Macroeconomics B Notes

Nicholas Umashev [*]

2019

Contents

[*] Add pages from google Notes

Dynamic Optimisation in Continuous Time

Continuous vs Discrete Time: in discrete time the intervals between periods are $\Delta > 0$ (e.g. the interval between t and $t+1$ is 1), in continuous time the intervals between periods are $\Delta \to 0$

- Continuous Variables Issue: while state variables are often naturally continuous, when using computational methods we can only evaluation the value function on a finite number of points
  - Solution: we can address this problem via using (1) discretization of the state space, or, (2) polynomial approximation to the value function

Discretization Method: consider a problem with continuous state and control variables $\mathfrak{x} \in \mathbb{R}$, discretization just replaces $\mathfrak{x}$ and $\mathfrak{u}$ by the finite grids $\widehat{\mathfrak{x}} = \{x^1, \ldots, x^n\}$ and $\widehat{\mathfrak{u}} = \{u^1, \ldots, u^n\}$

- Value Function Impact: now the value function becomes a finite list of numbers, $V = [V^1, \ldots, V^n]^T$
- Advantage: the maximization step is much simpler than under the original bellman equation, which is a key advantage of discretization methods
- Disadvantage: there is a "curse of dimensionality" in muiltidimensional state spaces where we must decides whether to have N points for a one-dimensional state space vs N^k points for a k-dimensional state space
  - Grid Decision: requires some a-priori information about the state space, which is sometimes difficult to obtain (e.g. upper and lower bounds)
- Optimal Growth Illustration: suppose $V(k) = \max_{k'} \{\ln(Ak^\alpha - k') + \beta V(k')\}$ where in this case $\mathfrak{x} = \mathfrak{u}$
  - Discretization: if we discretize $\mathfrak{x}$ then the Bellman equation becomes $V_i = \max_j \{\ln(Ak_i^\alpha - k_j) + \beta V_j\}$ for all $i = 1, \ldots, n$
  - Value Function Iteration Solution: suppose we applied value function iteration and therefore iterate on the mapping $V_i^s = \max\left\{\ln(Ak_i^\alpha - k_1) + \beta V_1^{s-1}, \ldots, \ln(Ak_i^\alpha - k_n) + \beta V_n^{s-1}\right\}$ for all $i = 1, \ldots, n$ where s indexes the iteration step
    * VF ITERATION

      (1) Start w/ a guess:

      $$\underset{n\times 1}{V^0} = \begin{bmatrix} V_1^0 \\ V_2^0 \\ \vdots \\ V_n^0 \end{bmatrix}, \quad \text{where } V_i^0 \equiv V^0(k_i)$$

      (2) Find V^1 by solving:

      $$V_i^1 = \max_j \left\{\ln(Ak_i^\alpha - k_j) + \beta V_j^0\right\}$$

      expanding

$$V_i^1 = \max \begin{bmatrix} \ln(Ak_i^\alpha - k_1) + \beta V_1^0 \\ \ln(Ak_i^\alpha - k_2) + \beta V_2^0 \\ \vdots \\ \ln(Ak_i^\alpha - k_n) + \beta V_n^0 \end{bmatrix} \quad \forall i$$

so I end up with:

$$\underset{n \times 1}{V^1} = \begin{bmatrix} V_1^1 \\ V_2^1 \\ \vdots \\ V_n^1 \end{bmatrix}$$

*

③ Check for convergence:

$$\underbrace{\|V^1 - V^0\|}_{} \leq \varepsilon \longrightarrow \text{tolerance level}$$

$$\sqrt{\sum_{i=1}^{n} (V_i^1 - V_i^0)^2} \longrightarrow \text{Euclidean Norm.}$$

④ Compute V^s by solving:

$$V_i^s = \max \left\{ \ln(Ak_i^\alpha - k_j) + \beta V_j^{s-1} \right\} \quad \forall i$$

⑤ check convergence

$$\|V^s - V^{s-1}\| \leq \varepsilon \longrightarrow$$

*

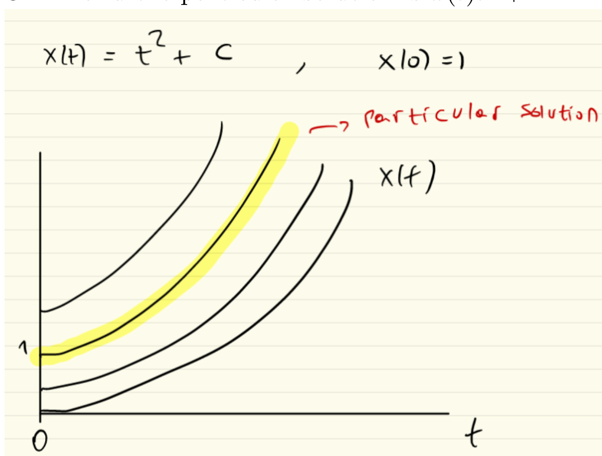
- If $\|V^s - V^{s-1}\| \leq \varepsilon$ holds $\longrightarrow$ STOP
- If $\|V^s - V^{s-1}\| > \varepsilon$ does not hold

