Dynare

Wouter J. Den Haan
London School of Economics

Introduction

- What is the objective of perturbation?
- Peculiarities of Dynare & some examples
- Incorporating Dynare in other Matlab programs
- Impulse Response Functions
- Local and/or global approximation?
- Perturbation and the effect of uncertainty on the solution
- Pruning

Objective of 1st-order perturbation

- Obtain *linear* approximations to the policy functions that satisfy the first-order conditions
- state variables: $x_t = [x_{1,t}\ x_{2,t}\ x_{3,t}\ \cdots x_{n,t}]'$
- result:

$$y_t = \bar{y} + (x_t - \bar{x})'a$$

- a bar above a variable indicates steady state value

Underlying theory

- Model:

$$\mathbb{E}_t\left[f(g(x))\right] = 0,$$

  - $f(x)$ is completely known
  - $g(x)$ is the unknown policy function.
- Perturbation: Solve *sequentially* for the coefficients of the Taylor expansion of $g(x)$.

Neoclassical growth model

- $x_t = [k_{t-1}, z_t]$
- $y_t = [c_t, k_t, z_t]$
- linearized solution:

$$\begin{aligned}
c_t &= \bar{c} + a_{c,k}(k_{t-1} - \bar{k}) + a_{c,z}(z_t - \bar{z}) \\
k_t &= \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z}(z_t - \bar{z}) \\
z_t &= \rho z_{t-1} + \varepsilon_t
\end{aligned}$$

Linear in what variables?

- Dynare does not understand what c_t is.
  - could be level of consumption
  - could be log of consumption
  - could be rainfall in Scotland

- Dynare simply generates a **linear** solution in what you specify as the variables
- More on this below

Peculiarities of Dynare

- Variables known at beginning of period t *must* be dated $t-1$.
- Thus,
  - k_t: the capital stock *chosen* in period t
  - k_{t-1}: the capital stock available at beginning of period t

Peculiarities of Dynare

$$k_t = \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z}(z_t - \bar{z})$$
$$z_t = \rho z_{t-1} + \varepsilon_t$$

can of course be written (less conveniently) as

$$k_t = \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z_{-1}}(z_{t-1} - \bar{z}) + a_{k,z}\varepsilon_t$$
$$z_t = \rho z_{t-1} + \varepsilon_t$$

with

$$a_{k,z_{-1}} = \rho a_{k,z}$$

Peculiarities of Dynare

- Dynare gives the solution in the less convenient form:

$$\begin{gathered} c_t = \bar{c} + a_{c,k}(k_{t-1} - \bar{k}) + a_{c,z_{-1}}(z_{t-1} - \bar{z}) + a_{c,z}\varepsilon_t \\ k_t = \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z_{-1}}(z_{t-1} - \bar{z}) + a_{k,z}\varepsilon_t \\ z_t = \rho z_{t-1} + \varepsilon_t \end{gathered}$$

- But you can rewrite it in the more convenient shorter form

Dynare program blocks

- **Labeling block**: indicate which symbols indicate what
  - variables in "var"
  - exogenous shocks in "varexo"
  - parameters in "parameters"
- **Parameter values block**: Assign values to parameters

Dynare program blocks

- **Model block**: Between "model" and "end" write down the n equations for n variables
  - the equations have conditional expectations, having a $(+1)$ variable makes Dynare understand there is one in this equation

Dynare program blocks

- **Initialization block**: Dynare has to solve for the steady state. This can be the most difficult part (since it is a true non-linear problem). So good initial conditions are important
- **Random shock block**: Indicate the standard deviation for the exogenous innovation

Dynare program blocks

- **Solution & Properties block**:
    - Solve the model with the command
        - 1st-order: stoch_simul(order=1,nocorr,nomoments,IRF=0)
        - 2nd-order: stoch_simul(order=2,nocorr,nomoments,IRF=0)
    - Dynare can calculate IRFs and business cycle statistics. E.g.,
        - stoch_simul(order=1,IRF=30),
        - but I would suggest to program this yourself (see below)

Running Dynare

- In Matlab change the directory to the one in which you have your *.mod files
- In the Matlab command window type

```
dynare programname
or
dynare programname.mod
```

- This will create and run several Matlab files

Model with productivity in levels (FOCs A)

Specification of the problem

$$\max_{\{c_t,k_t\}_{t=1}^{\infty}} \mathbb{E}\sum_{t=1}^{\infty} \beta^{t-1}\frac{c_t^{1-\nu}-1}{1-\nu}$$

s.t.

