Chapter 1

Macroeconomic Applications of Mathematical Economics

In this chapter, you will be introduced to a subset of mathematical economic applications to macroeconomics. In particular, we will consider the problem of how to address macroeconomic questions when we are presented with data in a rigorous, formal manner. Before delving into this issue, let us consider the importance of studying macroeconomics, address why mathematical formality may be desirable and try to place into context some of the theoretical models to which you will shortly be introduced.

1.1 Introduction

Why should we care about macroeconomics and macroeconometrics?

Why should we care about macroeconomics and macroeconometrics? Among others, here are four good reasons. The first reason has to do with a central tenet, *viz.* self-interest from the father of microeconomics, Adam Smith

> 'It is not from the benevolence of the butcher, the brewer, or the baker that we expect our dinner, but from their regard to their own interest.' (Wealth of Nations I, ii,2:26-27)

Macroeconomic aggregates affect our daily life. So, we should certainly care about macroeconomics. Secondly, the study of macroeconomics improves our cultural literacy. Learning about macroeconomics can help us to better understand our world. Thirdly, as a group of people, common welfare is an important concern. Caring about macroeconomics is essential for policymakers in order to create good policy. Finally, educating ourselves on the study of macroeconomics is part of our civic responsibility since it is essential for us to understand our politicians.

Why take the formal approach in Economics?

The four reasons given above may have more to do with why macroeconomics may be considered important rather than why macroeconometrics is important. However, we have still to address the question of why formality in terms of macroeconometrics is desirable. Macroeconometrics is an area that fuses econometrics and macroeconomics (and sometimes other subjects). In particular, macroeconometricians tend to focus on questions that are relevant to the aggregate economy (i.e. macroeconomic issues) and either apply or develop tools that we use to interpret data in terms of economics. The question of whether macroeconomics is a science, as opposed to a philosophy say, does not have a straight answer, but the current mainstream economic discipline mostly approaches the subject with a fairly rigorous scientific discipline. Among others, issues that may weaken the argument that macroeconomics is a science include the inherent unpredictability of human behaviour, the issue of aggregating from individual to aggregate behaviour and certain data issues. Both sides of the debate have many good arguments as to why these particular three reasons may be admissible or inadmissible, as well as further arguments on why their angle may be more correct. Maths may be seen to be a language for experts to communicate between each other so that the meaning of their communication is precise. People involved in forecasting, policymakers in governments and elsewhere, people in financial firms, etc. all want estimates and answers to questions including the precision of the answers themselves (hopefully with little uncertainty). In ‘Public Policy in an Uncertain World’, Northwestern University's Charles Manski has attributed the following quote to US President Lyndon B Johnson in response to an economist reporting the uncertainty of his forecast:

> 'Ranges are for cattle. Give me a number.'

Context

Placing macroeconomic modelling in context, such modelling has been important for many years for both testing economic theory and for policy simulation and forecasting. Use of modern macroeconomic model building dates to Tinbergen (1937, 1939). Keynes was unhappy about some of this work, though Haavelmo defended Tinbergen against Keynes. Early simultaneous equations models took off from the notion that you can estimate equations together (Haavelmo), rather than separately (Tinbergen). By thinking of economic series as realisations from some probabilistic process, economics was able to progress.[1]

The large scale Brookings model applied to the US economy, expanding the simple Klein and Goldberger (1955) model. However, the Brookings model came under scrutiny by Lucas (1976) with his critique (akin to Cambpell's law and Goodhart's law). These models were not fully structural models and failed to take account of rational expectations, i.e. they were based upon fixed estimates of parameters. However, when these models were used to determine how people would respond to demand or supply shocks under a new environment, they failed to take into account that the new policies would change how people behaved and consequentially fully backward looking models were inappropriate for forecasting. Lucas summarised his critique as

> 'Given that the structure of an econometric model consists of optimal decision rules of economic agents, and that optimal decision rules vary systematically with changes in the structure of series relevant to the decision maker, it follows that any change in policy will systematically alter the structure of econometric models.'

[1] Watch `http://www.nobelprize.org/mediaplayer/index.php?id=1743`.

While this is not the place to introduce schools of thought in detail regarding their history and evolution, loosely speaking, there are two mainstream schools of macroeconomic thought, namely versions of the neoclassical/free market/Chicago/Real Business Cycle school and versions of the interventionalist/Keynesian/New Keynesian school. Much of what is done today in mainstream academia, central banking and government policy research is of the New Keynesian variant, which is heavily mathematical (influenced by modern neoclassicals (more recent versions of the New Classical Macro School / Real Business Cycle school). Adding frictions to the Real Business Cycle model (few would agree with the basic version, which is simply a benchmark from which to create deviations), one can arrive at the New Keynesian model. The debate is still hot given the recent global financial crisis and European sovereign debt crisis, though there has been a lot of convergence in terms of modelling in recent years given the theoretical linkages aforementioned. Before introducing sophisticated, structural macroeconometric models (DSGE models), let us first spend some time thinking about how to prepare data for such an investigation.

1.2 Data Preparation

Introduction

Econometrics may be thought of as making economic sense out of the data. Firstly, we need to prepare the data for investigation. This section will describe how we might use filters for preparing the data. In particular, we will discuss the use of frequency domain filters. Most of the concepts of filtering in econometrics have been borrowed from the engineering literature. Linear filtering involves generating a linear combination of successive elements of a discrete time signal x_t as represented by

$$y_t = \psi(L)x_t = \sum_j \psi_j x_{t-j}$$

where L is the lag operator defined as follows:

$$\begin{aligned}
x_{t-1} &= Lx_t \\
x_{t-2} &= Lx_{t-1} = LLx_t = L^2 x_t \\
x_{t+1} &= L^{-1} x_t \\
\Delta x_t &= (1-L)x_t
\end{aligned}$$

where the last case is called the 'first-difference' filter. Assuming $|\rho| < 1$

$$\begin{aligned}
x_t &= \rho x_{t-1} + \epsilon_t \\
x_t &= \rho L x_t + \epsilon_t \\
(1-\rho L)x_t &= \epsilon_t \\
x_t &= \frac{\epsilon_t}{1-\rho L}
\end{aligned}$$

$$\begin{aligned}
\frac{1}{1-\rho} &= 1 + \rho + \rho^2 + \cdots \text{ if } |\rho| < 1 \\
\frac{1}{1-\rho L} &= 1 + \rho L + \rho^2 L^2 + \rho^3 L^3 + \cdots \text{ if } |\rho L| < 1
\end{aligned}$$

Why should we study the frequency domain?

As for why one might investigate the frequency domain, there are quite a few reasons including the following. Firstly, we may want to extract that part from the data that our model tries to explain (e.g. business cycle frequencies). Secondly, some calculations are easier in the frequency domain (e.g. auto-covariances of ARMA processes); we sometimes voyage into the frequency domain and then return to the time domain. In general, obtaining frequency domain descriptive statistics and data preparation can be important. For instance, suppose your series is $X_t = X_t^{LR} + X_t^{BC}$ where LR and BC refer to the long-run and the business cycle components, respectively. It only makes sense to split the series into long-run and short-run if the features are independent. In contrast to assuming X_t^{LR} and X_t^{BC} are independent, in Japan X_t^{BC} seems to have affected X_t^{LR}.

Data can be thought of as a weighted sum of cosine waves

We will soon see that we can think of data as a sum of cosine waves. First, let us study the Fourier transform. Moving to the frequency domain from the time domain through the use of the Fourier transform F on a discrete data sequence $\{x_j\}_{j=-\infty}^{\infty}$, the Fourier transform is defined as

$$F(\omega) = \sum_{j=-\infty}^{\infty} x_j e^{-i\omega j}$$

where $\omega \in [-\pi, \pi]$ is the frequency, which is related to the period of the series $\frac{2\pi}{\omega}$.[2] If $x_j = x_{-j}$, then

$$F(\omega) = x_0 + \sum_{j=1}^{\infty} e^{-i\omega j} + e^{i\omega j} = x_0 + \sum_{j=1}^{\infty} 2x_j \cos(\omega j)$$

and the Fourier transform is a real-valued symmetric function. So, the Fourier transform is simply a definition, which turns out to be useful. Given a Fourier transform $F(\omega)$, we can back out the original sequence using

$$x_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega) e^{i\omega j} d\omega = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega)(\cos \omega j + i \sin \omega j) d\omega$$

and if $F(\omega)$ is symmetric, then

$$x_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega) \cos \omega j d\omega = \frac{1}{\pi} \int_{0}^{\pi} F(\omega) \cos \omega j d\omega$$

You can take the Fourier transform of any sequence, so you can also take it of a time series. And it is possible to take finite analogue if time-series is finite. The finite Fourier transform of $\{x_t\}_{t=1}^{T}$ scaled by $\sqrt{T}$ is

$$\bar{x}(\omega) = \frac{1}{\sqrt{T}} \sum_{t=1}^{T} e^{-i\omega t} x_t$$

[2] We could replace the summation operator by the integral if x_j is defined on an interval with continuous support.

Let $\omega_j = (j-1)2\pi/T$ for $j = 1, \ldots, T$. We can vary the frequency as high or low as we want. The finite inverse Fourier transform is given by So, we can move back and forth between the frequency domain and the time series domain through the use of the Fourier transform. Using $\bar{x}(\omega) = |\bar{x}(\omega)|e^{i\phi(\omega)}$ gives (because of symmetry)

$$x_t = \frac{1}{\sqrt{T}} \left(\bar{x}(0) + 2 \sum_{\omega_j \leq \pi} |\bar{x}(\omega_j)| \cos\left(\omega_j t + \phi(\omega_j)\right) \right)$$

Since the Fourier transform involves a cosine, data can be thought of as cosine waves. Mathematically, we can use the inverse Fourier transform to move back from the frequency domain to the time domain to represent the time series x_t. Graphically, we may think of cosine waves increasing in frequency and a mapping from a stochastic time-series $\{x_t\}_{t=1}^{\infty}$. So, we can think of a time-series as a sum of cosine waves. The cosine is a basis function. We regress x_t on all cosine waves (with different frequencies) and the weights $|\bar{x}(\omega_j)|$ measure the importance of a particular frequency in understanding the time variation in the series x_t. We get perfect fitting by choosing $|\bar{x}(\omega_j)|$ and $\phi(\omega_j)$; the shift is given by $\cos\left(\omega_j t + \phi(\omega_j)\right)$. So, we have no randomness, but deterministic, regular cosine waves where x_t is the dependent variable (T observations), $\omega_j t$ are the T independent variables.