$\Rightarrow$ Go Back to step ④

*

<u>**Ordinary Differential Equations (ODEs)**</u>: a ”differential” equation” is one where the unknown is a function (instead of a variable) and the equation includes one or more of the derivatives of the function, an ”ordinary differential equation” equation is one for which the unknown is a function of only one variable (typically time)

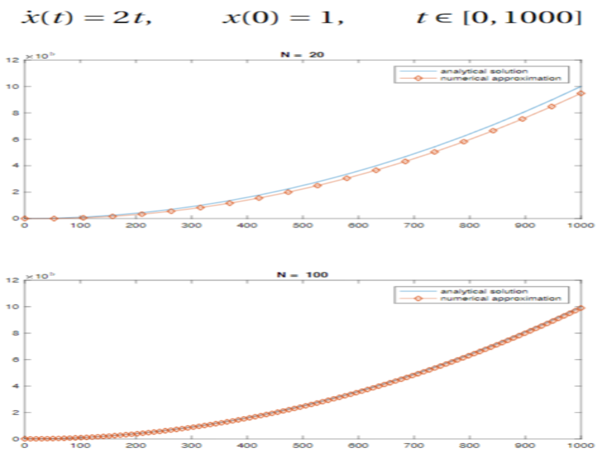
- <u>Partial ODEs</u>: where the unknown is a function of more than one variable
- <u>First-Order ODE Form</u>: $\dot{x}(t) = F(t, x(t))$, where $\dot{x}(t) \equiv dx(t)/dt$ and $t \in [t_a, t_b]$

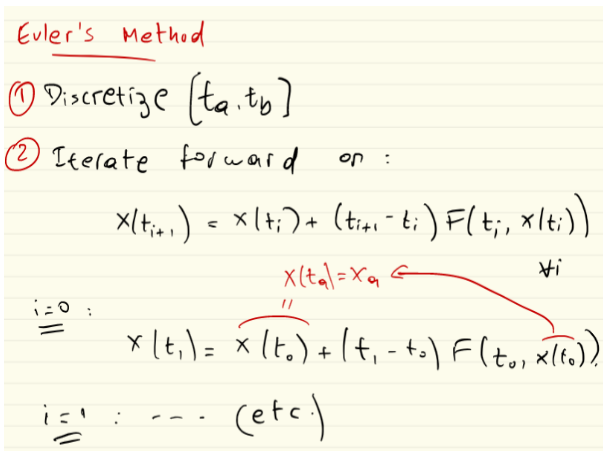
- Unknown: here the unknown is a function $x(t)$ with $x : [t_a, t_b] \to \mathbb{R}$
- Uniqueness: the solution is not unique with the form having infinitely many solutions indexed by an integrating constant C. However, generally the constant C can be uniquely determined by requiring the solution to pass through a given point on the tx-plane
- Example: $\dot{x}(t) = 2t$ with $t \in [0, \infty)$
  * Solution without Initial Condition: the general solution is $x(t) = t^2 + C$
  * Solution with Initial Condition: suppose we impose the initial condition $x(0) = 1$, therefore $C = 1$ and the particular solution is $x(t)t^2 + 1$
  * 

- Finite-Difference Methods for Solution: here we approximate derivates using finite-differences to approximate the solution to the ODE. This involves finding $\dot{x}(t) = F(t, x(t))$, $x(t_a) = x_a$, and $t \in [t_a, t_b]$. There are a wide range of methods depending on how the derivatives are approximated
  - Euler's Method
    * Step 1: specify a grid for t, i.e. $t_0 = t_a < t_1 < t_2 < \cdots < t_N = t_b$
    * Step 2: approximate the ODE using the difference equation

$$\frac{x(t_{i+1}) - x(t_i)}{t_{i+1} - t_i} = F(t_i, x(t_i))$$

over $i = 0, \ldots, N-1$, where $x(t_0) = x_a$ is fixed by the initial condition

* 

* 

Euler's Method

① Discretize $[t_a, t_b]$

② Iterate forward on :

$$x(t_{i+1}) = x(t_i) + (t_{i+1} - t_i) F(t_i, x(t_i)) \qquad \forall i$$

i=0 :

$$x(t_1) = \overset{x(t_a) = x_a}{x(t_0)} + (t_1 - t_0) F(t_0, x(t_0))$$

i=1 : ... (etc.)

- ODE Systems: consider the system of two equations in two unknowns

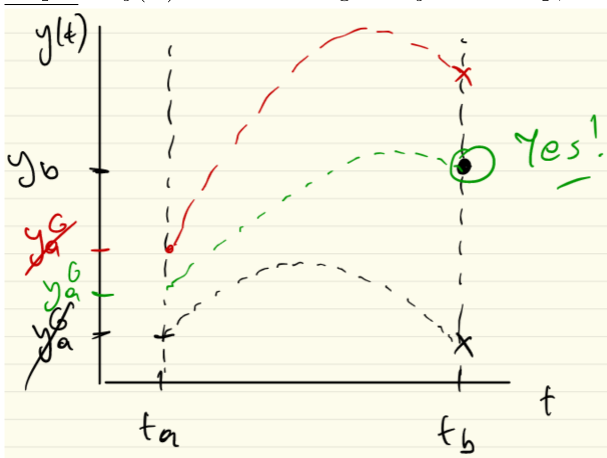
$$\dot{x}(t) = f(t, x(t), y(t))$$
$$\dot{y}(t) = g(t, x(t), y(t))$$

where $t \in [t_a, t_b]$. The general solution typically depends on two arbitrary constants, say A and B, and can be written as

$$x(t) = \phi_1(t; A, B)$$
$$y(t) = \phi_2(t; A, B)$$

Here A and B can be pinned down via two conditions on the solution and two types of problems emerge depending on the nature of such conditions. Note that, like single-equation ODEs, exact solutions to ODE systems can only be obtained under special cases