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

$$z_t = (1-\rho) + \rho z_{t-1} + \varepsilon_t$$

k_0 given

$$\mathbb{E}_t[\varepsilon_{t+1}] = 0 \;\&\; \mathbb{E}_t[\varepsilon_{t+1}^2] = \sigma^2$$

Distribution of innovation

- 1st-order approximations:
    - solution assumes that $\mathbb{E}_t[\varepsilon_{t+1}] = 0$
    - other properties of the distribution do not matter
- 2nd-order approximations:
    - solution assumes that $\mathbb{E}_t[\varepsilon_{t+1}] = 0$
    - σ matters, it affects the constant of the policy function
    - other properties of the distribution do not matter

Everything in levels: FOCs A

Model equations:

$$\begin{aligned} c_t^{-\nu} &= \mathbb{E}_t\left[\beta c_{t+1}^{-\nu}(\alpha z_{t+1}k_t^{\alpha-1}+1-\delta)\right] \\ c_t+k_t &= z_t k_{t-1}^{\alpha}+(1-\delta)k_{t-1} \\ z_t &= (1-\rho)+\rho z_{t-1}+\varepsilon_t \end{aligned}$$

Dynare equations:

```
c^(-nu)
 =beta*c(+1)^(-nu)*(alpha*z(+1)*k^(alpha-1)+1-delta);
c+k=z*k(-1)^alpha+(1-delta)k(-1);
z=(1-rho)+rho*z(-1)+e;
```

Policy functions reported by Dynare

- $\delta = 0.025, \nu = 2$, $\alpha = 0.36$, $\beta = 0.99$, and $\rho = 0.95$

```
POLICY AND TRANSITION FUNCTIONS

             k           z           c
 constant    37.989254   1.000000    2.754327
 k(-1)       0.976540    -0.000000   0.033561
 z(-1)       2.597386    0.950000    0.921470
 e           2.734091    1.000000    0.969968
```

!!!! You have to read output as

	k	z	c
constant	37.989254	1.000000	2.754327
k(-1)-k_{ss}	0.976540	-0.000000	0.033561
z(-1)-z_{ss}	2.597386	0.950000	0.921470
e	2.734091	1.000000	0.969968

- That is, explanatory variables are relative to steady state.
- (Note that steady state of e is zero by definition)
- If explanatory variables take on steady state values, then choices are equal to the constant term, which of course is simply equal to the corresponding steady state value

Changing amount of uncertainty

Suppose $\sigma = 0.1$ instead of 0.007

```
POLICY AND TRANSITION FUNCTIONS

              k            z            c
 constant     37.989254    1.000000     2.754327
 k(-1)        0.976540     -0.000000    0.033561
 z(-1)        2.597386     0.950000     0.921470
 e            2.734091     1.000000     0.969968
```

- Any change?

Model with productivity in logs

Specification of the problem

$$\max_{\{c_t,k_t\}_{t=1}^{\infty}} \mathbb{E} \sum_{t=1}^{\infty} \beta^{t-1} \frac{c_t^{1-\nu} - 1}{1-\nu}$$

s.t.

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

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

$$k_0 \text{ given}, \;\; \mathbb{E}_t[\varepsilon_{t+1}] = 0$$

Variables in levels & prod. in logs - FOCs B

Model equations:

$$c_t^{-\nu} = \mathbb{E}_t\left[\beta c_{t+1}^{-\nu}(\alpha \exp(z_{t+1})k_t^{\alpha-1} + 1 - \delta)\right]$$
$$c_t + k_t = \exp(z_t)k_{t-1}^{\alpha} + (1-\delta)k_{t-1}$$
$$z_t = \rho z_{t-1} + \varepsilon_t$$

Dynare equations:

```
c^(-nu)
=beta*c(+1)^(-nu)*(alpha*exp(z(+1))*k^(alpha-1)+1-delta);
c+k=exp(lz)*k(-1)^alpha+(1-delta)k(-1);
lz=rho*lz(-1)+e;
```

Linear solution in what?

Dynare gives a linear system in what you specify the variables to be

Variables in logs - FOCs C

Model equations:

$$(\exp(\tilde{c}_t))^{-\nu} =$$

$$= \mathbb{E}_t \left[\beta(\exp(\tilde{c}_{t+1}))^{-\nu}(\alpha \exp(z_{t+1})(\exp(\tilde{k}_t))^{\alpha-1} + 1 - \delta) \right]$$

$$\exp(\tilde{c}_t) + \exp(\tilde{k}_t) = \exp(z_t)(\exp(\tilde{k}_{t-1}))^{\alpha} + (1-\delta)\exp(\tilde{k}_{t-1})$$

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

The variables $\tilde{c}_t$ and $\tilde{k}_t$ are the *log* of consumption and capital.

