cdot})']^{-1}P^{U\cdot}\Pi\Pi'(P^{S\cdot})'$$

- If $X'X$ is singular, either
    1. No solution exists
    2. $\Phi$ can be trivially found via inspection
- Often (2) is the case in simple, one-shock models

# Procedure

- **Step 6:** Use $\Phi$-matrix to get rid of expectation shocks

$$\begin{bmatrix} w_{S,t} \\ w_{U,t} \end{bmatrix} = \begin{bmatrix} \Lambda_S \\ 0 \end{bmatrix} w_{S,t-1} + \begin{bmatrix} P^{S\cdot} C \\ [I - \Lambda_U]^{-1} P^{U\cdot} C \end{bmatrix} + \begin{bmatrix} I & -\Phi \\ 0 & 0 \end{bmatrix} P^{-1} \Psi z_t$$

- Now use $y_t = P w_t$ to go back!

$$y_t = \Theta_1 y_{t-1} + \Theta_c C + \Theta_z z_t$$

1. $\Theta_1 = P_{\cdot S} \Lambda_S P^{S\cdot}$
2. $\Theta_C = P_{\cdot S} P^{S\cdot} + P_{\cdot U} [I - \Lambda_U]^{-1} P^{U\cdot}$
3. $\Theta_z = [P_{\cdot S} P^{S\cdot} - P_{\cdot S} \Phi P^{U\cdot}] \Psi$

- Expectation error shocks gone!

# Application

- To completely characterize mapping from initial conditions and $z$, we need to initialize expectation terms i.e.

$$E_t y_{t+s} = \Theta_1^s y_t + (I - \Theta_1^{s+1})(I - \Theta_1)^{-1}\Theta_c C$$

- Can simulate model, analyze partial derivatives, etc.
- Impulse response functions can be computed in closed-form

$$y_{t+s} - E_t y_{t+s} = \sum_{\nu=0}^{s-1} \Theta_1^{\nu}\Theta_z z_{t+s-\nu}$$

# Example: Stochastic NCG Model

- Log-linearized equations

$$\hat{k}_{t+1} = [\alpha \bar{k}^{\alpha-1} + 1 - \delta]\hat{k}_t + \bar{k}^{\alpha} z_t - \frac{\bar{c}}{\bar{k}}\hat{c}_t$$

$$-\sigma \hat{c}_t = -\sigma E_t \hat{c}_{t+1} + \left[\frac{\alpha(\alpha-1)\bar{k}^{\alpha-1}}{\alpha \bar{k}^{\alpha-1} + 1 - \delta}\right]\hat{k}_{t+1} + \left[\frac{\alpha \bar{k}^{\alpha-1}}{\alpha \bar{k}^{\alpha-1} + 1 - \delta}\right]\underbrace{E_t z_{t+1}}_{=\rho z_t}$$

- Substitute out $\hat{k}_{t+1}$ into EE

$$\hat{k}_{t+1} = [\alpha \bar{k}^{\alpha-1} + 1 - \delta]\hat{k}_t + \bar{k}^{\alpha} z_t - \frac{\bar{c}}{\bar{k}}\hat{c}_t$$

$$\left[\frac{\alpha(\alpha-1)\bar{k}^{\alpha-1}}{\alpha \bar{k}^{\alpha-1} + 1 - \delta}\frac{\bar{c}}{\bar{k}} - \sigma\right]\hat{c}_t = -\sigma E_t \hat{c}_{t+1} + \left[\alpha(\alpha-1)\bar{k}^{\alpha-1}\right]\hat{k}_t$$

$$+ \left[\frac{\alpha(\alpha-1)\bar{k}^{2\alpha-1} + \rho\alpha \bar{k}^{\alpha-1}}{\alpha \bar{k}^{\alpha-1} + 1 - \delta}\right] z_t$$

# Example: Stochastic NCG Model

1. $y_t = [\hat{k}_t, \hat{c}_t, E_t[\hat{c}_{t+1}], z_t]$
2. $z_t = \epsilon_t$ (from AR(1))
3. $C = 0$
4. $\hat{c}_t = E_{t-1}[\hat{c}_t] + \eta_t$

- $z_t$ term in Euler Equation: $z_t = \rho z_{t-1} + \sigma_z \epsilon_t$
- Lag resource constraint
- Substitute RC into Euler Equation to eliminate $\hat{k}_{t+1}$

# Example: Stochastic NCG Model

$$\Gamma_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{\alpha(\alpha-1)\bar{k}^{\alpha-1}}{\sigma} & \left[\frac{\alpha(\alpha-1)\bar{k}^{\alpha-1}}{\sigma(\alpha\bar{k}^{\alpha-1}+1-\delta)}\frac{\bar{c}}{\bar{k}} - 1\right] & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Gamma_1 = \begin{bmatrix} \alpha\bar{k}^{\alpha-1} + 1 - \delta & -\frac{\bar{c}}{\bar{k}} & 0 & \bar{k}^{\alpha} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \left[\frac{\alpha(\alpha-1)\bar{k}^{2\alpha-1}+\rho\alpha\bar{k}^{\alpha-1}}{(\alpha\bar{k}^{\alpha-1}+1-\delta)\sigma}\right]\rho \\ 0 & 0 & 0 & \rho \end{bmatrix}$$

$$\Psi = \left[0, 0, \left[\frac{\alpha(\alpha-1)\bar{k}^{2\alpha-1}+\rho\alpha\bar{k}^{\alpha-1}}{(\alpha\bar{k}^{\alpha-1}+1-\delta)\sigma}\right]\sigma_z, \sigma_z\right]'$$

$$\Pi = [0, 1, 0, 0]'$$

# Idea of Perturbation

- Build approximate solutions to economy by starting from
  1. Exact solution of a particular case
  2. Solution of a nearby model whose solution we have access to
- Perturbation algorithms
  1. Taylor series approximation
  2. Around a deterministic steady state
  3. Using implicit-function theorems
- Pros
  1. Accurate around approximation point (in some cases reasonable global performance)
  2. Structure intuitive and easily interpretable
     - e.g. Second-order expansion includes term to correct for volatility of shocks
  3. Traditional linearization $\iff$ First-order perturbation
  4. Dynare makes it accessible

## Mathematical Structure

- Recall we wish to solve functional equation

$$\mathcal{H}(d) = \mathbf{0}$$

- If $d$ is an $n$-differentiable function, apply Taylor's Theorem. For any $n > 0$...