Further examples of filters

Briefly returning to filters, we have already seen an example of a filter in the 'first difference' filter $1 - L$. Other examples include any combination of forward and backward lag operators, the band-pass filter (focusing on a range of frequencies and 'turning-off' frequencies outside that range) or the Hodrick-Prescott filter. A filter is just a transformation of the data, typically with a particular purpose (e.g. to remove seasonality or 'noise'). Filters can be represented as

$$x_t^f = b(L)x_t$$
$$b(L) = \sum_{j=-\infty}^{\infty} b_j L^j$$

the latter being an ‘ideal’ filter (one where we have infinite data). Recall the first difference filter $b(L) = 1-L$ implies that $x_t^f = x_t - x_{t-1}$; similarly, another example could be $b(L) = -\frac{1}{2}L^{-1} + 1 - \frac{1}{2}L$. A ‘band-pass’ filter switches off certain frequencies (think of it like turning up the bass on your i-Phone or turning down the treble):

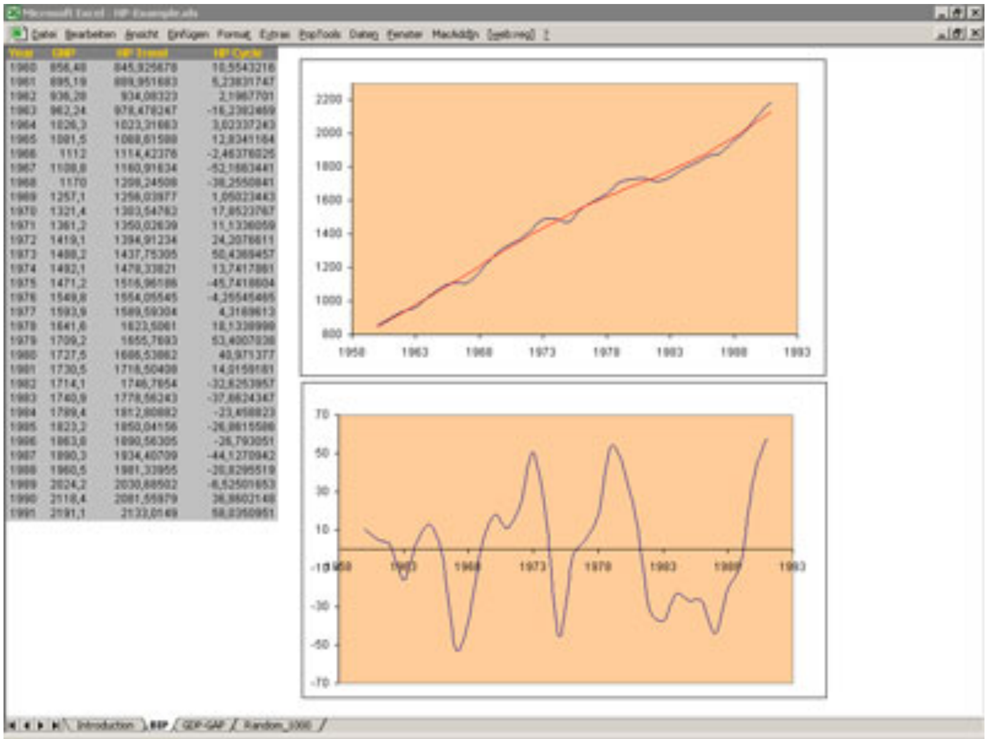
$$y_t = b(L)x_t$$
$$b(e^{-i\omega}) = \begin{cases} 1 & \text{if } \omega_1 \leq \omega \leq \omega_2 \\ 0 & \text{else} \end{cases}$$

Aside: We can find the coefficients of b_j that correspond with this by using the inverse of the Fourier transform since $b(e^{-i\omega})$ is a Fourier transform.

$$\begin{aligned} b_j &= \frac{1}{2\pi}\int_{-\pi}^{\pi} b(e^{-i\omega}e^{i\omega j}d\omega \\ &= \frac{1}{2\pi}\left(\int_{-\omega_2}^{-\omega_1} 1 \times e^{i\omega j}d\omega + \int_{\omega_1}^{\omega_2} 1 \times e^{i\omega j}d\omega\right) \\ &= \frac{1}{2\pi}\left(\int_{\omega_1}^{\omega_2} \left(e^{i\omega j} + e^{-i\omega j}\right) d\omega\right) \\ &= \frac{1}{2\pi}\int_{\omega_1}^{\omega_2} 2\cos(\omega j)d\omega \\ &= \frac{1}{\pi}\frac{1}{j}\sin\omega j|_{\omega_1}^{\omega_2} = \frac{\sin(\omega_2 j) - \sin(\omega_1 j)}{\pi j} \end{aligned}$$

Using l’Hôpital’s rule for $j = 0$ we get

$$b_0 = \frac{\omega_2 - \omega_1}{\pi}$$



Year	GDP	HP Trend	HP Cycle
1960	856,48	845,925678	10,5543216
1961	895,19	889,951683	5,23831747
1962	936,28	934,08323	2,1967701
1963	962,24	978,478247	-16,2382469
1964	1026,3	1023,31663	3,02337243
1965	1081,5	1068,61588	12,8341164
1966	1112	1114,42376	-2,46376025
1967	1108,8	1160,91634	-52,1663441
1968	1170	1208,24508	-38,2550841
1969	1257,1	1256,03977	1,05023443
1970	1321,4	1303,54763	17,8523767
1971	1361,2	1350,02639	11,1336059
1972	1419,1	1394,91234	24,2076611
1973	1488,2	1437,75305	50,4369457
1974	1492,1	1478,33821	13,7417861
1975	1471,2	1516,96186	-45,7418604
1976	1549,8	1554,05545	-4,25545465
1977	1593,9	1589,59304	4,3169613
1978	1641,6	1623,5061	18,1338998
1979	1709,2	1655,7693	53,4007038
1980	1727,5	1686,53862	40,971377
1981	1730,5	1716,50408	14,0159181
1982	1714,1	1746,7654	-32,6253957
1983	1740,9	1778,56243	-37,6624347
1984	1789,4	1812,80882	-23,408823
1985	1823,2	1850,04156	-26,8815586
1986	1863,8	1890,56305	-26,793051
1987	1890,3	1934,40709	-44,1270942
1988	1960,5	1981,33955	-20,8295519
1989	2024,2	2030,88502	-6,52501653
1990	2118,4	2081,55979	36,8602148
1991	2191,1	2133,0149	58,0350951

Figure 1.1: Hodrick-Prescott Filter

The Hodrick-Prescott (HP) trend $x_{\tau,t}$ is defined as follows

$$\{x_{\tau,t}\}_{t=1}^{T} = \arg \min_{\{x_{\tau,t}\}_{t=1}^{T}} \sum_{t=2}^{T-1} (x_t - x_{\tau,t})^2 + \lambda \sum_{t=2}^{T-1} \left\{ [(x_{\tau,t+1} - x_{\tau,t}) - (x_{\tau,t} - x_{\tau,t-1})]^2 \right\}$$

The first term is the penalty term for the cyclical component and the second term penalises variations in the growth rate of the trend component (higher the higher λ is – the smoothing coefficient the researcher chooses); λ is low for annual data [6] and higher for higher frequency (quarterly [1600] / monthly [129,600]) data. The HP filter is approximately equal to the band-pass filter with $\omega_1 = \pi/16$ and $\omega_2 = \pi$, i.e. it keeps that part of the series associated with cycles that have a period less than 32 ($= 2\pi/(\pi/16)$) periods (i.e. quarters).

It is important that when we filter data that we think in the frequency domain. White noise (all frequencies are equally important – has to do with white light) is not serially correlated, but filtered white noise may be serially correlated. Den Hann (2000) considers a demand shock model with positive correlation between prices and demand. However, he shows that filtered price and demand data may not be positively correlated and so when we are examining filtered data, it is important that we reshape our intuition from the raw data to the filtered data, which may be tricky to understand.

1.3 DSGE Models

Introduction

DSGE stands for Dynamic Stochastic General Equilibrium. By equilibrium, we mean that (a) agents *optimise* given preferences and technology and that (b) agent's actions are *compatible* with each other. *General* equilibrium incorporates the behavior of supply, demand, and prices in a whole economy with *several* or many interacting markets, by seeking to prove that a set of prices exists that will result in an overall (or 'general') equilibrium; in contrast, partial equilibrium analyzes single markets only. Prices are 'endogenised' (determined within the model) in general equilibrium models whereas they are determined outside the model (exogenous) in partial equilibrium models. You may encounter famous results this year in EC3010 Micro from general equilibrium theory, studying work by Kenneth Arrow, Gérard Debreu, Rolf Ricardo Mantel, Herbert Scarf, Hugo Freund Sonnenschein and others. Returning to the abbreviation DSGE, the 'S' for stochastic relates to the random nature of systems as opposed to deterministic systems. We have seen this before in problem set 1 question 3; see also Harrison & Waldron footnote 3 in example 1.2.2 and the first paragraph of section 14.4. Mostly, DSGE models are systems of stochastic difference equations. Finally, the word 'Dynamic' signifies the contrast with static models. Allowing variables to evolve over time enables us to explore questions of transition dynamics for instance. Suppose we are interested in changing a pension scheme from pay as you go (you pay for current pensioners and hope to receive the same treatment when you are retired) to fully funded (you pay for your own pension). We may not only be interested in the welfare effects of each scheme but also in how people fare while the scheme is 'in transition'. For example, future pensioners may be better off under the new scheme and future workers may be better off too, but in the transition period, it is likely that current pensioners may be a lot worse off, especially if they suddenly are told they are entitled to no pension!

Stability, Multiplicity and Solutions to Linearised Systems

Introduction

This section explores sunspots, Blanchard-Kahn conditions and solutions to linearised systems. With a model $H(p_{+1}, p) = 0$, a solution is given by $p_{+1} = f(p)$. Figure 1.2 depicts the situation with both a unique solution and multiple steady states. Once reached, the system will remain forever at either of the intersections between the policy function f (curved line) and the 45° line. However, the higher value of p is unstable since if we move slightly away from p above or below, we diverge away from this higher steady state. In contrast, the lower steady state value for p is stable since if we diverge away from this steady state, we will return to it (unless of course we diverge to a level greater or equal to the higher steady state level. Figure 1.3 illustrates the case with multilple solutions an a unique (non-zero) steady state. Figure 1.4 shows the case where there are multiple steady states and sometimes multiple solutions depending on people's expectations; this could be caused by sunspot solutions.

Sunspots in Economics

A solution is a sunspot solution if it depends on a stochastic variable from outside the system.[3] Suppose the model is

$$0 = \mathbb{E}\left[H(p_{t+1}, p_t, d_{t+1}, d_t)\right]$$

$$d_t : \text{exogenous random variable}$$

[3] See the NASA video on sunspots at `https://www.youtube.com/watch?v=UD5VViT08ME`. There was even a 'Great Moderation' in sunspots; see figure 1.6.