All variables in logs - FOCs C

Model equations (rewritten a bit)

$$\exp(-\nu\tilde{c}_t)$$
$$= \mathbb{E}_t\left[\beta\exp(-\nu\tilde{c}_{t+1})(\alpha\exp(z_{t+1}+(\alpha-1)\tilde{k}_t)+1-\delta)\right]$$

$$\begin{aligned}\exp(\tilde{c}_t)+\exp(\tilde{k}_t) &= \exp(z_t+\alpha\tilde{k}_{t-1})+(1-\delta)\exp(\tilde{k}_{t-1})\\ z_t &= \rho z_{t-1}+\varepsilon_t\end{aligned}$$

All variables in logs - FOCs C

Dynare equations:

```
exp(-nu*lc)=beta*exp(-nu*lc(+1))*
(alpha*exp(lz(+1)+(alpha-1)*lk))+1-delta);

exp(lc)+exp(lk)
=exp(lz+alpha*lk(-1))+(1-delta)exp(lk(-1));

lz=rho*lz(-1)+e;
```

All variables in logs - FOCs C

- This system gives policy functions that are linear in the variables `lc`, i.e., $\ln(c_t)$, `lk`, i.e., $\ln(k_t)$, and `lz`, i.e., $\ln(z_t)$

- Note that we could have found the coefficients of the log system **without** rerunning Dynare.
  To understand consider the following:

  1. How would coefficients change if we use $(c_t - c_{ss})/c_{ss}$ instead of c_t as variable?
  2. How would they change if we use $\ln(c_t)$ instead of $(c_t - c_{ss})/c_{ss}$?

All variables in logs - FOCs C

Is the following system any different?

```
exp(-nu*c)=beta*exp(-nu*c(+1))*
(alpha*exp(z(+1)+(alpha-1)*k))+1-delta);

exp(c)+exp(k)=exp(z+alpha*k(-1))+(1-delta)exp(k(-1));

z=rho*z(-1)+e;
```

Example with analytical solution

- If $\delta = \nu = 1$ then we know the analytical solution. It is

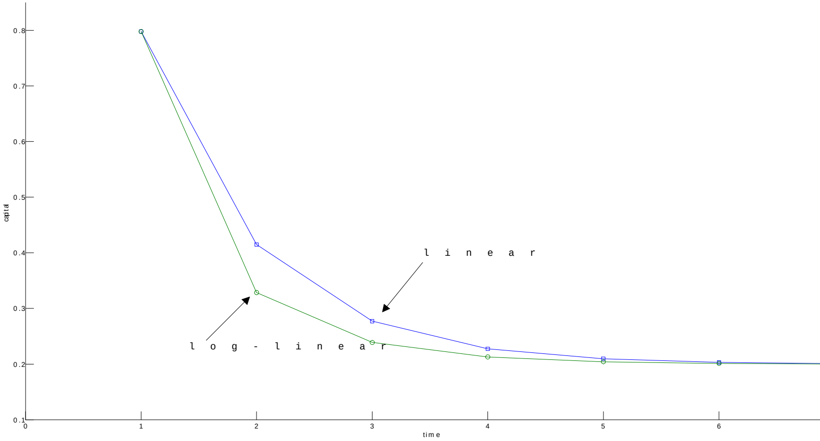
$$k_t = \alpha\beta \exp(z_t) k_{t-1}^{a}$$
$$c_t = (1-\alpha\beta) \exp(z_t) k_{t-1}^{a}$$

or

$$\begin{aligned} \ln k_t &= \ln(\alpha\beta) + \alpha \ln k_{t-1} + z_t \\ \ln c_t &= \ln(1-\alpha\beta) + \alpha \ln k_{t-1} + z_t \end{aligned}$$

Are linear and loglinear the same?

Suppose that $k_0 = 0.798$ & $z_t = 0\ \forall t$. Then the two time paths are given by



Substitute out consumption- FOCs D

Model equations:

$$\left[z_t \exp(\alpha \tilde{k}_{t-1}) + (1-\delta)\exp(\tilde{k}_{t-1}) - \exp(\tilde{k}_t)\right]^{-\nu}$$
$$=$$
$$\mathbb{E}_t \left\{ \beta \left(\begin{array}{c} \left[\exp(z_{t+1} + \alpha \tilde{k}_t) + (1-\delta)\exp(\tilde{k}_t) - \exp(\tilde{k}_{t+1})\right]^{-\nu} \times \\ (\alpha \exp(z_{t+1} + (\alpha-1)\tilde{k}_t) + 1 - \delta) \end{array} \right) \right\}$$

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

Does this substitution affect the solution?

Do it yourself!

- Try to do as much yourself as possible

What (not) to do your self

- Find the policy functions:
  - can be quite tricky, so let Dynare do it.
- IRFs, business cycle statistics, etc:
  - easy to program yourself
  - you know exactly what you are getting

Why do things yourself?

- Dynare linearizes *everything*
- Suppose you have an approximation in levels
- Add the following equation to introduce output

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

- Dynare will take a first-order condition of this equation to get a first-order approximation for y_t
- But you already have solutions for k_t and h_t

Why do things yourself?