**Theorem**

$d(x) = \sum_{i=0}^{n-1} x^i \frac{d^i(0)}{i!} + \int_0^x \frac{(x-u)^{n-1}}{(n-1)!} d^n(u) du$

- Without remainder term, we get an approximation with a quantifiable error in a local range i.e. if $x$ is 'near' $\bar{x}$

$$d(x) \approx d_i(x; \bar{x}) = \sum_{k=0}^{i} (x - \bar{x})^k \frac{d^k(\bar{x})}{k!}$$

# Which Point?

- Generally restrict attention to *Dynamic, Stochastic, General Equilibrium* (DSGE) models
    - Basically (large) set of extensions of stochastic NCG model
    - Includes RBC model, New Keynesian model, BGG Financial accelerator model, etc.
- This class of models usually has a **steady state**
    - A deterministic point in the state space toward which the system converges over time (in the absence of shocks)

## Solving the Steady State

- Re-write $\mathcal{H}(d) = 0$ as

$$E_t[\mathcal{H}(y_t, \tilde{y}_{t+1}, x_t, \tilde{x}_{t+1})] = \mathbf{0}$$

  - $y_t$ is $n_y \times 1$ vector of controls (choice variables)
  - $x_t$ is $n_x \times 1$ vector of states
  - $n = n_y + n_x$ number of equilibrium conditions
- Partition states: $x_t = [x_{1t}'; x_{2t}']'$
  - $x_{1t}$: $(n_x - n_\epsilon) \times 1$ vector of endogenous states (capital, debt, etc.)
  - $x_{2t}$: $n_\epsilon \times 1$ vector of exogenous states (productivity, preference shocks, etc.)

# Solving the Steady State

- The steady state is defined as a pair of vectors $(\bar{x}, \bar{y})$ such that

$$\mathcal{H}(\bar{y}, \bar{y}, \bar{x}, \bar{x}) = \mathbf{0}$$

  - Solution can often be found analytically
  - Not necessarily the same as *stochastic steady state* $(\hat{y}, \hat{x})$

$$E_t \mathcal{H}(\hat{y}, \hat{y}, \hat{x}, \hat{x}) = \mathbf{0}$$

Stochastic steady state includes response to risk e.g. precautionary saving

# Exogenous Process and the Perturbation Parameter

- Exogenous process

$$x_{2,t+1} = \mathbf{C}(x_{2,t}) + \sigma \eta_\epsilon \epsilon'$$

- $\eta_\epsilon$ is covariance matrix
- Generally assume Hessian of $\mathbf{C}$ at $\bar{x}_2$ has eigenvalues in the unit circle (stationary)
- $\sigma \geq 0$ is the **perturbation parameter**
  - Defines what (exogenous states) we're approximating over
  - Set $\sigma = 0$ at steady state
  - Set $\sigma = 1$ in solution
- Restrict attention to cases in which only shocks are perturbed
  - $\sigma = 0$ implies a deterministic model

# Nonlinear Shocks

- Assumption that shocks enter linearly is wlog. If instead

$$x_{2,t} = \mathbf{D}(x_{2,t-1}, \sigma\eta_\epsilon\epsilon_t)$$

- Re-define state to be $\tilde{x}_{2,t} = [x_{2,t}', \epsilon_t']'$
  - Re-define $x_{2,t} = \tilde{\mathbf{D}}(\tilde{x}_{2,t-1}, \sigma\eta_\epsilon)$

$$\underbrace{\begin{bmatrix} x_{2,t} \\ \epsilon_{t+1} \end{bmatrix}}_{\tilde{x}_{2,t}} = \begin{bmatrix} \tilde{\mathbf{D}}(\tilde{x}_{2,t-1}, \sigma\eta_\epsilon) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \epsilon_{t+1} \end{bmatrix}$$

- Can accommodate wide variety of models (stochastic volatility, GARCH, etc.)

# Example: Stochastic NCG/RBC Model

- $y_t = c_t$, $(n_y = 1)$
- $x_t = [z_t, k_t]'$, $(n_x = 2,\ n_\epsilon = 1)$
- Function $\mathcal{H}$ written as two equations (replace $\sigma$ with $\gamma$ so as not to confuse it with perturbation parameter)

$$\textbf{(1)} \quad c_t^{-\gamma} - \beta(\alpha e^{z_{t+1}} k_{t+1}^{\alpha-1} + 1 - \delta) c_{t+1}^{-\gamma} = 0$$

$$\textbf{(2)} \quad c_t + k_{t+1} - e^{z_t} k_t^{\alpha} - (1-\delta) k_t = 0$$

- $x_{2,t} = z_t$. Perturb the shock in the AR(1) process

$$z_{t+1} = \rho z_t + \sigma \eta_\epsilon \epsilon'$$

- Recall steady state sets $\sigma = 0$

## Steady State

1. Law of motion implies $\bar{z} = 0$
2. Given this, EE implies

$$\bar{k} = \left(\frac{\alpha}{\rho_\beta + \delta}\right)^{\frac{1}{1-\alpha}}$$

   where $\rho_\beta = 1/\beta - 1$

3. The RC then tells us

$$\bar{c} = \left(\frac{\alpha}{\rho_\beta + \delta}\right)^{\frac{\alpha}{1-\alpha}} - \delta\left(\frac{\alpha}{\rho_\beta + \delta}\right)^{\frac{1}{1-\alpha}}$$