  - Initial Value Problem(IVP): $x(t_a) = x_a$, $y(t_a) = y_a$, can be solved numerically by applying Euler's Methods
    * Step 1: discretize the domain

    - Step 2: iterate forward starting from $x(t_a) = x_a$ and $y(t_a) = y_a$
  - Boundary Value Problem (BVP): $x(t_a) = x_a$, $y(t_b) = y_b$, can be solved by applying a shooting algorithm
    - Step 1: make an initial guess for $y(t_a)$ called y_a^G
    - Step 2: solve the ODE system by applying Euler's method given $x(t_a) = x_a$ and $y(t_a) = y_a^G$
    - Step 3: if $y(t_b)$ is 'close enough' to y_b then stop, else update y_a^G and go back to step 2
    - 

- Higher-Order ODEs: techniques for solving ODE systems also apply to analysing higher-order Equations
  - Example: consider the second-order ODE

$$\ddot{x}(t) = f(t, \dot{(t)}, x(t))$$

where $\ddot{x}(t) \equiv \partial^2 x(t)/\partial t^2$. Define the new variables $y = \dot{x}$ and you are left with the two-equation ODE system

$$\dot{y}(t) = f(t, y(t), x(t))$$
$$\dot{x}(t) = y(t)$$

The Maximum Principle: the typical continuous time optimization problem is

$$\max_{x(t),y(t)} \int_0^{t_1} f(t, x(t), y(t)) \quad dt$$

subject to

$$\dot{x}(t) = g\big(t, x(t), y(t)\big), \quad x(t) \in \mathcal{X} \ \forall t, \quad y(t) \in \mathcal{Y} \ \forall t, \quad x(0) = x_0$$

There are two main issues with conventional solution methods to this problem; (1) we are choosing over infinitely dimensional objects such as the function $x : [0, t_1] \to \mathcal{X}$, (2) the constaints include a differential equation rather than a set of equalities/inequalities. In order to overcome these issues and find a solution, we apply the maximum principle theorem

- Notation and Assumptions: x is the state variable, y is the control variable, $\mathcal{X} \subset \mathbb{R}$ and $\mathcal{Y} \subset \mathbb{R}$ are nonempty and convex, f and g are continuously differentiable in their arguments, we define a Hamiltonian, and for simplicity we assume $t_1 < \infty$
  - Hamiltonian: we define the hamiltonian

$$H(t, x(t), y(t), \mu(t)) \equiv f(t, x(t), y(t)) + \mu(t) g(t, x(t), y(t))$$

    where $\mu(t)$ is a continuously differentiable function called the costate variable
- Maximum Principle Theorem: suppose that the aforementioned continuous time problem has an interior continuous solution $(\widehat{x}(t), \widehat{y}(t))$. Then there exists a continuously differentiable function $\mu(t)$ such that

$$\begin{gathered} H_y(t, \widehat{x}(t), \widehat{y}(t), \mu(t)) = 0 \ \forall t \in [0, t_1] \\ \dot{\mu}(t) = -H_x(t, \widehat{x}(t), \widehat{y}(t), \mu(t)) \ \forall t \in [0, t_1] \\ \mu(t_1) = 0 \end{gathered}$$

  - ODE Usage [1]: by $H_y(t, \widehat{x}(t), \widehat{y}(t), \mu(t)) = 0$ we can write $y = Y(t, x\mu)$. Using $\dot{\mu}(t) = -H_x(t, \widehat{x}(t), \widehat{y}(t), \mu(t))$ and $mu(t_1) = 0$ and combining the law of motion we are left with

$$\begin{gathered} \dot{x} = g\big(t, x, Y(t, x, \mu)\big) \\ \dot{\mu} = -H_x\big(t, x, Y(t, x, \mu), \mu\big) \\ x(0) = x_0, \quad \mu(t_1) = 0 \end{gathered}$$

    which is an ODE system in x and μ that can be solved numerically by applying the shooting algorithm
  - Sufficient Condition Requirements: the maximum principle only provides necessary conditions for an optimum, sufficient conditions for a maximum rely on concavity properties of the objective and on convexity of the feasible set
  - Technique Generalization: the techniques generalize to multidimensional problems (where x and y are vectors), infinite horizon problems (ie $t_1 = \infty$), and to including terminal conditions on the final state (e.g. $x(t_1) = x_1$)
  - Index: the variable t may represent time or any other index