- Getting the policy rules requires a bit of programming $\Longrightarrow$ let Dynare do this for you
- But, program more yourself $\Longrightarrow$ you understand more
- Thus program yourself the simpler things:
  - IRFs, simulated time paths, business cycle statistics, etc
  - That is, use stoch_simul(order=1,nocorr,nomoments,IRF=0)

Tricks

- Incorporating Dynare in other Matlab programs
- Read parameter values in *.mod file from external file
- Read Dynare policy functions *as they appear on the screen*
- How to get good initial conditions to solve for steady state

Keeping variables in memory

- Dynare clears all variables out of memory
- To overrule this, use

```
dynare program.mod noclearall
```

Saving solution to a file

- Replace the file "disp_dr.m" with alternatives available on my website
- I made two changes:
    - The original Dynare file only writes a coefficient to the screen if it exceeds 10^{-6} in absolute value. I eliminated this condition
    - I save the policy functions, *exactly* the way Dynare now writes them to the screen

To load the policy rules into a matrix called "decision" simply type

load dynarerocks

Loops

- The last trick allows you to run the same dynare program for different parameter values
- Suppose your Dynare program has the command

```
nu=3;
```

- You would like to run the program twice; once for nu=3, and once for nu=5.

Loops

1. In your Matlab program, loop over the different values of nu. Save the value of nu and the associated name to the file "parameterfile":

   save parameterfile nu

   and *then* run Dynare
2. In your Dynare program file, replace the command "nu = 3" with

   load parameterfile

   set_param_value('nu',nu);

This does the same

1. Loop over eta instead of nu

```
save hangten eta
```

2. In your *.mod file

```
load hangten
set_param_value('nu',eta);
```

- the name of the file is abitrary
- in set_param_value('·',·), the first argument is the name in your *.mod file and the second is the numerical vlaue

Homotophy

- Hardest part of Dynare is to solve for steady state
- Homotophy makes this a lot easier
- Suppose you want to the x such that

$$f\left(x;\alpha_1\right) = 0$$

  and suppose that you know the solution for α_0
- Consider

$$f(x;\omega\alpha_1 + (1-\omega)\alpha_0) = 0$$

Homotophy

- $$f(x;\omega\alpha_1+(1-\omega)\alpha_0)=0$$
- Set ω to a small value
- Solve for x using solution for α_0 as initial condition
- Increase ω slightly
- Solve for x using the latest solution for x as initial condition
- Continue until $\omega=1$

Homotophy

You could even use

- $$\omega f(x;\alpha_1) + (1-\omega)\, g(x;\alpha_0) = 0$$

  as your homotophy system
- Works best is $f(x;\alpha_1)$ is close to $g(x;\alpha_0)$

Using loop to get good initial conditions

With a loop you can update the initial conditions used to solve for steady state

1. Use parameters to define initial conditions
2. Solve model for simpler case
3. Gradually change parameter
4. Alternatives:
    1. use different algorithm to solve for steady state: solve_algo=1,2, or 3
    2. solve for coefficients instead of variables

Simple model with endogenous labor

1. Solve for c, k, h using

$$\begin{gathered} 1 = \beta\left(\alpha\,(k/h)^{\alpha-1} + 1 - \delta\right) \\ c + k = k^{\alpha}h^{1-\alpha} + (1-\delta)k \\ c^{-\nu}(1-\alpha)(k/h)^{\alpha} = \phi h^{\kappa} \\ \phi = 1 \end{gathered}$$

2. Or solve for c, k, ϕ using

$$\begin{gathered} 1 = \beta\left(\alpha\,(k/h)^{\alpha-1} + 1 - \delta\right) \\ c + k = k^{\alpha}h^{1-\alpha} + (1-\delta)k \\ c^{-\nu}(1-\alpha)(k/h)^{\alpha} = \phi h^{\kappa} \\ h = 0.3 \end{gathered}$$

Impulse Response functions

Definition: The effect of a one-standard-deviation shock

- Take as given k_0, z_0, and time series for ε_t, $\{\varepsilon_t\}_{t=1}^T$
- Let $\{k_t\}_{t=1}^T$ be the corresponding solutions

Impulse Response functions

- Consider the time series ε_t^* such that

$$\begin{array}{c} \varepsilon_t^* = \varepsilon_t \qquad \text{for } t \neq \tau \\ \varepsilon_t^* = \varepsilon_t + \sigma \text{ for } t = \tau \end{array}$$

- Let $\{k_t^*\}_{t=1}^T$ be the corresponding solutions
- Impulse response functions are calculated as

$$\begin{aligned} IRF_j^k &= k_{\tau+j}^* - k_{\tau+j} \quad \text{for } j \geq 0 \text{ if } k \text{ is in logs} \\ IRF_j^k &= \frac{k_{\tau+j}^* - k_{\tau+j}}{k_{\tau+j}} \quad \text{for } j \geq 0 \text{ if } k \text{ is in levels} \end{aligned}$$

Impulse Response functions

- Consider the time series ε_t^* such that

$$\begin{aligned} \varepsilon_t^* &= \varepsilon_t \qquad \text{for } t \neq \tau \\ \varepsilon_t^* &= \varepsilon_t + \sigma \text{ for } t = \tau \end{aligned}$$