- Usually a line-by-line procedure can find the steady state
    - Occasionally, a non-linear solver must be used, but problem is generally low dimension

# Solution Form

- The (full) solution will be a pair of functions

$$\textbf{Policy Function:} \quad y = g(x;\sigma)$$

$$\textbf{Law of Motion:} \quad x' = h(x;\sigma) + \sigma\eta\epsilon'$$

where $g : \mathcal{R}^{n_x} \times \mathcal{R}^{+} \rightarrow \mathcal{R}^{n_y}$ and $h : \mathcal{R}^{n_x} \times \mathcal{R}^{+} \rightarrow \mathcal{R}^{n_x}$ and

$$\eta = \begin{bmatrix} \emptyset \\ \eta_\epsilon \end{bmatrix}$$

- Define

$$F(x;\sigma) = E_t\mathcal{H}(g(x;\sigma), g(h(x;\sigma) + \sigma\eta\epsilon';\sigma), x, h(x;\sigma) + \sigma\eta\epsilon')$$

Notice $F : \mathcal{R}^{n_x+1} \rightarrow \mathcal{R}^n$

# Perturbing

- Notice that, by definition, $F(x;\sigma) = 0$ for any $x$ and $\sigma$
  - Thus, all derivatives of $F$ must be zero as well

$$F_{x_i^k \sigma^j}(x;\sigma) = 0, \quad \forall x, \sigma, i, k, j$$

- Suppose we want a first-order approximation of solution around steady state

$$g(x;\sigma) \approx \bar{y} + g_x(\bar{x};0)(x - \bar{x}) + g_\sigma(\bar{x};0)\sigma$$

$$h(x;\sigma) \approx \bar{x} + h_x(\bar{x};0)(x - \bar{x}) + h_\sigma(\bar{x};0)\sigma$$

- Need to find four derivative coefficients; $n \times (n_x + 1)$ distinct terms

## Perturbing

- These terms can be pinned down with

$$F_{x_i}(\bar{x};0) = 0 \quad \forall i$$

which provides $n \times n_x$ equations, and

$$F_{\sigma}(\bar{x};0) = 0$$

which gives $n$ more

- Things are about to get messy...let's prepare...

# Tensor Notation

1. A simple way to express loads of terms
2. Gets rid of $\sum$ and $\partial$ signs
3. Points at which derivative is evaluated are also dropped for simplicity

- The derivative of $\mathcal{H}$ with respect to $y$ is an $n \times n_y$ matrix

$$[\mathcal{H}_y]^i_\alpha$$

is the $i$th row and $\alpha$th column of this matrix

# Tensor Notation

- When a sub-index reappears as a superindex in the next term, we are omitting a sum
- For instance, if we want to express how $x_j$ influences $y_i'$, we use the chain rule to write

$$[\mathcal{H}_y]^i_\alpha [g_x]^\alpha_\beta [h_x]^\beta_j = \sum_{\alpha=1}^{n_y} \sum_{\beta=1}^{n_x} \frac{\partial \mathcal{H}^i}{\partial y^\alpha} \frac{\partial g^\alpha}{\partial x^\beta} \frac{\partial h^\beta}{\partial x^j}$$

- For higher derivatives...if we have $[\mathcal{H}_{y'y'}]^i_{\alpha\gamma}$
  - This is a 3-dimensional array with $n$ rows, $n_y$ columns, and $n_y$ pages
  - $i$th row, $\alpha$th column, and $\gamma$th page

# Perturbation Again

- With tensor notation, we can write

$$[F_x(\bar{x};0)]^i_j = [\mathcal{H}_{y'}]^i_\alpha [g_x]^\alpha_\beta [h_x]^\beta_j + [\mathcal{H}_y]^i_\alpha [g_x]^\alpha_j + [\mathcal{H}_{x'}]^i_\beta [h_x]^\beta_j + [\mathcal{H}_x]^i_j = \mathbf{0}$$

- There's a lot of these derivatives, but they're all known!
- Notice it's quadratic since we have $g_x h_x$ terms
    - Comes from impact of variables today affecting variables tomorrow through law of motion *and* policy function

## Perturbing the Shocks

- Get our last $n$ equations from

$$[F_\sigma(\bar{x};0)]^i = E_t\Big[[\mathcal{H}_{y'}]^i_\alpha[g_x]^\alpha_\beta[h_\sigma]^\beta + [\mathcal{H}_{y'}]^i_\alpha[g_x]^\alpha_\beta[\eta]^\beta_\phi[\epsilon']^\phi + [\mathcal{H}_{y'}]^i_\alpha[g_\sigma]^\alpha$$

$$+[\mathcal{H}_y]^i_\alpha[g_\sigma]^\alpha + [\mathcal{H}_{x'}]^i_\beta[h_\sigma]^\beta + [\mathcal{H}_{x'}]^i_\beta[\eta]^\beta_\phi[\epsilon']^\phi\Big]$$

- Since shocks are mean zero and linear, they drop out easily

$$\mathbf{0} = [\mathcal{H}_{y'}]^i_\alpha[g_x]^\alpha_\beta[h_\sigma]^\beta + [\mathcal{H}_{y'}]^i_\alpha[g_\sigma]^\alpha + [\mathcal{H}_y]^i_\alpha[g_\sigma]^\alpha + [\mathcal{H}_{x'}]^i_\beta[h_\sigma]^\beta$$