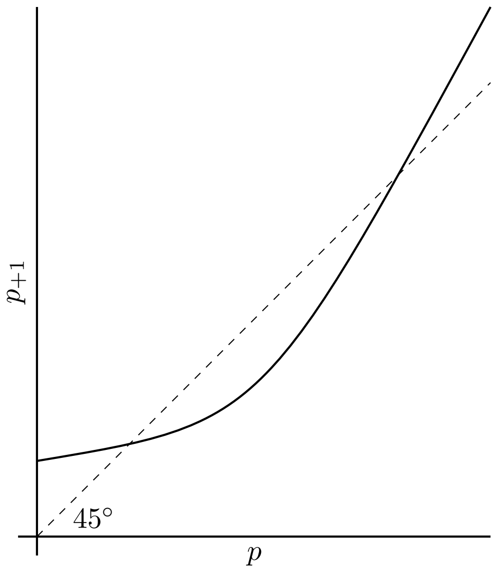


Figure 1.2: Unique solution and multiple steady states

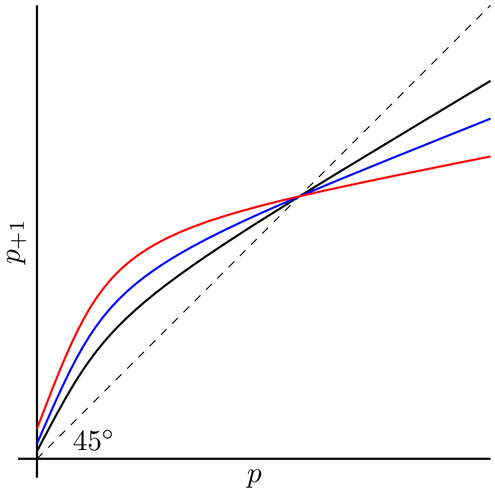


Figure 1.3: Multiple solutions and unique (non-zero) steady state

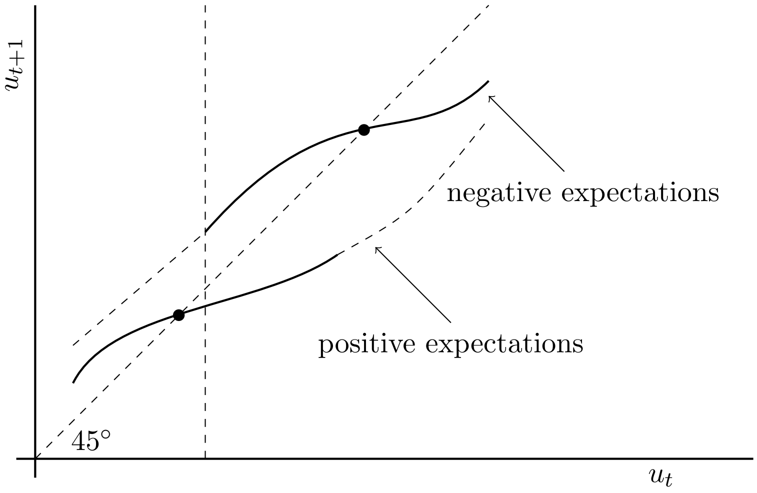


Figure 1.4: Multiple steady states and sometimes multiple solutions

A non-sunspot solution is

$$p_t = f(p_{t-1}, p_{t-2}, \ldots, d_t, d_{t-1}, \ldots)$$

A sunspot solution is

$$\begin{aligned} p_t &: f(p_{t-1}, p_{t-2}, \ldots, d_t, d_{t-1}, \ldots, s_t) \\ s_t &: \text{random variable with } \mathbb{E}\left[s_{t+1}\right] = 0 \end{aligned}$$

Sunspots can be attractive for various reasons including the following: (i) sunspots s_t matter just because agents believe this – after all, self-fulfilling expectations don't seem that unreasonable; (ii) sunspots provide many sources of shocks – this is important because the number of sizable fundamental shocks is small. . On the other hand, sunspots might not be so attractive for other reasons including the following: (i) the purpose of science is to come up with predictions – if there is one sunspot solution, there are zillions of others as well; (ii) support for conditions that make them happen is not overwhelming – you need sufficiently large increasing returns to scale or externalities.

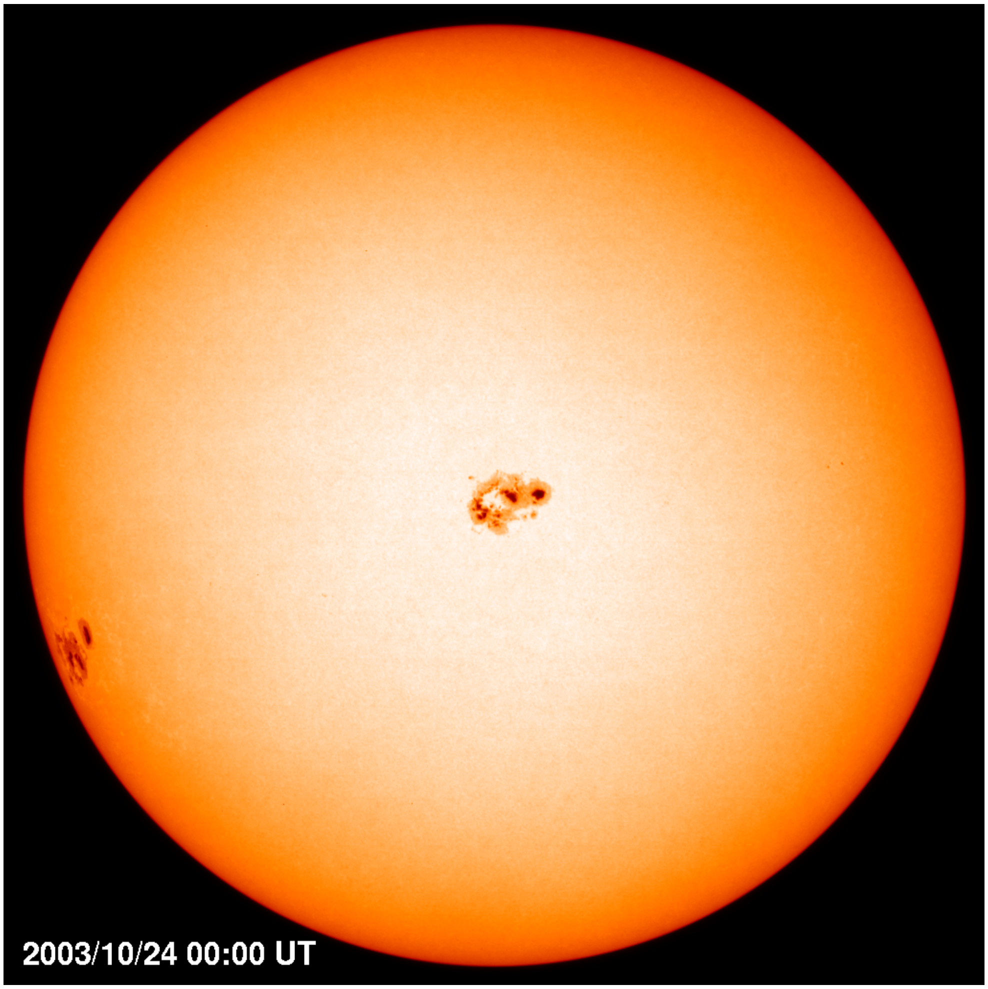
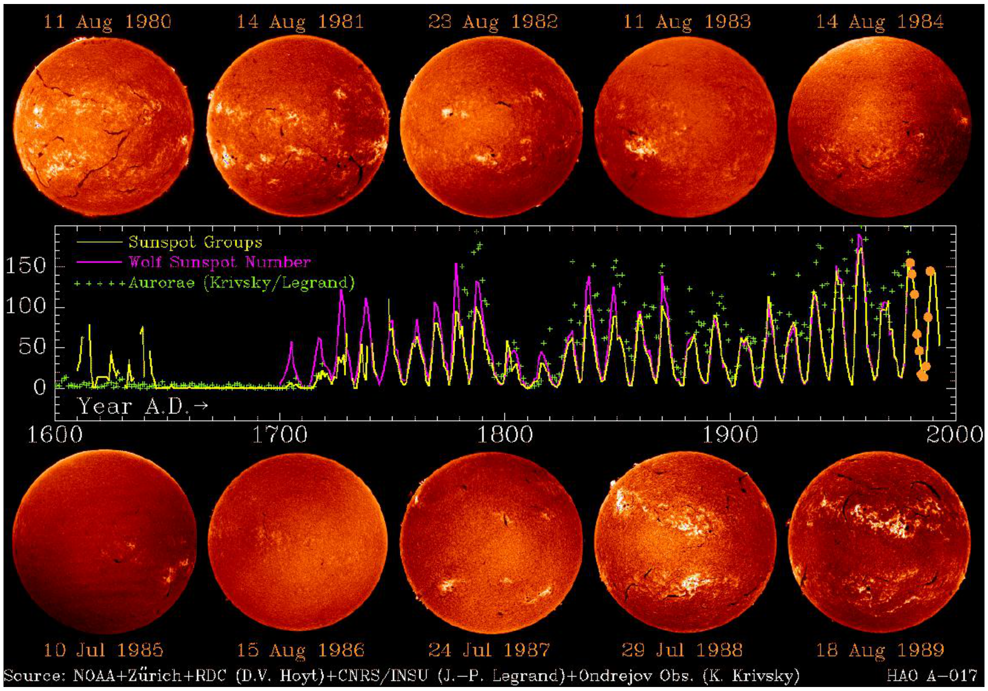


Figure 1.5: Large sunspots (MDI image of sunspot region 10484).



Source: NOAA+Zürich+RDC (D.V. Hoyt)+CNRS/INSU (J.–P. Legrand)+Ondrejov Obs. (K. Krivsky)

Figure 1.6: Past sun spot cycles – sun spots had a 'Great Moderation'.

Blanchard-Kahn conditions

Moving on from sunspots, our goal is to find conditions upon which we have a unique solution, multiplicity of solutions or no stable solutions. Assume we have the following model:

$$\textbf{Model:} \begin{cases} y_{t+1} = \rho y_t \\ y_0 \text{ is given} \end{cases}$$

In this case, we will have a unique solution, independent of the value of ρ. This is because with y_0 given, y_1 will simply be ρy_0, y_2 will be $\rho y_1 = \rho^2 y_0$, etc. So, for any t, $y_t = \rho^t y_0$.