- Let $\{k_t^*\}_{t=1}^T$ be the corresponding solutions
- Impulse response functions are calculated as

$$IRF_j^k(\sigma) = k_{\tau+j}^* - k_{\tau+j} \quad \text{for } j \geq 0$$

IRFs in general

- In general, IRFs will depend on
  - State of the economy when the shock occurs
    - thus depends on $\{\varepsilon_t\}_{t=1}^{\tau}$
  - Future shocks
    - thus depends on $\{\varepsilon_t\}_{t=\tau+1}^{\infty}$
- In general, $IRF_j^k(\sigma)/\sigma$ depends on sign and size of σ

IRFs in linear models

- In **linear** models, IRFs do **not** depend on
  - State of the economy when the shock occurs
  - Future shocks
- In **linear** modles, $IRF_j^k\left(\sigma\right)/\sigma$ does **not** depend on sign and size of σ

$\Longrightarrow$ You are free to pick the conditions anyway you want (including the easiest ones)

IRFs in linear models

Dynare gives you

$$k_t = \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z_{-1}}(z_{t-1} - \bar{z}) + a_{k,\varepsilon}\varepsilon_t$$

Easiest conditions:

- Start at $k_0 = \bar{k}$ and $z_0 = \bar{z}$ $(= 0)$
- Let $\varepsilon_1 = \sigma_\varepsilon$ and $\varepsilon_t = 0$ for $t > 1$
- Calculate time path for z_t
- Calculate time path for k_t
- Calculate time path for other variables

Impulse Response functions

higher-order case:

- One could repeat procedure described in last slide
- But this is just one of the many impulse response functions of the nonlinear model
- How to proceed?
    - calculate IRF for interesting initial condition (e.g., boom & recession)
    - simulate time series $\{k_t\}_{t=1}^{T}$ and calculate IRF at *each* point
        - IRF becomes a band

Properties perturbation solutions

❶ Impact uncertainty on policy function
❷ Accuracy as a global approximation
❸ Preserving shape & stability with higher-order approximations

Perturbation and impact of uncertainty

- Let σ be a parameter that scales all innovation standard deviations
  - $\sigma = 0 \Longrightarrow$ no uncertainty at all
- 1st-order: σ has *no* effect on policy rule at all
  - certainty equivalence
- 2nd-order: σ only affects the constant
- 3rd-order: σ only affects constant and linear terms

Perturbation and impact of uncertainty

Consequences for returns and risk premia:

- 1st-order: returns not affected by σ
  $\Longrightarrow$ no risk premium
- 2nd-order: σ only shifts returns
  $\Longrightarrow$ no time-varying risk premium
- 3rd-order: lowest possible order to get *any* time variation in returns

Theory

- Local convergence is guaranteed
- Global approximation *could* be good
- If the function is analytical $\Longrightarrow$ successive approximations converge towards the truth
- Theory says nothing about convergence patterns
- Theory doesn't say whether second-order is better than first
- For complex functions, this is what you have to worry about

Example with simple Taylor expansion

Truth:

$$\begin{aligned} f(x) &= -690.59 + 3202.4x - 5739.45x^2 \\ &\quad +4954.2x^3 - 2053.6x^4 + 327.10x^5 \end{aligned}$$