# Certainty Equivalence

- Suppose we have solved the first system for $(g_x, g_y, h_x, h_y)$
- Plug in with already known $\mathcal{H}$ terms to get that system is
  1. Linear in $h_\sigma$, $g_\sigma$
  2. Homogeneous in $h_\sigma$, $g_\sigma$
- Thus, if a unique solution exists, it must be that

$$g_\sigma = 0$$

$$h_\sigma = 0$$

- Solution

$$g(x;\sigma) \approx \bar{y} + g_x(\bar{x};0)(x-\bar{x})'$$

$$h(x;\sigma) \approx \bar{x} + h_x(\bar{x};0)(x-\bar{x})'$$

# Certainty Equivalence

- Solution exhibits **Certainty Equivalence** i.e. first-order approximation identical to first-order approximation of the same model under perfect foresight
- This will *not* be the case for perturbations higher than 1
- Intuitive in context of DSGE model
  - Precautionary motive kicks in when $u'''(\cdot) \neq 0$
  - Euler equation has $u'(\cdot)$, so approximation contains $u''(\cdot)$ terms
    - But no $u'''(\cdot)$ terms!
- Agents respond to *current realizations of shocks*, but take no alternate action *ex ante* because of the presence of shocks

# Solving Quadratic Systems

- Square matrix $P$ such that

$$AP^2 - BP - C = 0$$

1. Define two new matrices ($2n \times 2n$)

$$D = \begin{bmatrix} A & \mathbf{0}_n \\ \mathbf{0}_n & I_n \end{bmatrix}, \quad F = \begin{bmatrix} B & C \\ I_n & \mathbf{0}_n \end{bmatrix}$$

2. Find generalized Schur decomposition $(QZ)$ of $D$ and $F$ i.e.

$$Q'\Sigma Z = D$$

$$Q'\Phi Z = F$$

Note: Multiple $QZ$ decompositions based on sorting of diagonals of $\Sigma$ and $\Phi$

# Solving Quadratic Systems

3. Partition $Z$ into

$$Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix}$$

4. Solution is just $P = -Z_{21}^{-1} Z_{22}$
   - Multiple $QZ$ decompositions $\implies$ Multiple solutions
   - Generally only one will be stable in the eigenvalue $< 1$ sense
   - This will be the one such that $|\phi_{ii}/\sigma_{ii}|$ are in increasing order as we move down the diagonal

# Simplifying a Bit

- Quadratic systems can take a while to solve
- Can partition to speed things up a bit
    1. Separate conditions related to $g_x(\bar{x};0)$ and $h_x(\bar{x};0)$ related to the endogenous state variables
    2. Solve for these (solution exists)
    3. Plug in solution to remaining conditions to get response of exogenous states/stochastic shocks from *linear system*

- Suppose $n_x = 20$, $n_y = 1$, and $n_\epsilon = 5$
    1. All-at-once approach: Quadratic system with 420 unknowns
    2. Partition approach: Quadratic system with 315 unknowns
        - Followed by linear system of 105 unknowns

# Our Case

- Our system can be written as $F(k_t, z_t; \sigma) = \mathcal{H}(c_t, c_{t+1}, k_t, k_{t+1}, z_t; \sigma) =$

$$\begin{bmatrix} c(k_t, z_t; \sigma)^{-\gamma} - \beta E_t[(\alpha e^{\rho z_t + \sigma \eta \epsilon_{t+1}} k(k_t, c_t; \sigma)^{\alpha-1} + 1 - \delta) c(k(k_t, z_t; \sigma), \rho z_t + \sigma \eta \epsilon_{t+1}; \sigma)^{-\gamma}] \\ c(k_t, z_t; \sigma) + k(k_t, z_t; \sigma) - e^{z_t} k_t^{\alpha} - (1-\delta) k_t \end{bmatrix}$$

- Derivatives tell us

$$\mathcal{H}_1 c_k + \mathcal{H}_2 c_k k_k + \mathcal{H}_3 + \mathcal{H}_4 k_k = \mathbf{0}$$

$$\mathcal{H}_1 c_z + \mathcal{H}_2 (c_k k_z + c_k \rho) + \mathcal{H}_4 k_z + \mathcal{H}_5 = \mathbf{0}$$

- Partition system: First row of conditions $\rightarrow (c_k, k_k)$
  - Plug solution into second row $\rightarrow (c_z, k_z)$

## Ordering the System

- Gather constant, linear, and higher order terms as follows

$$\begin{bmatrix} 0 & \mathcal{H}_2^1 \\ 0 & \mathcal{H}_2^2 \end{bmatrix} \begin{bmatrix} k_k^2 & 0 \\ c_k k_k & 0 \end{bmatrix} + \begin{bmatrix} \mathcal{H}_4^1 & \mathcal{H}_1^1 \\ \mathcal{H}_4^2 & \mathcal{H}_1^2 \end{bmatrix} \begin{bmatrix} k_k & 0 \\ c_k & 0 \end{bmatrix} + \begin{bmatrix} \mathcal{H}_3^1 & 0 \\ \mathcal{H}_3^2 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

- Rewrite to look quadratic

$$\underbrace{\begin{bmatrix} 0 & \mathcal{H}_2^1 \\ 0 & \mathcal{H}_2^2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} k_k & 0 \\ c_k & 0 \end{bmatrix}^2}_{P^2} + \underbrace{\begin{bmatrix} \mathcal{H}_4^1 & \mathcal{H}_1^1 \\ \mathcal{H}_4^2 & \mathcal{H}_1^2 \end{bmatrix}}_{-B} \underbrace{\begin{bmatrix} k_k & 0 \\ c_k & 0 \end{bmatrix}}_{P} + \underbrace{\begin{bmatrix} \mathcal{H}_3^1 & 0 \\ \mathcal{H}_3^2 & 0 \end{bmatrix}}_{-C} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