We will soon see the Blanchard-Kahn condition for uniqueness of solutions for the rational expectations model. As a preview of what is to come, the Blanchard-Kahn condition states that the solution of the rational expec-

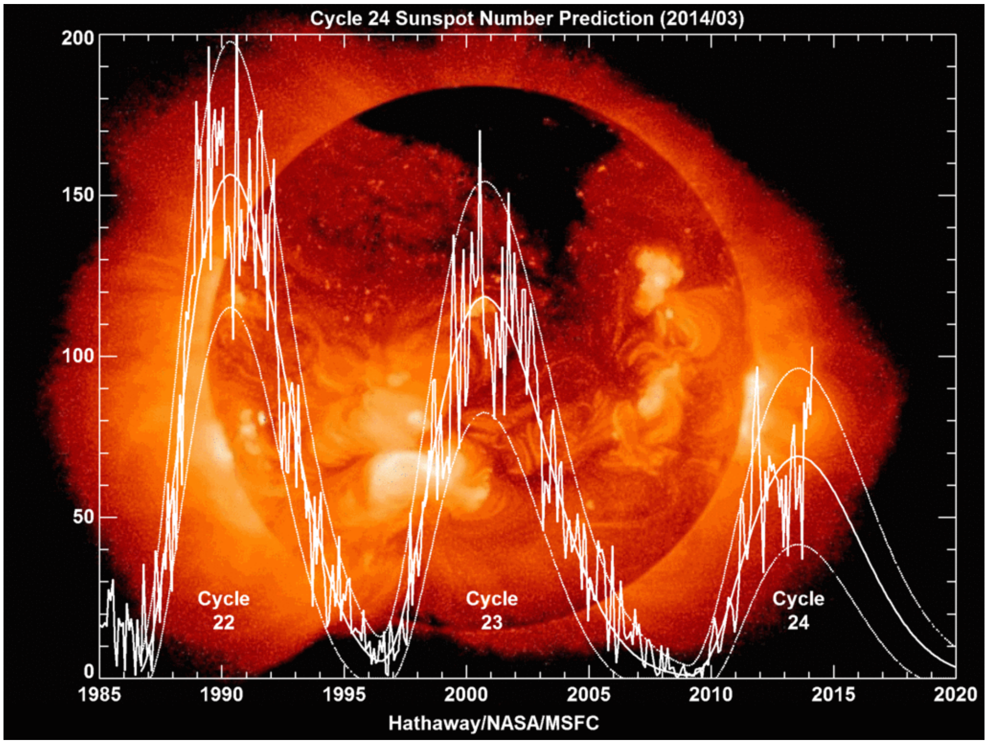


Figure 1.7: Current cycle (at peak again).

tations model is unique if the number of unstable eigenvectors of the system is exactly equal to the number of forward-looking (control) variables. In terms of conditions for uniqueness of solution, multiplicity of solutions or no stable solutions, the Blanchard-Kahn conditions apply to models that add as a requirement that the series do not explode. Now suppose the model is

$$\textbf{Model:} \begin{cases} y_{t+1} = \rho y_t \\ y_t \text{ cannot explode} \end{cases}$$

When $\rho > 1$, we will have a unique solution, namely $y_t = 0$ for all t. This can be seen from setting $\rho = 0$ and obseving that $y_{t+1} = 0 \times y_t$ for all t; hence, $y_t = 0$ for all t. Rewriting the system $\mathbf{A}\mathbf{y}_{t+1} + \mathbf{B}\mathbf{y}_t = \boldsymbol{\epsilon}_{t+1}$ where $\mathbb{E}\left[\boldsymbol{\epsilon}_{t+1}|I_t\right] = 0$, I_t denotes the information set available at time t and $\mathbf{y}_t$ is an

$n \times 1$ vector with $m \leq n$ elements that are not determined

$$\begin{aligned}
\mathbf{y}_{t+1} &= -\mathbf{A}^{-1}\mathbf{B}\mathbf{y}_t + \mathbf{A}^{-1}\boldsymbol{\epsilon}_{t+1} \\
&= \mathbf{D}\mathbf{y}_t + \mathbf{A}^{-1}\boldsymbol{\epsilon}_{t+1} \\
&= \mathbf{D}^t\mathbf{y}_1 + \sum_{l=1}^{t} \mathbf{D}^{l-1}\mathbf{A}^{-1}\boldsymbol{\epsilon}_{l+1}
\end{aligned}$$

where the last equality followed from recursive substitution. With Jordan matrix decomposition

$$\mathbf{D} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^{-1}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with the eigenvalues of $\mathbf{D}$ and assuming without loss of generality that $|\lambda_1| \geq |\lambda_2| \geq \cdots |\lambda_n|$ let

$$\mathbf{P}^{-1} = \begin{bmatrix} \tilde{\mathbf{p}}_1 \\ \vdots \\ \tilde{\mathbf{p}}_n \end{bmatrix}$$

where $\tilde{\mathbf{p}}$ is a $(1 \times n)$ vector. So,

$$\begin{aligned}
\mathbf{y}_{t+1} &= \mathbf{D}^t\mathbf{y}_1 + \sum_{l=1}^{t} \mathbf{D}^{l-1}\mathbf{A}^{-1}\boldsymbol{\epsilon}_{l+1} \\
&= \mathbf{P}\boldsymbol{\Lambda}^t\mathbf{P}^{-1}\mathbf{y}_1 + \sum_{l=1}^{t} \mathbf{P}\boldsymbol{\Lambda}^{t-l}\mathbf{P}^{-1}\mathbf{A}^{-1}\boldsymbol{\epsilon}_{l+1}
\end{aligned}$$

Multiplying the dynamic state system with $\mathbf{P}^{-1}$ gives

$$\mathbf{P}^{-1}\mathbf{y}_{t+1} = \boldsymbol{\Lambda}^t\mathbf{P}^{-1}\mathbf{y}_1 + \sum_{l=1}^{t} \boldsymbol{\Lambda}^{t-l}\mathbf{P}^{-1}\mathbf{A}^{-1}\boldsymbol{\epsilon}_{l+1}$$

or

$$\tilde{\mathbf{y}}_i\mathbf{y}_{t+1} = \lambda_i^t\tilde{\mathbf{p}}_i\mathbf{y}_1 + \sum_{l=1}^{t} \lambda_i^{t-l}\tilde{\mathbf{p}}_i\mathbf{A}^{-1}\boldsymbol{\epsilon}_{l+1}$$

Note that $\mathbf{y}_t$ is $n \times 1$ and $\tilde{\mathbf{p}}_i$ is $1 \times n$, so $\tilde{\mathbf{p}}_i \mathbf{y}_t$ is a scalar. The model becomes

$$\textbf{1: } \tilde{\mathbf{p}}_i \mathbf{y}_{t+1} = \lambda_i^t \tilde{\mathbf{p}}_i \mathbf{y}_1 + \sum_{l=1}^{t} \lambda_i^{t-l} \tilde{\mathbf{p}}_i \mathbf{A}^{-1} \boldsymbol{\epsilon}_{l+1}$$

$$\textbf{2: } \mathbb{E}\left[\boldsymbol{\epsilon}_{t+1} | I_t\right] = 0$$

3: m elements of y_1 are not determined

4: y_t cannot explode

Suppose that $|\lambda_1| > 1$. To avoid explosive behaviour it must be the case that

$$\textbf{1: } \tilde{\mathbf{p}}_1 \mathbf{y}_1 = 0 \text{ and} \tag{1.1}$$

$$\textbf{2: } \tilde{\mathbf{p}}_1 \mathbf{A}^{-1} \boldsymbol{\epsilon}_l = 0 \text{ for all } l \tag{1.2}$$

How should we think about (1.1) and (1.2)? The first equation is simply an additional equation to pin down some of the free elements in $\mathbf{y}_1$; equivalently, this equation is the policy rule in the first period. The second equation pins down the prediction error as a function of the structural shock so the prediction error cannot be a function of other shocks, i.e. there are no sunspots. To see this more clearly, let us look at the example of the neoclassical growth model. The linearised model is

$$k_{t+1} = a_1 k_t + a_2 k_{t-1} + a_3 z_{t+1} + a_4 z_t + e_{E,t+1}$$

$$z_{t+1} = \rho z_t + e_{z,t+1}$$

where k_0 is given and is the end-of-period t capital (so k_t is chosen at time t). Now with (1.1), the neoclassical growth model has $y_1 = [k_1, k_0, z_1]^T$, where $|\lambda_1| > 1$, $|\lambda_2| < 1$ and $\lambda_3 = \rho < 1$, $\tilde{\mathbf{p}}_1 \mathbf{y}_1$ pins down k_1 as a function of k_0 and z_1 (this is the policy function in the first peroid). With (1.2), this pins down $e_{E,t}$ as a function of $\epsilon_{z,t}$, i.e. the prediction error ($e_{E,t}$) must be a function of the structural shock $\epsilon_{z,t}$ and cannot be a function of other shocks, i.e. there are no sunspots. For the neoclassical growth model, $\tilde{\mathbf{p}}_1 \mathbf{A}^{-1} \boldsymbol{\epsilon}_t$ says that the prediction error $e_{E,t}$ of period t is a fixed function of the innovation in period t of the exogenous process $e_{z,t}$. On how to think about the combination of (1.1) and (1.2), without sunspots (i.e. with $\tilde{\mathbf{p}}_1\mathbf{A}^{-1}\boldsymbol{\epsilon}_t = 0$ for all t) k_t is pinned down by k_{t-1} and z_t in *every period.*

The Blanchard-Kahn condition for uniqueness of the solution is that for every free element in $\mathbf{y}_1$, we need one $\lambda_i > 1$; if there are too many eigenvalues larger than one, then no solution will be stable; if there are not enough eigenvalues larger than one, then we will have a multiplicity of solutions.[4] For example, since z_t and k_t are determined before, we need k_{t+1} to be a function of z_t and k_t ($\tilde{\mathbf{p}}_1\mathbf{y}_1 = 0$ [Blanchard-Kahn]) and so k_{t+1} will be determined; else we may have many k_{t+1}.

What if A is not invertible?

In practice it is easy to get $\mathbf{A}\mathbf{y}_{t+1} + \mathbf{B}\mathbf{y}_t = \boldsymbol{\epsilon}_{t+1}$, but sometimes $\mathbf{A}^{-1}$ is not invertible so it is tricky to get the next step, namely $\mathbf{y}_{t+1} = -\mathbf{A}^{-1}\mathbf{B}\mathbf{y}_t + \mathbf{A}^{-1}\boldsymbol{\epsilon}_{t+1}$. The fact that $\mathbf{A}^{-1}$ may not be invertible can be bad news. However, the same set of results can be derived through Schur decomposition; see Klein (2000) and Soderlind (1999).[5] In this case, it is not necessary to get $\mathbf{A}^{-1}$. To repeat, solutions to linear systems using the analysis outlined above requires $\mathbf{A}$ to be invertible, while Klein (2000) provides a generalised version of the analysis above.

Time iteration

We will now apply a solution procedure called time iteration to linearised systems. Consider the model

$$\Gamma_2 k_{t+1} + \Gamma_1 k_t + \Gamma_0 k_{t-1} = 0$$

[4] We can check the Blanchard-Kahn conditions in Dynare through using the command `check;` after the model and initial condition part.