defined on $[0.7, 2]$

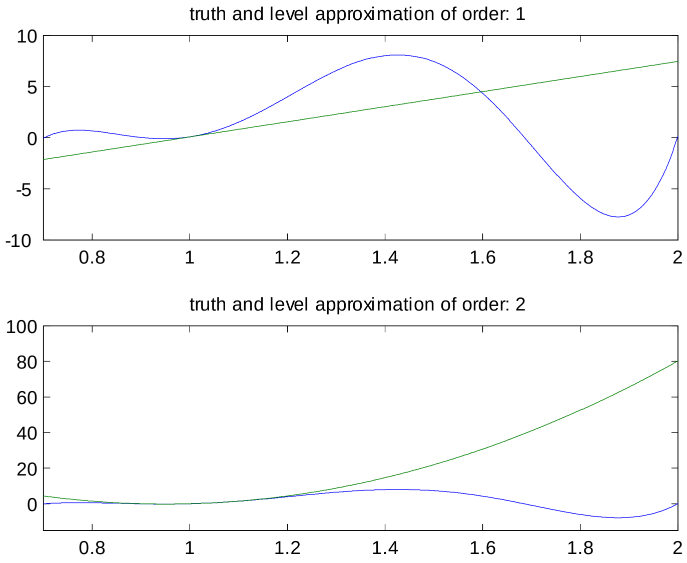


Figure: Level approximations

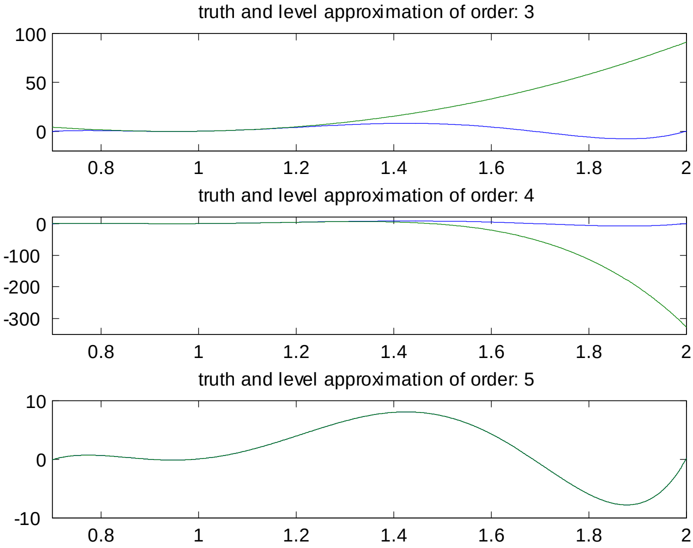


Figure: Level approximations continued

Approximation in log levels

Think of $f(x)$ as a function of $z = \log(x)$. Thus,

$$\begin{aligned} f(x) &= -690.59 + 3202.4\exp(z) - 5739.45\exp(2z) \\ &\quad +4954.2\exp(3z) - 2053.6\exp(4z) + 327.10\exp(5z). \end{aligned}$$

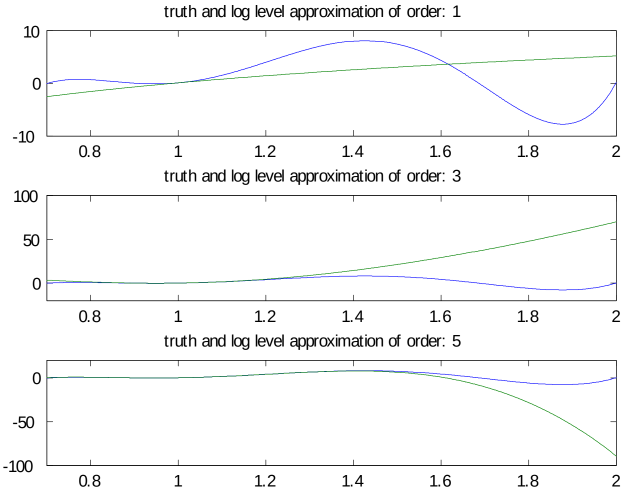


Figure: Log level approximations

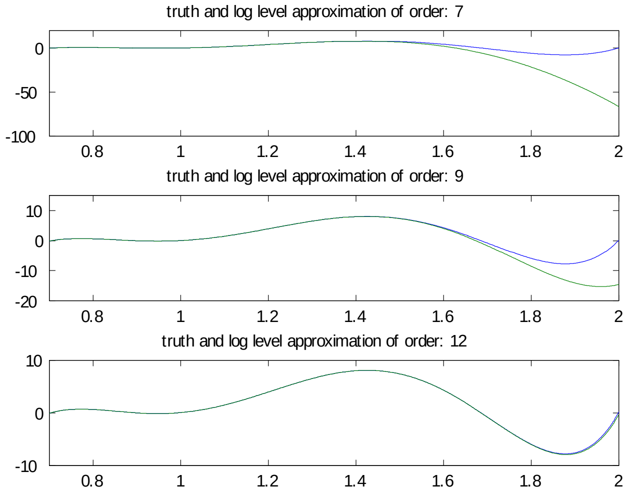
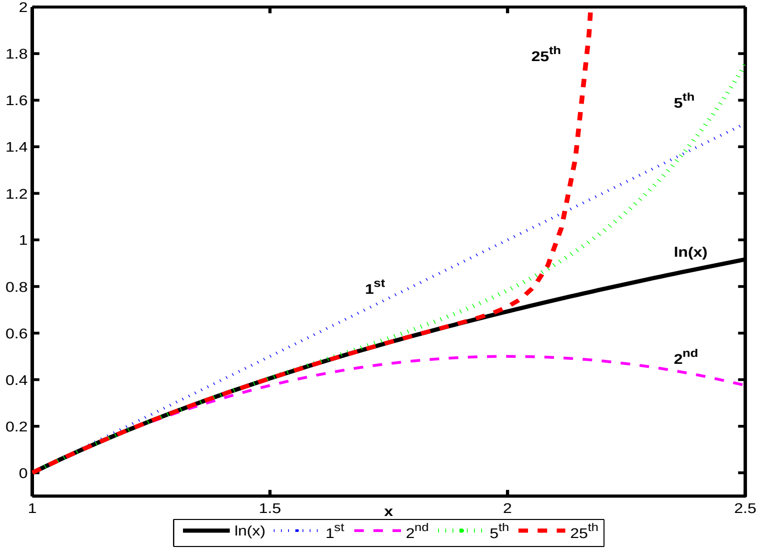


Figure: Log level approximations continued

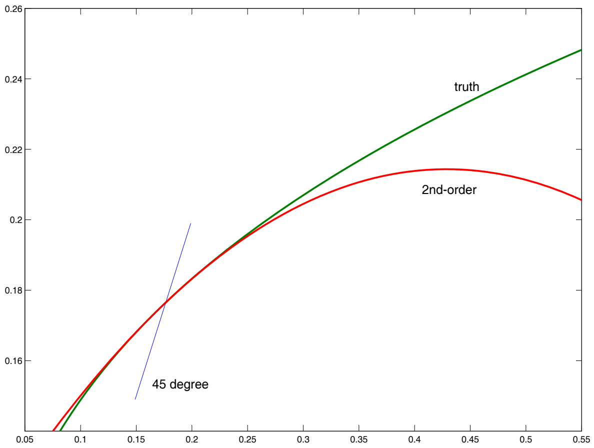
ln(x) & Taylor series expansions at x = 1