- Column two is filled with zeros...eq'm conditions contained in column one

## Our Case

- The $\mathcal{H}$ vectors are as follows

$$\mathcal{H}_1 = \begin{bmatrix} -\gamma \bar{c}^{-\gamma-1} \\ 1 \end{bmatrix}, \quad \mathcal{H}_2 = \begin{bmatrix} \beta(\alpha \bar{k}^{\alpha-1} + 1 - \delta)\gamma \bar{c}^{-\gamma-1} \\ 0 \end{bmatrix}$$

$$\mathcal{H}_3 = \begin{bmatrix} 0 \\ -\alpha \bar{k}^{\alpha-1} - 1 + \delta \end{bmatrix}, \quad \mathcal{H}_4 = \begin{bmatrix} -\beta\alpha(\alpha-1)\bar{k}^{\alpha-2}\bar{c}^{-\gamma} \\ 1 \end{bmatrix}$$

$$\mathcal{H}_5 = \begin{bmatrix} -\beta\rho\alpha \bar{k}^{\alpha-1}\bar{c}^{-\gamma} \\ -\bar{k}^{\alpha} \end{bmatrix}$$

- Our stable solution ought to feature $k_k < 1$, since

$$k_{t+1} = \bar{k} + k_k(k_t - \bar{k}) + \dots$$

# Wrapping things up

- Rest of the system linear

$$\begin{bmatrix} \mathcal{H}_2^1 & \mathcal{H}_1^1 \\ \mathcal{H}_2^2 & \mathcal{H}_1^2 \end{bmatrix} \begin{bmatrix} k_z \\ c_z \end{bmatrix} = - \begin{bmatrix} \mathcal{H}_5^1 + \mathcal{H}_4^1 k_k + \mathcal{H}_2^1 \rho c_k \\ \mathcal{H}_5^2 + \mathcal{H}_4^2 k_k + \mathcal{H}_2^2 \rho c_k \end{bmatrix}$$

- Implies

$$\begin{bmatrix} k_z \\ c_z \end{bmatrix} = - \begin{bmatrix} \mathcal{H}_2 & \mathcal{H}_1 \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{H}_5 + \mathcal{H}_4 k_k + \mathcal{H}_2 \rho c_k \end{bmatrix}$$

- We know that all $\sigma$-terms are zero, so we're done!

# Comparing Solutions

- Take our first-order solution and manipulate to get in terms of log-differences

$$c_t = \bar{c} + c_k(k_t - \bar{k}) + c_z z_t$$

$$k_{t+1} = \bar{k} + k_k(k_t - \bar{k}) + k_z z_t$$

- Rearrange

$$\hat{c}_t = \frac{\bar{k}}{\bar{c}} c_k \hat{k}_t + \frac{c_z}{\bar{k}} z_t$$

$$\hat{k}_{t+1} = k_k \hat{k}_t + \frac{k_z}{\bar{k}} z_t$$

- Can compare this to our log-linearized, LRE method

# Comparing Solutions

- Very different predictions/IRF functions!
- How do we know which one is better?
- Popular method: Euler Equation errors

$$EEE(k_t, z_t) = \left| 1 - \frac{\beta E_t\left[(\alpha e^{\tilde{z}_{t+1}} k_{t+1}(k_t, z_t)^{\alpha-1} + 1 - \delta) c_t(k_{t+1}(k_t, z_t), \tilde{z}_{t+1})^{-\gamma}\right]}{c_t(k_t, z_t)^{-\gamma}} \right|$$

1. Simulate model for a long time
2. Compute deviations of approximate solution from true Euler Equations:
3. Average over entire sample
   - Straight average or weighted by ergodic distribution
4. Accuracy Metric: $\log_{10} \bar{EE}$
   - e.g. $-2(-3) \implies$ \$1 mistake for every \$100(\$1000) consumed

# A Quick Thought

- Some authors took a different route to simplify
- Kydland and Prescott (1982)
    1. Take second-order approximation of *utility function*
    2. Solve this simpler problem (Linear-Quadratic)
- Also approach taken by Mackowiak and Wiederholdt (2009)
- If all constraints linear, yields same solution as first-order perturbation

# Higher-Order

- Natural next step: Up the approximation degree!
- Not clear how to do this when log-linearizing, but straightforward with true perturbation
- Second-order approximation to policy function

$$
\begin{aligned}
[g(x;\sigma)]^i = {} & [g(\bar{x};0)]^i + [g_x(\bar{x};0)]^i_a [(x-\bar{x})]_a + [g_\sigma(\bar{x};0)]^i \sigma \\
& + \frac{1}{2} [g_{xx}(\bar{x};0)]^i_{ab} [(x-\bar{x})]_a [(x-\bar{x})]_b \\
& + [g_{x\sigma}(\bar{x};0)]^i_a [(x-\bar{x})]_a \sigma \\
& + \frac{1}{2} [g_{\sigma\sigma}(\bar{x};0)]^i \sigma^2
\end{aligned}
$$