[5] Schur's theorem only requires that $\mathbf{A}$ is square with real entries and real eigenvalues and allows us to find an orthogonal matrix $\mathbf{P}$ and an upper triangular matrix with eigenvalues of $\mathbf{A}$ along the diagonal repeated according to multiplicity where this upper triangular matrix is given by $\mathbf{P}^T\mathbf{A}\mathbf{P}$.

or

$$\begin{bmatrix}\Gamma_2 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}k_{t+1}\\ k_t\end{bmatrix}+\begin{bmatrix}\Gamma_1 & \Gamma_0\\ 1 & 0\end{bmatrix}\begin{bmatrix}k_{t+1} & k_t\end{bmatrix}=0$$

The method outline above implies a unique solution of the form $k_t = ak_{t-1}$ if the Blanchard-Kahn conditions are satisfied. With time iteration, let us impose that the solution is of the form $k_t = ak_{t-1}$ and solve for a from

$$\Gamma_2 a^2 k_{t-1} + \Gamma_1 a k_{t-1} + \Gamma_0 k_{t-1} = 0 \text{ for all } k_{t-1} \tag{1.3}$$

The time iteration scheme can be used, starting with $a_{[i]}$, where time iterations means using the guess for *tomorrow's* behaviour and then solving for *today's* behaviour. Use $a_{[i]}$ to describe next period's behaviour, i.e.

$$\Gamma_2 a_{[i]} k_t + \Gamma_1 k_t + \Gamma_0 k_{t-1} = 0$$

which is different to (1.3). Obtain $a_{[i]}$ from

$$\begin{aligned}(\Gamma_2 a_{[i]} + \Gamma_1)k_t + \Gamma_0 k_{t-1} = 0 &\\ k_t &= -(\Gamma_2 a_{[i]} + \Gamma_1)^{-1}\Gamma_0 k_{t-1}\\ a_{[i+1]} &= -(\Gamma_2 a_{[i]} + \Gamma_1)^{-1}\Gamma_0\end{aligned}$$

As for advantages of time iteration, it is simple even if the **A** matrix is not invertible (the inversion required by time iteration seems less problematic in practice). Furthermore, since time iteration is linked to value function iteration, it has nice convergence properties.

Solving and estimating DSGEs[6][7]

We will begin with a general a specification of a DSGE model. Let $\mathbf{x}_t$ be a $n \times 1$ vector of stationary variables (mean and variance are constant over time), the

[6] This section borrows from DeJong and Dave (2011), which you may want to consult as a reference if you are unsure about what is described in this section.

[7] Dynare is an engine for MATLAB that allows us to solve and estimate DSGE models. See `dynare.org`. One guide that is particularly helpful for economists starting to learn MATLAB/Dynare is available at `http://www.dynare.org/documentation-and-support/user-guide/Dynare-UserGuide-WebBeta.pdf`

complete set of variables associated with the model. The environment and corresponding first-order conditions associated with any given DSGE model can be converted into a nonlinear first-order system of expectational difference equations

$$\mathbf{\Gamma}(\mathbb{E}_t\mathbf{z}_{t+1}, \mathbf{z}_t, \mathbf{v}_{t+1}) = \mathbf{0}$$

where $\mathbf{v}_t$ is a vector of structural shocks and $\mathbb{E}_t\mathbf{z}_{t+1}$ is the expectation of $\mathbf{z}_{t+1}$ formed by the model's decision makers conditional on information available up to and including period t. The deterministic steady state of the model is expressed as $\bar{\mathbf{z}}$ satisfying

$$\mathbf{\Gamma}(\bar{\mathbf{z}}, \bar{\mathbf{z}}, 0) = 0$$

where variables belonging to $\mathbf{z}_t$ can be either exogenous state variables, endogenous state variables or control variables ($\mathbf{c}_t$). The latter $\mathbf{c}_t$ represent optimal choices by decision makers taking as given values of state variables inherited in period t; $\mathbf{c}_t$ is a $n_c \times 1$ vector. Exogenous state variables evolve over time independently of the decision makers' choices, while the evolution of endogenous state variables is influenced by these choices; collectively, state variables are denoted by the $n_s \times 1$ vector s_t, where $n_c + n_s = n$.

From here on, denote the vector $\mathbf{x}_t$ as the collection of model variables written (unless indicated otherwise) in terms of logged deviations from steady state values. So, for a model consisting of output y_t, investment i_t and labour hours n_t, $\mathbf{x}_t$ is given by

$$\mathbf{x}_t = \begin{bmatrix} \tilde{y}_t & \tilde{i}_t & \tilde{n}_t \end{bmatrix}^T$$

where $\tilde{z}_{it} = \ln\left(\frac{z_{it}}{\bar{z}_i}\right)$.

Now we will discuss DSGE model specification in more detail. First we must formulate our DSGE model, which can be cast in either log-linear form or represented as a non-linear model. Log-linear representations of structural models are expressed as

$$\mathbf{A}\mathbf{x}_{t+1} = \mathbf{B}\mathbf{x}_t + \mathbf{C}\mathbf{v}_{t+1} + \mathbf{D}\boldsymbol{\eta}_{t+1} \tag{1.4}$$

where the elements of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and $\mathbf{D}$ are functions of the k structural parameters $\boldsymbol{\mu}$ and $\boldsymbol{\eta}_t$ is an $r \times 1$ vector of expectational errors associated with intertemporal optimality conditions Note that $\boldsymbol{\eta}_t = f(\mathbf{v}_t)$, i.e. expectational errors arise from realisation of structural shocks. Solutions of (1.4) are expressed as

$$\mathbf{x}_{t+1} = \mathbf{F}(\boldsymbol{\mu})\mathbf{x}_t + \mathbf{G}(\boldsymbol{\mu})\mathbf{v}_{t+1} \tag{1.5}$$

In this equation, certain variables in the vector $\mathbf{x}_t$ are unobservable, whereas others are observable (so we need to use filtering methods to evaluate the system empirically). Observable variables are denoted by

$$\mathbf{X}_t = \mathbf{H}(\boldsymbol{\mu})^T\mathbf{x}_t + \mathbf{u}_t \tag{1.6}$$

with $\mathbb{E}\left[\mathbf{u}_t\mathbf{u}_t^T\right] = \boldsymbol{\Sigma}_{\mathbf{u}}$, where $\mathbf{u}_t$ is measurement error. Defining $\mathbf{e}_{t+1} = \mathbf{G}(\boldsymbol{\mu})\mathbf{v}_{t+1}$, the covariance matrix of $\mathbf{e}_{t+1}$ is given by

$$\mathbf{Q}(\boldsymbol{\mu}) = \mathbb{E}\left[(\mathbf{e}_{t+1}\mathbf{e}_{t+1}^T)\right] \tag{1.7}$$

Nonlinear approximations of structural models are represented using three sets of equations, written with variables expressed in terms of levels (possibly normalised to eliminate trend behaviour), which are (i) the laws of motion for the state variables $\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{v}_t)$, (ii) the policy functions representing optimal specification of the control variables in the model as a function of the state variables $\mathbf{c}_t = c(\mathbf{s}_t)$ and (iii) a set of equations mapping the full collection of model variables into the observables $\mathbf{X}_t = g(\mathbf{s}_t, \mathbf{u}_t)$ where $\mathbf{u}_t$ denotes measurement error.

As for model solution techniques, there are linear solution techniques and non-linear solution techniques. Linear solution techniques include Blanchard and Kahn's method, Sims' method, Klein's method and using the method of undetermined coefficients. Nonlinear solution techniques include projection (global) methods (e.g. finite element methods and orthogonal polynomials), iteration (global) techniques such as value function iteration and policy function iteration and perturbation (local) techniques. Solutions to log-linear model representations are expressed as in (1.5). Solutions to the log-linear system (1.5) can be converted to a form analogous to policy functions for non-linear systems by solving for the control variables contained in $\mathbf{x}_t$ as functions of the state variables contained in $\mathbf{x}_t$.

As for estimation, given assumptions regarding the stochastic nature of measurement errors and the structural shocks, (1.5)–(1.7) yield a log-likelihood function $\log L(\mathbf{X}|\mathbf{\Lambda})$, where $\mathbf{\Lambda}$ collects the parameters in $\mathbf{F}(\boldsymbol{\mu})$, $\mathbf{H}(\boldsymbol{\mu})$, $\mathbf{\Sigma_u}$, and $\mathbf{Q}(\boldsymbol{\mu})$. For non-linear model representations, parameters associated with $f(\mathbf{s}_{t-1}, \mathbf{u}_t)$, $c(\mathbf{s}_t)$ and $g(\mathbf{s}_t, \mathbf{u}_t)$ are also obtained as mappings from $\boldsymbol{\mu}$, so their associated likelihood function is also written as $L(\mathbf{X}|\boldsymbol{\mu})$.

To estimate the models after solving them, we typically first need to calibrate certain parameters usually through using microeconometric studies, or by implications for various quantities of interest or through moment matching; in fact, after formulating our DSGE model, we generally solve for steady state, then solve the model having fixed the calibration of many if not all the parameters and those that we have not fully calibrated, we allow to vary, solving for each set of parameter calibrations and potentially estimating the likelihood of the data given the model and maximising this or minimising the difference (or some function) between the moments found in the data and the moments implied by the simulations arising from the model that is solved and estimated for each particular set of parameter calibrations. Moment matching methods such as GMM, SMM and indirect inference allow us to estimate DSGE models. Similarly, we can use maximum likelihood to estimate DSGE models. We can also evaluate likelihood and filter in state-space representations through Bayesian methods using Sequential Monte Carlo methods and Markov Chain Monte Carlo methods. We may be interested in the moments of the model, telling us how good our model may be relative to the data, variance decompositions to decompose the aggregate volatility into the different shocks and impulse response functions tracing out the evolution of control variables or other variables or transformations of variables in response to shocks to the model.

1.4 Other Quantitative Tools

Data classification

Data may be classified as time-series, cross-section or panel. In the first case, data may be arranged over time, e.g. national income data, interest rate data, Irish Consumer Price Indices, etc. In the second case, data varies across individual units (people, firms, industries, countries, etc.) at and we only look at a given point in time, e.g. reports on surveys conducted in a given year across a large sample of households. In the third case, we have access to both variation in the time-dimension and the cross-section dimension, e.g. quarterly consumption expenditure for each member of the euro area over 2000Q1 to 2010Q4, say. In your further studies, you may encounter various cross-section, time-series and panel data techniques to deal with issues arising from the data.