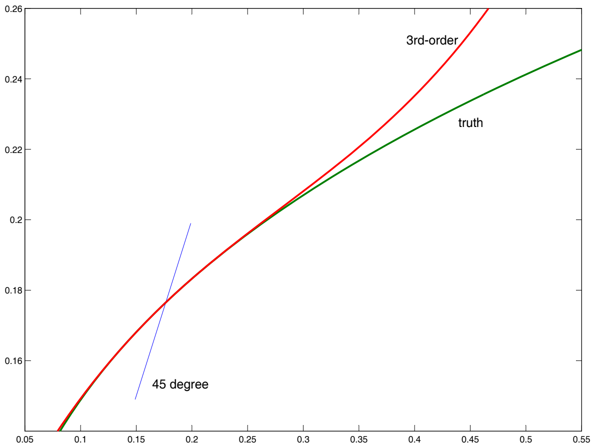
Problems with preserving shape

- nonlinear higher-order polynomials *always* have "weird" shapes
- weirdness may occur close to or far away from steady state
- thus also in the standard growth model

Standard growth model and odd shapes due to perturbation (log utility)



Standard growth model and odd shapes due to perturbation (log utility)

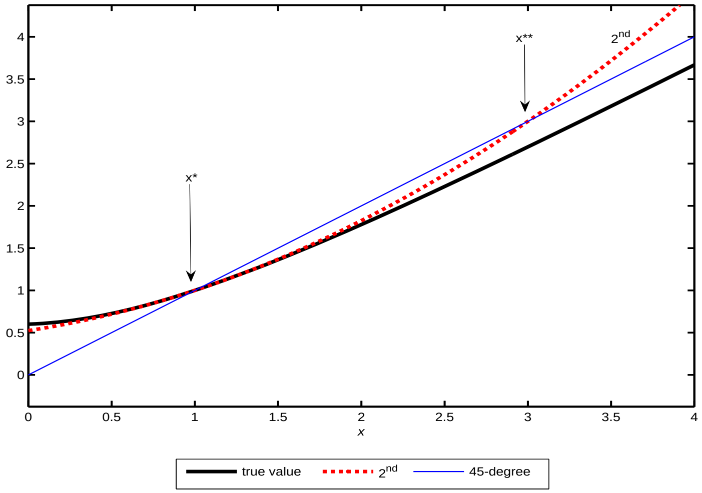


Problems with stability

$$
\begin{aligned}
h(x) &= \alpha_0 + x + \alpha_1 e^{-\alpha_2 x} \\
x_{+1} &= h(x) + \mathsf{shock}_{+1}
\end{aligned}
$$

- Unique globally stable fixed point

Perturbation approximation & stability



How to calculate a simulated data set

Dynare gives you

$$k_t = \bar{k} + a_{k,k}(k_{t-1} - \bar{k}) + a_{k,z_{-1}}(z_{t-1} - \bar{z}) + a_{k,\varepsilon}\varepsilon_t$$

- Start at $k_0 = \bar{k}$ and $z_0 = \bar{z}$ $(= 0)$
- Use a random number generator to get a series for ε_t for $t = 1$ to $t = T$
- Calculate time path for z_t
- Calculate time path for k_t
- Calculate time path for other variables
- Discard an initial set of observations
- Same procedure works for higher-order case
  - except this one could explode

Simulate higher-order & pruning

- first-order solutions are by construction stationary
  - simulation cannot be problematic
- simulation of higher-order can be problematic
- simulation of 2nd-order *will be* problematic for large shocks
- pruning:
  - ensures stability
  - solution used is no longer a policy *function*

Simulate higher-order & pruning

- pruning:
  - ensures stability
  - solution used is no longer a policy *function* of the original state variables
  - also changes the time path if it is not explosive
  - makes it possible to calculate moments analytically (see Andreasen, Fernandez-Villaverde, and Rubio-Ramirez 2014)

Pruning

- $k^{(n)}(k_{-1}, z)$: the n^{th}-order perturbation solution for k as a function of k_{-1} and z.
- $k_t^{(n)}$: the value of k_t generated with $k^{(n)}(\cdot)$.