- Exactly the same for $h$

# Solving

- Take another derivative of our system $[F_x(\bar{x};0)]^i_j$ w.r.t. $x$ (each line is w.r.t. $x_k$)

$$0 = \left[F_{xx}(\bar{x};0)^i_{jk}\right] =$$

$$\left([\mathcal{H}_{y'y'}]^i_{\alpha\gamma}[g_x]^\gamma_\delta[h_x]^\delta_k + [\mathcal{H}_{y'y}]^i_{\alpha\gamma}[g_x]^\gamma_k + [\mathcal{H}_{y'x'}]^i_{\alpha\delta}[h_x]^\delta_k + [\mathcal{H}_{y'x}]^i_{\alpha k}\right)[g_x]^\alpha_\beta[h_x]^\beta_j$$

$$+ [\mathcal{H}_{y'}]^i_\alpha[g_{xx}]^\alpha_{\beta\delta}[h_x]^\delta_k[h_x]^\beta_j + [\mathcal{H}_{y'}]^i_\alpha[g_x]^\alpha_\beta[h_{xx}]^\beta_{jk}$$

$$+\left([\mathcal{H}_{yy'}]^i_{\alpha\gamma}[g_x]^\gamma_\delta[h_x]^\delta_k+[\mathcal{H}_{yy}]^i_{\alpha\gamma}[g_x]^\gamma_k+[\mathcal{H}_{yx'}]^i_{\alpha\delta}[h_x]^\delta_k+[\mathcal{H}_{yx}]^i_{\alpha k}\right)[g_x]^\alpha_j+[\mathcal{H}_y]^i_\alpha[g_{xx}]^\alpha_{jk}$$

$$+\left([\mathcal{H}_{xy'}]^i_{\beta\gamma}[g_x]^\gamma_\delta[h_x]^\delta_k+[\mathcal{H}_{x'y}]^i_{\beta\gamma}[g_x]^\gamma_k+[\mathcal{H}_{x'x'}]^i_{\beta\delta}[h_x]^\delta_k+[\mathcal{H}_{x'x}]^i_{\beta k}\right)[h_x]^\beta_j+[\mathcal{H}_{x'}]^i_\beta[h_{xx}]^\beta_{jk}$$

$$+[\mathcal{H}_{xy'}]^i_{j\gamma}[g_x]^\gamma_\delta[h_x]^\delta_k + [\mathcal{H}_{xy}]^i_{j\gamma}[g_x]^\gamma_k + [\mathcal{H}_{xx'}]^i_{j\delta}[h_x]^\delta_k + [\mathcal{H}_{xx}]^i_{jk}$$

# Solving

- Looks complicated, BUT
  - All derivatives of $\mathcal{H}$ are known
  - All first-order terms have already been solved for
- What's left is a series of $n \times n_x \times n_x$ equations in $n \times n_x \times n_x$ unknowns
  - But it's all linear! No quadratic parts anymore!
  - Computer can solve for it easily: Delivers $g_{xx}$ and $h_{xx}$
- First-order approximation determines whether we are in stable manifold
  - Once we know that, no additional solutions need to be ruled out as we increase the approximation degree

# Solving

- Rest of solution can be found separately
- First, note $g_{\sigma x} = h_{\sigma x} = 0$

$$0 = [F_{\sigma x}(\bar{x}; 0)]^i_j =$$

$$[\mathcal{H}_{y'}]^i_\alpha [g_x]^\alpha_\beta [h_{\sigma x}]^\beta_j + \mathcal{H}_{y'}]^i_\alpha [g_{\sigma x}]^\alpha_\gamma [h_{\sigma x}]^\gamma_j [\mathcal{H}_y]^i_\alpha [g_{\sigma x}]^\alpha_j + [\mathcal{H}_{x'}]^i_\beta [h_{\sigma x}]^\beta_j$$

- Cross-partials necessarily zero in unique solution since system is linear and homogeneous

# Breaking Certainty Equivalence

- Derivative system for $F_{\sigma\sigma}$ reveals departure from certainty equivalence
- Following system of $n$ equations in $n$ unknowns: $g_{\sigma\sigma}$ and $h_{\sigma\sigma}$
  - Shift constant term in perturbation; precautionary behavior

$$0 = [F_{\sigma\sigma}(\bar{x};0)]^i = [\mathcal{H}_{y'}]^i_\alpha [g_x]^\alpha_\beta [h_{\sigma\sigma}]^\beta$$

$$+ [\mathcal{H}_{y'y'}]^i_{\alpha\gamma}[g_x]^\gamma_\delta[\eta]^\delta_\xi[g_x]^\alpha_\beta[\eta]^\beta_\phi[I]^\phi_\xi + [\mathcal{H}_{y'x'}]^i_{\alpha\delta}[\eta]^\delta_\xi[g_x]^\alpha_\beta[\eta]^\beta_\phi[I]^\phi_\xi$$

$$+ [\mathcal{H}_{y'}]^i_\alpha[g_{xx}]^\alpha_{\beta\delta}[\eta]^\delta_\xi[\eta]_{\phi\beta}[I]^\phi_\psi + [\mathcal{H}_{y'}]^i_\alpha[g_{\sigma\sigma}]^\alpha$$

$$+ [\mathcal{H}_y]^i_\alpha[g_{\sigma\sigma}]^\alpha + [\mathcal{H}_{x'}]^i_\beta[h_{\sigma\sigma}]^\beta$$

$$+ [\mathcal{H}_{x'y'}]^i_{\beta\gamma}[g_x]^\gamma_\delta[\eta]^\delta_\xi[\eta]^\beta_\phi[I]^\phi_\xi + [\mathcal{H}_{x'x'}]^i_{\beta\delta}[\eta]^\delta_\xi[\eta]^\beta_\phi[I]^\phi_\xi$$

# Higher-Order

- As we move up to third, fourth, fifth-order nothing changes
- Each new system is linear given solution of lower-order system
- If computer is autodifferentiating, memory becomes the biggest issue
    - Even very large matrices can be inverted relatively quickly
- We won't solve our benchmark example for second/higher-order yet...
    - Too big to be a useful exercise
    - We'll use Dynare to do it in a bit