Bayesian Econometrics[8]

While traditionally, you may be schooled in the Classical, or more appropriately Frequentist approach to statistics and econometrics, an alternative school of thought deserves mentioning. Let us motivate this approach with an example. We may know the ratio of faulty cars to good cars produced from a particular Ford factory, say, but we would like to know the probability that a faulty Ford car we are driving happens to have been produced by that factory. Fortunately, there is a formula that allows us to 'back-out' this probability, once we know (or specify) other probabilities such as the fraction of Ford cars that are faulty, the fraction of Ford cars produced by that particular Ford factory and the fraction of faulty Ford cars produced by that factory. Named after Reverend Thomas Bayes (1701–1761), Bayes' rule states

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

[8] You should consult `http://sims.princeton.edu/yftp/EmetSoc607/AppliedBayes.pdf`.

This shows us what the probability of A is given we observe B. Returning to our example, A is the event that the car was produced by the particular Ford factory and B is the event that the car is faulty.

What if we do not have figures on the overall probability of faulty cars? No problem! Bayes rule is a method of transforming subjective, prior beliefs into posterior probabilities once we observe data. We may specify a prior probability distribution on $P(A)$ (faulty cars) and simply use Bayes' rule to update our prior belief to a posterior probability. The more strength we place on the prior distribution, the less we allow our data to speak ($P(B|A)$ or the Frequentist likelihood e.g. $L(data|parameters)$) and the more the posterior distribution will look like the prior distribution. This subjectivist probability philosophy clearly differs from Frequentist ideology, where probabilities are thought to be the frequency of occurrence if we had a very large sample. But what about once-off events? Frequentists have a hard time explaining these.

The Nobel Laureate Christopher Sims at Princeton University is one vocal advocate for increasing use of Bayesian methodology in economics and econometrics. You should take a look at his bullet-point argument on 'Bayesian Methods in Applied Econometrics, or, Why Econometrics Should Always and Everywhere Be Bayesian'.[9]

Non-linear models[10]

While this course has predominantly focused on linear models, these are only a tiny subset of the entire range of possible models from which we can choose. Clearly, any model that is not a linear model is by definition a non-linear model, so non-linear models are more general.[11] Most DSGE models are non-linear. Linearised versions with Normal shocks are chosen typically since

[9] This is available at http://sims.princeton.edu/yftp/EmetSoc607/AppliedBayes.pdf.

[10] You should consult 'Why Non-Linear/Non-Gaussian DSGE Models?' available at http://www.nber.org/econometrics_minicourse_2011/.

[11] Most of this section on non-linear models relies upon the video and slides on 'Why Non Linear / Non-Gaussian DSGE Models?' available at http://www.nber.org/econometrics_minicourse_2011/.

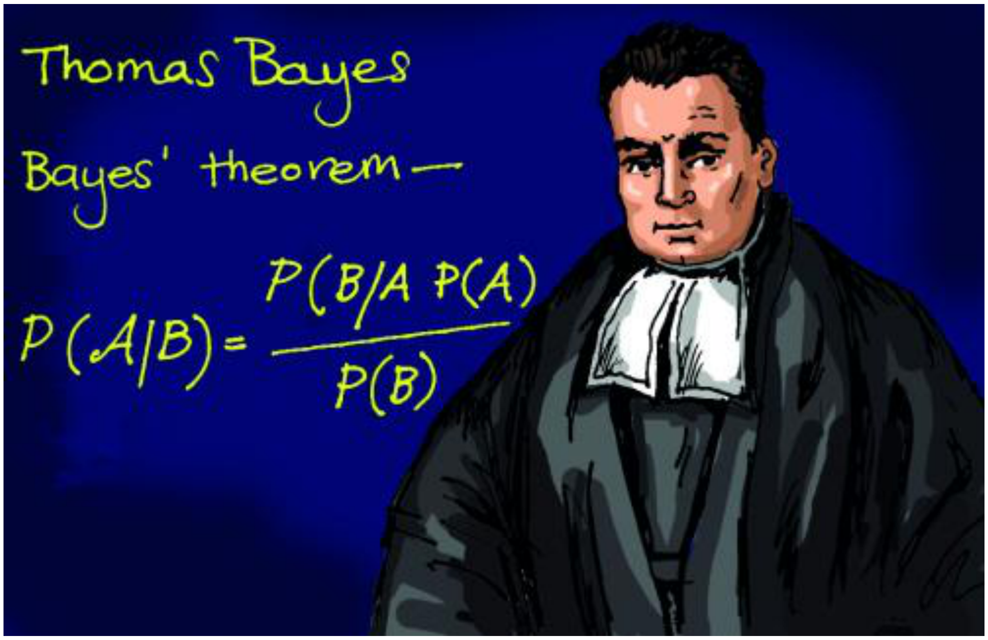


Figure 1.8: Thomas Bayes – Bayes' Rule

the stochastic neoclassical growth model is nearly linear for the benchmark calibration.[12]

Three examples serve to illustrate cases where non-linear (and non-Normal) models may be necessary. Firstly, non-linear models are necessary to deal with a more general (exotic) set of preferences in macroeconomics. For example, recursive preferences like Kreps-Porteus-Epstein-Zin-Weil, which are popular in accounting for asset pricing observations, allow risk aversion and the intertemporal elasticity of consumption substitution to move independently. Secondly, volatility shocks require higher-order (e.g. third-order perturbation) solution methods and non-linear econometric models such as stochastic volatility (non-Normal) further motivate non-linear modelling.[13] Thirdly, studying the impact on the Great Moderation from the mechanisms of fortune or virtue requires non-linear modelling.

[12] For example, Aruoba, Fernández-Villaverde & Rubio-Ramírez (2005) shows this fact.

[13] Two examples of this strategy are Fernández-Villaverde *et al* (2011) and Curran (2014).

Furthermore, linearisation limits studies, eliminating phenomena of interest such as asymmetries, threshold effects, precautionary behaviour, big shocks, convergence away from steady states, etc. Related to this point, linearisation limits our study of dynamics e.g. we need non-linear models to analyse the zero-lower bound on the nominal interest rate, finite escape time, multiple steady states, limit cycles, subharmonic, harmonic or almost-periodic oscillations and chaos.

Additionally, linearisation induces approximation error (worse than you would think). Theoretical arguments include the fact that second-order errors in the approximated policy function induce first-order errors in the log-likelihood function, the fact that as the sample size grows the error in the likelihood also grows and we may have inconsistent point estimates and the fact that linearisation complicates the identification of parameters; there is also some computational evidence on this.

Arguments against non-linear models include the following: (i) theoretical reasons: we know far less about non-linear and non-Normal systems; (ii) computational limitations; and (iii) bias (Maslow's Hammer 'to a man holding a hammer, everything looks like a nail' and Teller's Law: 'a state-of-the-art computation requires 100 hours of CPU time on the state-of-the-art computer, regardless of the decade').

1.5 Computers

A brief history of computing

In computing, floating point operations per second (FLOPS) is a measure of computing performance. As a brief history of computing, invented around 3000BC, the abacus is capable of 1FLOP. Pascal's adding machine (1642) was ahead of its time and during the industrial revolution, ancient devices gave way to new calculating tools, mechanical calculators, punched card data processing and eventually calculators with electric motors. Early mechanical computing machines included Babbage's difference engine (1832). Of course, the twentieth century brought with it huge improvements with the invention of the diode in 1904 and transistors in 1947. The early to mid twentieth century was a time of rapid change in computing history. Analog computers were developed, then electromagnetic computers and subsequently digital computers ultimately with electronic data processing. Alan Turing first described the principle of the modern computer in 1936. Later, Turing would work on deciphering the German Enigma code during WWII – the new film 'The Imitation Game' depicts this part in Turing's life.[14] The economist William Phillip's MOnetary National Income Analog Computer (MONIAC) was created in 1949 to model the national income economic processes of the UK while he was still a student at the London School of Economics. Modern computers really arrived with the likes of Colossus and then ENIAC. As part of the war effort for code breaking in Britain, Colossus was the world's first electronic digital computer that was programmable and flexible. Stored-program computers replaced fixed program machines (e.g. desk calculators). Transistors replaced vacuum tubes. Early computers had switches and tubes that had to be physically changed for computations and with all the sparks they often created, insects would be attracted towards the electronic parts, which may often have caused these early computers to crash, due to 'bugs', hence the terminology we still have today.

FORmula TRANslation (or Fortran) dates to 1957, a higher level language to assembly language, which is almost perfectly machine readable but hardly intelligible to humans. The language in its current form is still one of the most popular languages amongst the scientific community. Established in 1969, Arpanet was the progenitor of today's internet, funded by the US Department of Defense. Unix is a multitasking, multiuser operating system (i.e. managing computer hardware and software resources), developed at AT& T's Bell Labs by Ken Thomson, Dennis Ritchie and others. The C programming language was designed by Dennis Ritchie as a systems programming language for Unix. This operating system took off in academia and elsewhere partly as open

[14] See `http://www.youtube.com/watch?v=Fg85ggZSHMw` for a trailer. Marian Rejewski developed the first mechanical bomba (1938), which was vastly improved upon by Turing and others before the invention of Colossus. There were many competing machines on top of the secrecy that prevailed so it is non-trivial to clearly identify a general first modern computer, though easier to mark the first specific types of computers.

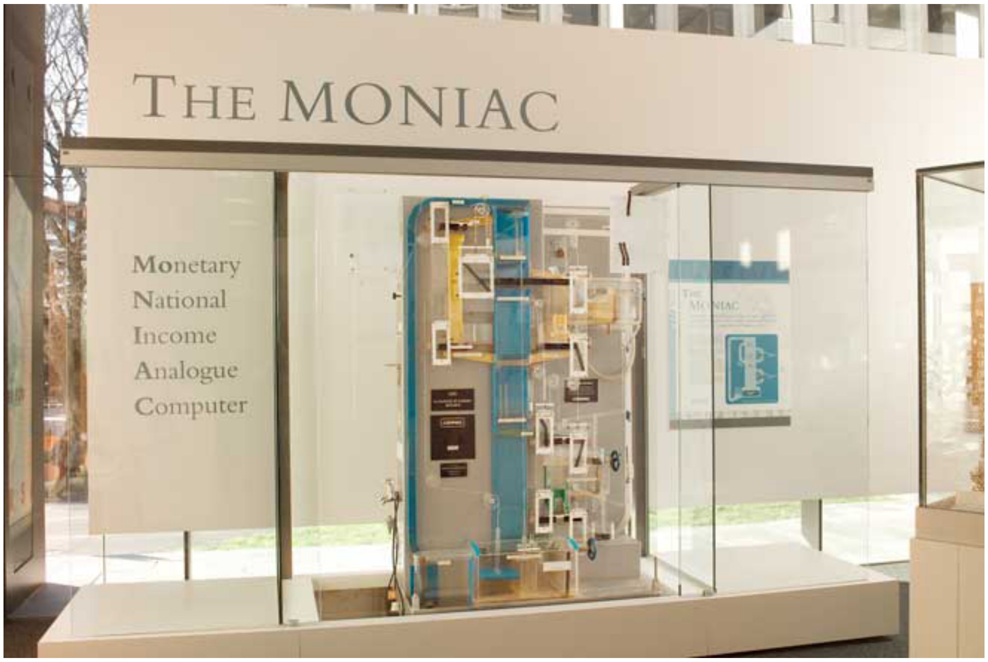


Figure 1.9: MONIAC.

source (free) and partly as commerical vendors developed versions of Unix. Supercomputers sprung from the 1962 development of the Atlas Computer and the first supercomputer designed by Seymour Cray in 1964.