Pruning for second-order perturbation

- The regular perturbation solution $k^{(2)}$ can be written as

$$k_t^{(2)} - k_{ss}$$
$$=$$
$$a^{(2)} + a_k^{(2)} \left(k_{t-1}^{(2)} - k_{ss} \right) + a_z^{(2)} \left(z_t - z_{ss} \right)$$

$$+ \tilde{k}^{(2)}(k_{t-1}^{(2)}, z_t)$$

Pruning for second-order perturbation

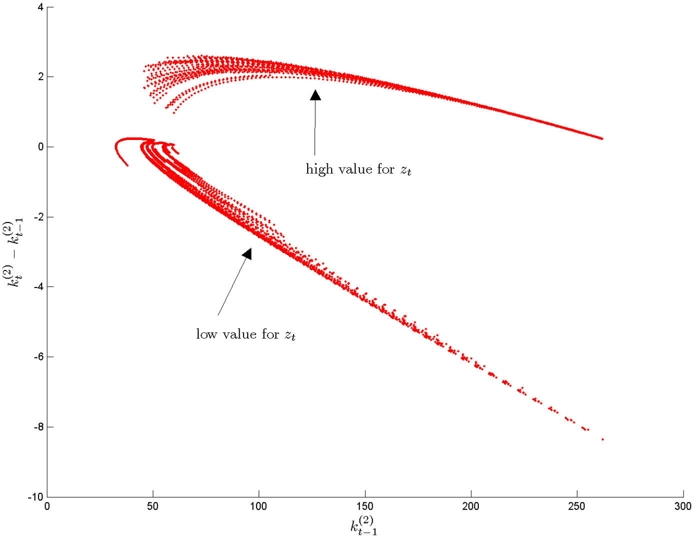
With pruning one would simulate *two* series $k_t^{(1)}$ and $k_t^{(2)}$

$$\mathbf{k_t^{(1)}} - k_{ss} = a_k^{(1)} \left(\mathbf{k_{t-1}^{(1)}} - k_{ss} \right) + a_z^{(1)} \left(z_t - z_{ss} \right)$$

$$\begin{aligned} k_t^{(2)} - k_{ss} = \; & a^{(2)} + a_k^{(2)} \left(k_{t-1}^{(2)} - k_{ss} \right) + a_z^{(2)} \left(z_t - z_{ss} \right) \\ & + \tilde{k}^{(2)} (\mathbf{k_{t-1}^{(1)}}, z_t) \end{aligned}$$

- solution used is $k_t^{(2)}$
- $k_t^{(2)}$ is *not* a function of z_t and $k_{t-1}^{(2)}$, but a function of three state variables!!!

Figure: 2nd-order pruned perturbation approximation for neoclassical growth model; $k_t^{(2)} - k_{t-1}^{(2)}$ as a "function" of $k_{t-1}^{(2)}$



Pruning for second-order perturbation

$$k_t^{(2)} - k_{ss} = \begin{array}{c} a^{(2)} + a_k^{(2)}\left(k_{t-1}^{(2)} - k_{ss}\right) + a_z^{(2)}\left(z_t - z_{ss}\right) \\ +\tilde{k}^{(2)}(\mathbf{k_{t-1}^{(1)}}, z_t) \end{array}$$

- $k_t^{(1)}$ is stationary as long as BK conditions are satisfied
- $\tilde{k}^{(2)}(k_{t-1}^{(1)}, z_t)$ is then also stationary
- $\left|a_1^{(2)}\right| < 1$ then ensures that $k_t^{(2)}$ is stationary

Third-order pruning

- $\tilde{k}^{(3)}(k_{t-1}, z_t)$: part of $k^{(3)}$ with second-order terms
- $\widetilde{\tilde{k}}^{(3)}(k_{t-1}, z_t)$: part of $k^{(3)}$ with third-order terms

$k_t^{(2)}$ is generated as above

$$k_t^{(3)} - k_{ss} = \begin{array}{c} a^{(3)} + a_k^{(3)} \left(k_{t-1}^{(3)} - k_{ss} \right) + a_z^{(3)} \left(z_t - z_{ss} \right) \\ \\ +\tilde{k}^{(3)}(k_{t-1}^{(2)}, z_t) + \widetilde{\tilde{k}}^{(3)}(k_{t-1}^{(2)}, z_t) \end{array}$$

Practical

- Dynare expects files to be in a regular path like e:\... and cannot deal with subdirectories like //few.eur.nl/.../...
- The solution is to put your *.mod files on a memory stick

Practical

- Dynare creates a lot of files
- To delete all those run the `gonzo.m` function.
- In particular:
  - copy gonzo.m in current directory (or directory that is part of your path)
  - if your dynare file is called `modela.mod` use (in command window or in file)

```
gonzo('modela')
```

References

- of course: www.dynare.org
- Andreasen, M.M. J. Fernandez-Villaverde, and J.F. Rubio-Ramirez, 2014, The pruned state-space system for non-linear DSGE models.
- Griffoli, T.M., Dynare user guide
- Den Haan, W.J., Perturbation techniques,
  - Relatively simple exposition of the theory and relation with (modified) LQ.
- Den Haan, W.J., and J. de Wind, Nonlinear and stable perturbation-based approximations equilibrium models
  - discussion of the problems of pruning
- Lombardo, G., Approximating DSGE Models by series expansions
  - derivation of the pruning solution as a perturbation solution