# Pruning

- Higher-order terms complicate things...
- Linear systems converge globally to their steady state
    - Not true of nonlinear systems!
    - Stable manifold may not span entire relevant space...
- With unbounded shocks (e.g. AR(1)), eventually system will get kicked out of stable region
- Intuition

$$k_{t+1} = a_0 + a_1(k_t - \bar{k}) + a_2(k_t - \bar{k})^2 + \cdots + b\epsilon_{t+1}$$

$$k_{t+1} = a_0 + a_1(a_0 + a_1(k_{t-1} - \bar{k}) + a_2(k_{t-1} - \bar{k})^2 + \cdots + b\epsilon_t - \bar{k})$$

$$+ a_2(a_0 + a_1(k_{t-1} - \bar{k}) + a_2(k_{t-1} - \bar{k})^2 + \cdots + b\epsilon_t - \bar{k})^2 + \cdots + b\epsilon_{t+1}$$

# Pruning

- Notice $k_{t+1}$ contains $k_{t-1}$ terms raised to the 3rd and 4th powers
- A sufficiently large $\epsilon_{t-1}$ shock can push it out of the *radius of convergence*
  - i.e. local space where perturbation is valid
  - Once outside, it will tend to explode as $k_{t-1}$ gets taken to successively higher powers
- Solution: 'Prune' the approximation
  1. Expand 1 period overlap to an *n* period recursion, where *n* is the order of the approximation
  2. Get rid of all terms with order (strictly) higher than *n* i.e. set coefficients to zero
- Pruned system guaranteed not to explode
- Closed-form expression for impulse response functions *and* theoretical moments
  - No need to run simulations to get first and second moments

# Perturbing the Value Function

- Sometimes, it is necessary to perturb the value function to solve model
  - Epstein-Zin preferences i.e. recursive utility

$$V(z_t, k_t) = \max_{k_{t+1}, c_t} \left[(1-\beta)u(c_T)^\rho + \beta E[V(\tilde{z}_{t+1}, k_{t+1})]^\rho\right]^{\frac{1}{\rho}}$$

- Other times, we may want it as a quick (accurate) initial guess for a globally accurate method
- Finally, we may want it to conduct welfare analysis

# Perturbing the Value Function

- The value function is treated as a part of the policy function (as opposed to the law of motion)
- The (second-order) perturbed value function is

$$V(k_t, z_t; 1) = \bar{V} + \bar{V}_1(k_t - \bar{k}) + \bar{V}_2 z_t + \frac{1}{2} V_{11}(k_t - \bar{k})^2$$

$$+ \frac{1}{2}\bar{V}_{22} z_t^2 + \bar{V}_{12}(k_t - \bar{k}) z_t + \frac{1}{2}\bar{V}_{33}$$

- Easy to prove that $\bar{V}_{33} < 0$. Implies interesting exercise

$$V(\bar{k}, 0; 1) = \bar{V} + \frac{1}{2}\bar{V}_{33} < \bar{V}$$

# Perturbing the Value Function

- Can think of $-\frac{1}{2}\bar{V}_{33}$ as the second-order approximation to the welfare cost of business cycles at steady state
  - Relative to perfect foresight economy
- If we wanted to get certainty-equivalent-consumption loss, we need only compute CEC for each case

$$\frac{u(c_{PF})}{1-\beta} = \bar{V} \qquad \frac{u(c_{BM})}{1-\beta} = \bar{V} + \frac{1}{2}\bar{V}_{33}$$

- Perfect foresight agent willing to give up to a fraction $\tau$ to avoid risk of benchmark business cycles, where

$$(1-\tau)c_{PF} = c_{BM}$$

# Solving Higher-Order Perturbations

- Not always ideal to do higher-order perturbations by hand
- Two avenues for mistakes
  1. Derivations by hand
  2. Implementation in code
- Autodifferentiation can get around this
- But others have already done it! Why write it yourself?
  - Dynare: Implementation in Matlab, Octave, Julia, etc.

# Dynare

- Dynare is a software platform specifically designed to solve DSGE and OLG models
- Maintained by Stephane Adjemian, with heavy advice from many leading figures in the field
- Relatively straightforward interface

1. Enter variables/parameters
2. Enter equilibrium conditions
3. Enter steady state (optional)
4. Define shocks
5. Select options
6. Dynare solves model with perturbation

# Navigating Dynare: Preamble

- Dynare works in '.mod' files (stands for 'model')
- Comments with '//' ('/*' comments out sections)
- End line with ';' (just like Matlab)
- Objects defined in the preamble

1. 'var' - Endogenous variables
2. 'varexo' - Shocks (not Markov process; literally the shocks)
3. 'parameters' - Model parameters

- List parameters first, then initiate line-by-line

# Navigating Dynare: Model

- Equilibrium conditions go in ‘model’ section
- Denote end with ‘end;’
- Write expressions with equality (Dynare will automatically set it to zero)
- All variables *assumed* to be denoted at time $t$ e.g. $y_t = y$
  - Denote $t+1$ variables as $y(+1)$
  - Denote earlier periods as $y(-1)$
  - Larger distances possible

# Navigating Dynare: Model Continued

- Note that Dynare will automatically take an expectation over all $(+1)$ variables (no need to specify expectations)
  - Best to denote as period $t$ variables anything *determined* in $t$
  - e.g. in RBC model, let $k_{t+1} = k$ and $k_t = k(-1)$, since it was determined yesterday
  - Be careful about nested/multiplicative expectations! Often need to define auxiliary variables!
- Dynare will warn you if the number of non-predetermined variables $>$ number of eigenvalues less than one
  - This is called *Blanchard-Kahn* Condition (think back to LRE models for intuition)
  - If not satisfied, solution may not exist or approximation may be unstable/unreliable