Returning to our measure of performance, the 1976 Cray I was capable of 100 MFLOPS (10^6), which by 1983 had increased to 570 MFLOPS in the form of the SX-1. The economist Enrique Mendoza used an ETA-10P in 1987 capable of 750 MFLOPS for his PhD dissertation 'Real Business Cycles in Small Open Economies'. In 1994, Message Passing Interface (MPI) language for parallel programming over distributed memory systems (clusers of different computers) was released. In 1997, OpenMP language for parallel programming over shared memory systems (your own PC – e.g. if you have two, four or more cores/processors in your laptop/PC, etc.). By 2001, supercomputers were delivering performance of 40 TFLOPS (10^{12}) in the form of the Earth Simulator computer. This number had increased to 150 TFLOPS in 2005 (BGL), 1 PFLOP in 2008 (10^{15} – Roadrunner) and 33.86 PFLOPS in early

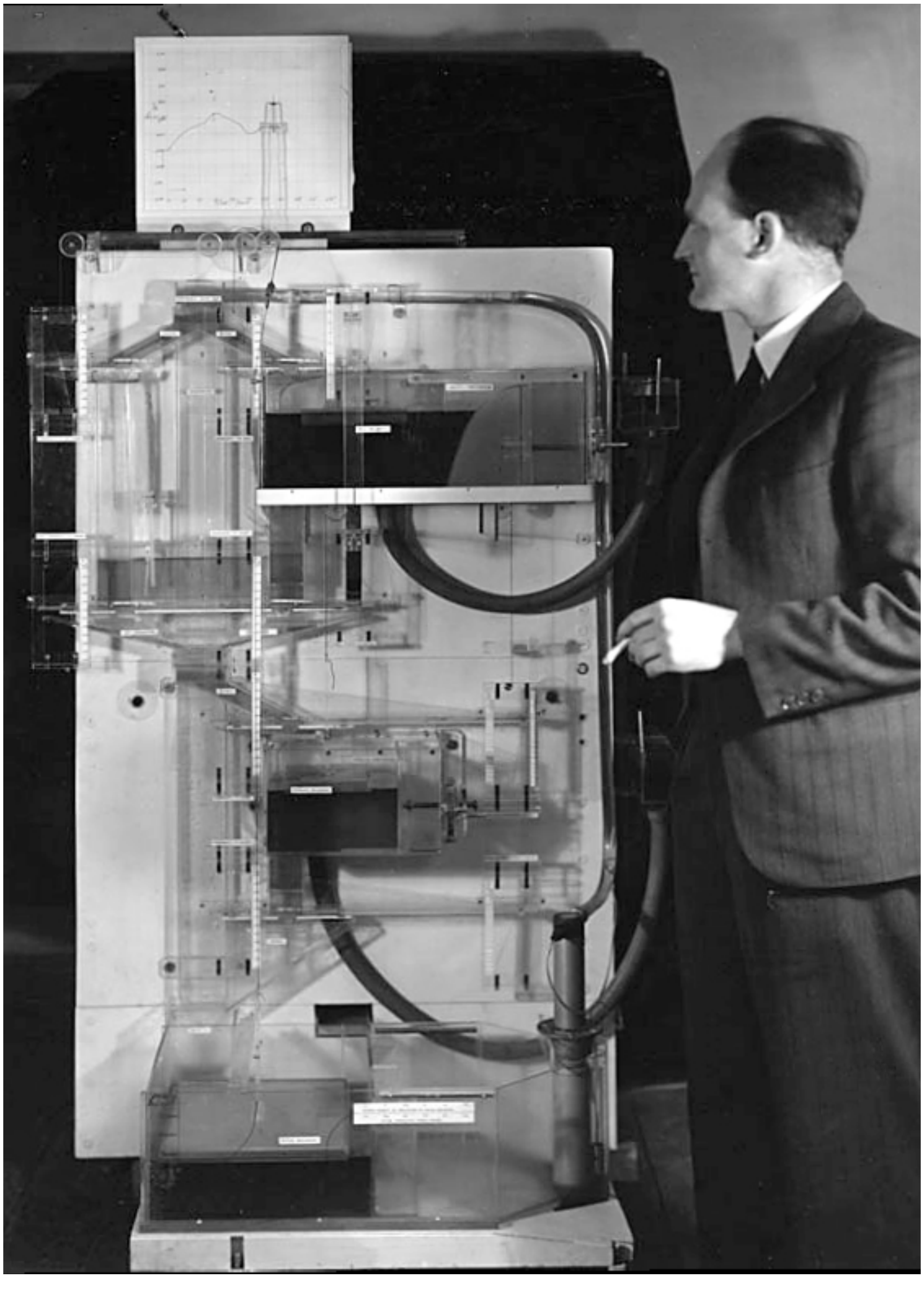


Figure 1.10: MONIAC and Phillips.

Figure 1.11: China's Tianhe-2 (MilkyWay-2).

2013 (quadrillions of calculations per second – China's Tianhe-2 [MilkyWay-2]). The top 500 is a list updated semi-annually of the most powerful computer systems in the world.[15] This list is announced at the International Supercomputing Conference (ISC) in June and at the Supercomputing Conference (SC) in November. The rankings are based on the Linpack benchmark results. Essentially, the computer systems solve a set of dense equations, which is a good test of processor performance. People do not agree on this test since for instance it does not test memory or interconnect but it has become the standard. Check out the latest results, which are available from Tuesday November 18th 2014 at www.top500.org.

[15] The list is available at www.top500.org. The 2013 'Fionn' run by ICHEC at WIT was the only Irish system on the list in June 2014, placing about 476 at 140 TFLOPS; as of November 2014, there are no Irish systems on the top500 list. Trinity College's TCHPC maintains 'Lonsdale', capable of 11 TFLOPS and also available to researchers. NVIDIA's graphics cards are capable of over 8 TFLOPS each (see more below).

Simulation

In many economic models you will have seen, solutions have generally been analytical. Such tractability typically hinges upon restrictive assumptions such as the representative agent assumption. This assumption in economics posits that we can study the aggregate performance of an economy from assuming that there is only one type of consumer or firm or at least that the outcome is the same 'as if' there were many similar agents populating the economy. When we want to look at models of heterogenous agents (different beliefs, different types such as ability or access to information, etc.), it is generally not as straightforward to aggregate from microeconomic models to macroeconomic models.[16] To do so requires keeping track of the distribution of firms with different levels of capital say. Numerical simulation is necessary for many more advanced problems such as these.

The law of large numbers states that the average of the results from a large number of trials should be close to the expected value and will tend to become closer as more trials are performed. See figure 1.12 for an example for rolling a dice. With six possible outcomes $\{1, 2, 3, 4, 5, 6\}$, the mean value is $\frac{1+2+3+4+5+6}{6} = 3.5$, towards which is what we tend to see the average of a large number of throws converging. The central limit theorem states that for a sufficiently large number of draws of random variables, the resulting distribution from the draws will be approximately normal irrespective of the underlying distribution for the original random variables. See figure 1.13 for an illustration of the distribution becoming smoother and tending towards a Normal distribution as we sum over more and more variables. Numerical simulation techniques make use of laws of large numbers (there are various laws depending on what we can assume about the data) and central limit theorems (there are multiple theorems for specific cases of what we can assume about the data, e.g. whether the sequence of the random variables is independent or not).

[16] Some other advanced topics in mathematical economics include learning in macroeconomics, e.g. Bayesian learning (updating) and other deviations from rationality, e.g. bounded rationality.

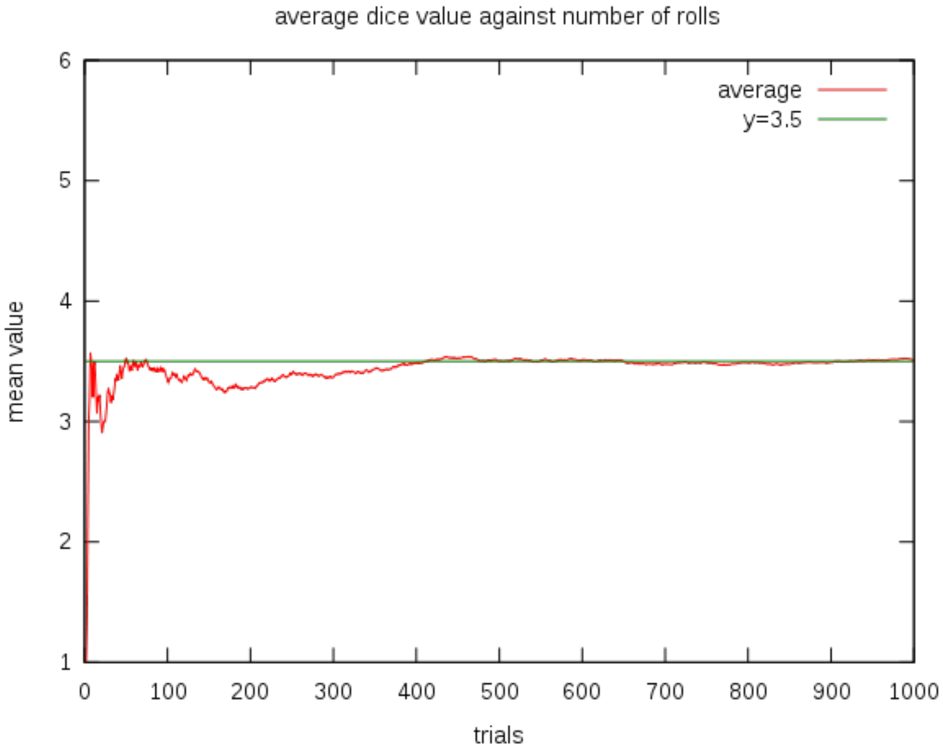


Figure 1.12: Illustration of law of large numbers.

Graphical programming[17]

As for the choice between graphical programming, parallel programming and serial programming, Aldrich *et al* (2011) 'Tapping the Supercomputer Under Your Desk: Solving Dynamic Equilibrium Models with Graphics Processors' show how to build algorithms that use graphical processors (GPUs) to solve dynamic equilibrium models in economics, suggesting significant speed up for time-consuming problems (hours to seconds) and allowing researchers answer questions where models were previously intractable, explore more parts of the state-space and achieve better convergence with lower errors. CUDA code for

[17] You should consult 'Tapping the supercomputer under your desk: Solving dynamic equilibrium models with graphics processors' by Aldrich, Fernández-Villaverde, Gallant and Rubio-Ramírez (2011).