# Navigating Dynare: Steady State

- Begin with 'initval;'
- End with 'end;'
- Provide steady state values
- Two options
  1. Use exactly what you provided (nothing further)
  2. Use what you provided as an initial guess and solve for SS
     - Follow 'end;' with the 'steady;' command
- If Dynare has trouble finding the steady state, can give it options to use other solvers
- Give command 'check;' after 'steady;' to determine if system is stable (in eigenvalue sense) before we proceed

## Navigating Dynare: Shocks

- Start with ‘shocks;’ command
- Set ‘var x = the_variance;’ for each shock
- Covariance set by ‘var x,y = the_covariance;’ for each shock
- End with ‘end;’

# Solving with Dynare

- When everything in order, end with ‘stoch_simul;’
  - Instructs Dynare to solve model with perturbation methods
  - Matlab command: dynare mydynarecode.mod
- Delivers
  1. Policy function
  2. Theoretical moments
  3. Variance decomposition
  4. Correlations/autocorrelations
  5. IRFs for all shocks
- Default Taylor order is 2 (automatically prunes)

# Solving with Dynare

- Can specify as arguments (among others)
  1. periods = number of simulation periods (default = 0)
  2. order = Taylor approximation order
  3. noprint (ensures nothing printed; may want in a loop)
  4. relative_irf (puts IRF in terms of standard error of shock instead of own % change)
- Dynare puts all results into a Matlab struct (.mat) for later potential use

# Example

- Return to stochastic NCG model

$$c_t^{-\sigma} = \beta E_t[(e^{\tilde{z}_{t+1}} \alpha k_{t+1}^{\alpha-1} + 1 - \delta) c_{t+1}^{-\sigma}$$

$$c_t + k_{t+1} = e^{z_t} k_t^{\alpha} + (1 - \delta) k_t$$

$$z_t = \rho z_{t-1} + \sigma_z \epsilon_t$$

- Note that $c = c_t$, but $k = k_{t+1}$
- Same parameters as before
- Notice ‘correction’ term in second-order approximation
  - This is precautionary term, which adjusts the constant
  - Grows as we increase the variance

# Example 2

- Nice feature: Can scale up *really* easily
- Consider following model with two shocks (TFP and investment efficiency)

$$\max E_t \sum_{\tau=t}^{\infty} \beta^{\tau-t} \left( \log(c_t) - \theta \frac{h_t^{1+\psi}}{1+\psi} \right)$$

$$c_t + i_t = y_t$$

$$k_{t+1} = e^{b_t} i_t + (1-\delta) k_t$$

$$y_t = e^{a_t} k_t^{\alpha} h_t^{1-\alpha}$$

$$\begin{pmatrix} a_t \\ b_t \end{pmatrix} = \begin{pmatrix} \rho & \tau \\ \tau & \rho \end{pmatrix} \begin{pmatrix} a_{t-1} \\ b_{t-1} \end{pmatrix} + \begin{pmatrix} \epsilon_t \\ \nu_t \end{pmatrix}$$

# Sidenote: Stationarizing Models

- All of our recursive solution techniques (except for finite-horizon models) assume stationarity
- What if we have a non-stationary model?
    - Sometimes we can solve a transformed, stationarized version
    - Sufficient information is then contained to solve original model
- Most common environment: Growth
    1. Deterministic growth
    2. Stochastic (permanent) growth shocks

# Deterministic Growth

- Example: RBC Model with Deterministic Growth
- Production function: $A_t^{1-\alpha} K_t^{\alpha}$, where

$$A_t = \hat{A}_t e^{z_t}$$

$$\hat{A}_t = e^g \hat{A}_{t-1}$$

for some constant $g > 0$; $z_t$ follows mean-zero AR(1) process

- Normal Bellman equation depends on time

$$V_t(A_t, K_t) = \max_{K_{t+1}} \frac{\left(A_t^{1-\alpha} K_t^{\alpha} + (1-\delta) K_t - K_{t+1}\right)^{1-\sigma}}{1-\sigma} +$$

$$\beta E_t \left[ V_{t+1}(\tilde{A}_{t+1}, K_{t+1}) \right]$$

i.e. non-stationary

# Deterministic Growth

- To solve, define two new objects:
  1. $\hat{V}_t = V_t(A_t, K_t)/\hat{A}_t^{1-\sigma}$
  2. $k_t = K_t/\hat{A}_t$
- Can re-write this Bellman as

$$\hat{V}_t(z_t, k_t) = \max_{k_{t+1}} \frac{\left(e^{(1-\alpha)z_t} k_t^{\alpha} + (1-\delta)k_t - e^g k_{t+1}\right)^{1-\sigma}}{1-\sigma} +$$

$$\beta e^{(1-\sigma)g} E_t\left[\hat{V}_{t+1}(\tilde{z}_{t+1}, k_{t+1})\right]$$

- This Bellman is stationary (can remove the $t$-subscript)
  - Once solved, can undo transform to get solution to original model

# Stochastic Growth

- Same environment, but suppose that

$$A_t = e^{g_t} A_{t-1}$$

and that $g_t$ follows an AR(1) process

- Now, we define
  1. $\hat{V}_t = V_t(A_t, K_t)/A_{t-1}^{1-\sigma}$
  2. $k_t = K_t/A_{t-1}$
- Transformed system is stationary!

$$\hat{V}_t(g_t, k_t) = \max_{k_{t+1}} \frac{\left(e^{(1-\alpha)g_t} k_t^{\alpha} + (1-\delta)k_t - e^{g_t} k_{t+1}\right)^{1-\sigma}}{1-\sigma} +$$

$$\beta e^{(1-\sigma)g_t} E_t \left[ \hat{V}_{t+1}(\tilde{g}_{t+1}, k_{t+1}) \right]$$