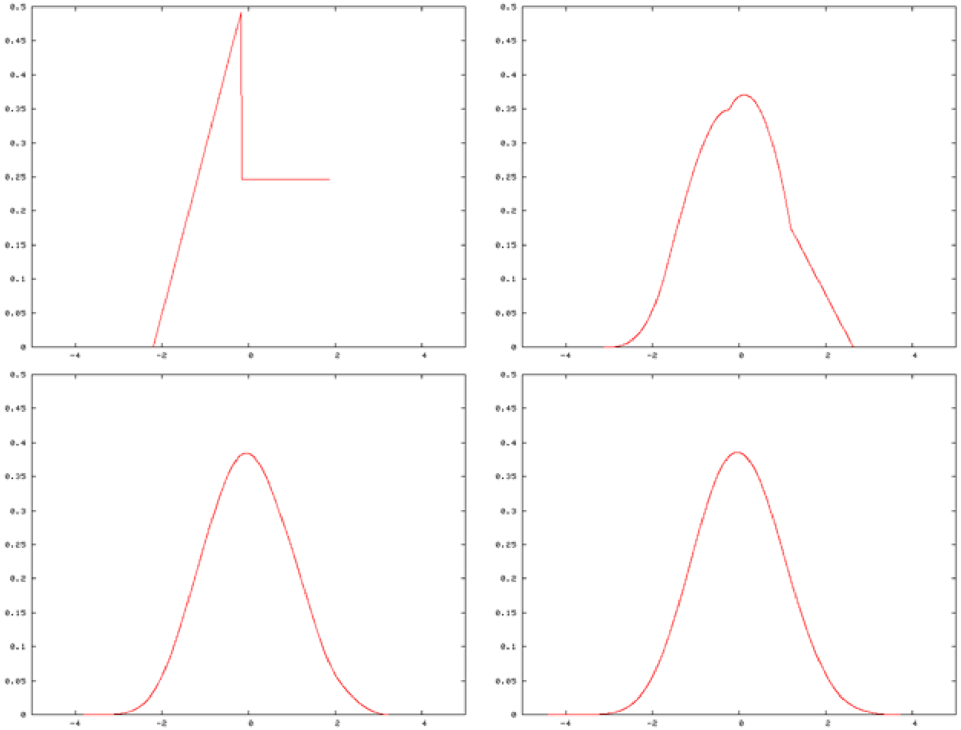


Figure 1.13: Illustration of central limit theorem.

graphical programming has somewhat lowered the barrier to entry for experienced programmers to graphical programming. With hundreds of cores (up to a few thousand with recent NVIDIA graphics cards where the researcher may also have multiple cards installed), GPU programming can achieve performance akin to supercomputers and the benefits of CUDA have been extended with third party wrappers for MATLAB, Fortran, Java, Python and others.[18] Solving a model through dynamic programming (value function iteration) across a grid, with only a few grid points, the overhead in memory

[18] NVIDIA's graphics cards are capable of over 8 TFLOPS each [Geforce GTX Titan Z] with 5,760 CUDA cores, of which you can place multiple cards in your computer under your desk! Watch `http://www.youtube.com/watch?v=2JjxgJcXVE0`. Google Brain uses 600kWatts of energy and costs $1,000,000 whereas with three of these Titan Z GPU cards, there are savings of 300 times less energy and 400 times lower cost, plus the GPU cards fit into a regular desktop computer.

transmission for CUDA does not make it optimal relative to standard CPU programming. However, as we increase the grid size, running time on CPU tends to increase linearly while the running time with CUDA programming on GPUs increases very slowly. In the example given in the paper, for 16 grid points, it takes 0.03 seconds for the total CPU solution but 1.42 seconds with the GPU solution, whereas for 65,356 grid points, it takes 24,588.5 seconds for the total CPU solution but only 48.32 seconds with the GPU solution. Given that the number of processors on graphics cards has increased multiples since the paper was written and given that we can use multiple cards simultaneously, these estimates form a conservative lower bound on the benefits of GPU programming.

Parallel programming

Parallel programming can reduce the running time of code by using multiple cores or multiple nodes (machines on a cluster) simultaneously for part of the code that can be run in parallel. Gene Amdahl (designer of IBM 360 mainframe) said that any task can be split into parts that can be parallelised and parts that cannot; serial parts: setup of problem by reading in data, generating statistics after each iteration, etc.; parallel parts: numerical solver, Monte Carlo, etc. Suppose we have a task of which 95% can be executed in parallel. Even if we use an infinite number of processes on the parallel part, we still need 5% of the original time to execute the serial part. Amdahl's law states that the maximum speed up is given by the formula

$$\frac{1}{S + \frac{1-S}{N}}$$

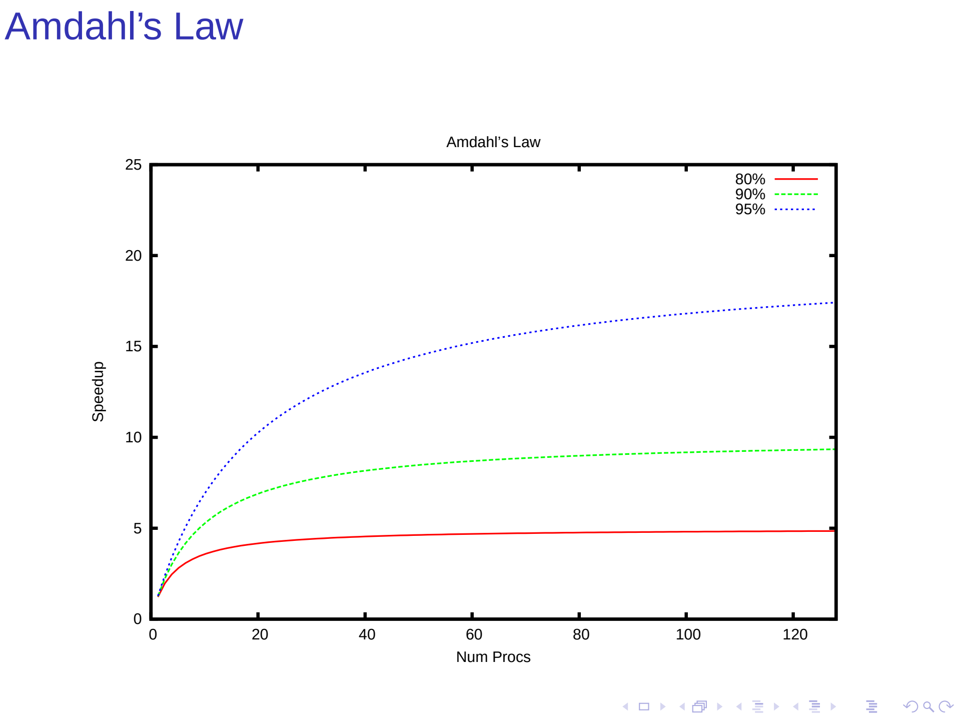
where S is the proportion of the code to be executed in serial and N is the number of processes in the parallel part; see figure 1.14. It is untrue that this suggests that there is no point in writing code for more than 8-16 processes, since as you run on larger problem sizes, often the serial part scales linearly but the parallel part scales with n^2 or n^3. By tackling larger problems, a 95% parallel problem can become a 99% parallel problem and eventually a 99.9%

parallel problem. Plugging in figures, a 99.9% problem on a 1024 way system gives a 506 speedup. Of course, Amdahl's law assumes that the parallel part is perfectly parallelisable – it does not take into consideration the time spent passing data between processes (overhead). Parallelism can occur in two different places in a parallel computer system: (i) processor – low level instruction parallelism (your compiler will try to take advantage of this); (ii) interconnect – higher level parallelism (you the programmer need to manage this). For shared memory systems (e.g. personal computer), OpenMP is an application programming interface that can be used say in C, C++ or Fortran. For distributed memory systems (e.g. cluster), MPI (Message Passing Interface) is a computer library that is built on C or Fortran and used to program parallel computers / clusters, etc.

Comparing programming languages in Economics[19]

As for the choice between computer programming languages, there are a number of issues relevant for researchers to consider: (i) computational burden in terms of writing the code (e.g. number of lines of code, higher barrier to entry on more low-level languages e.g. assembler/C/C++/Fortran where perhaps certain tasks may be easier to complete in higher-level, menu-driven programs or programs with in-built functions or packages for economics/econometrics e.g. Stata/MATLAB/R/E-Views, increasing returns from specialisation in a given language, etc.); (ii) code legacy (availability of plenty of code in Fortran/MATLAB on the web vs less common languages); (iii) choice of compiler for compiled languages can make a difference in terms of speed (e.g. Intel Fortran Compilers vs GNU compilers); (iv) replication of code (open source available to all freely vs standards like MATLAB/Intel compilers and expensive packages like IMSL for Fortran); (v) running time concerns may be an issue for larger problems. On this last concern, the paper by Aldrich *et al* (2014) provide a comparison of programmming languages for economists, breaking the conventional 'folk' wisdom that Fortran is the fastest language

[19] You should consult `http://economics.sas.upenn.edu/~jesusfv/comparison_languages.pdf`.



Amdahl's Law

Figure 1.14: Amdahl's law.

(C++ seems to be slightly faster for some problems), showing how choice of compilers matter too. They solve the stochastic neoclassical growth model, which is the workhorse model in economics, using value function iteration; while more advanced solution methods such as projection and perturbation are useful for DSGEs, value function iteration is more applicable to other general problems in economics from game theory to econometrics, etc. They compare compiled (e.g. Fortran/C++/Java) and scripted programming languages (e.g. MATLAB/Julia/Python/Mathematica/R) along with hybrids of both; another paper (Amador *et al* (2014)) investigates functional programming languages (e.g. Ocaml/Haskell). Rankings of the fastest language and compiler in terms of run-time vary depending on whether the system is a Windows or a Mac. C++ tends to slighly outpeform Fortran and whereas open source compilers are faster on Mac (similar to a Linux system), the Intel compiler is faster for Windows systems. Java and Julia come in next, about 2 to 3 times as slow for the problem in the paper, whereas MATLAB is about 10 times as slow; hybrid MATLAB using Mex files with C/C++ is significantly faster to standard MATLAB, only about 1.5 times as slow as stand-alone C++. Python is about 50 to 150 times as slow as C++, while the relative figures for R and Mathematica are 250 to 500 and 625 to 800, respectively; significant speed-up is achieved with hybrid R, Python and Mathematica bringing the relative figures to between about 1.5 to 5 times as slow as C++.

Bibliography

DeJong, D. N. and Dave, C. (2011). *Structural Macroeconometrics: (Second Edition)*. Princeton University Press.