Thomas J. Sargent & John Stachurski

37. Irrelevance of Capital Structures with Complete Markets

Lecture (PDF)
Book (PDF) (_pdf/quantecon-python-adv

```
!pip install --upgrade quantecon
!pip install interpolation
!conda install -y -c plotly plotly plotly-orca
```

```
Requirement already satisfied: quantecon in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (0.5.2)
```

```
Requirement already satisfied: numba>=0.38 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from quantecon)
(0.54.1)
Requirement already satisfied: numpy in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from quantecon)
(1.20.3)
Requirement already satisfied: sympy in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from quantecon) (1.9)
Requirement already satisfied: requests in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from quantecon)
(2.26.0)
Requirement already satisfied: scipy>=1.0.0 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from quantecon)
(1.7.1)
```

```
Requirement already satisfied: setuptools in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from numba>=0.38-
>quantecon) (58.0.4)
Requirement already satisfied: llvmlite<0.38,>=0.37.0rc1 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages
(from numba>=0.38->quantecon) (0.37.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
requests->quantecon) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
requests->quantecon) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages
(from requests->quantecon) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from requests-
>quantecon) (3.2)
Requirement already satisfied: mpmath>=0.19 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from sympy-
>quantecon) (1.2.1)
```

```
Requirement already satisfied: interpolation in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (2.2.1)
Requirement already satisfied: tempita>=0.5.2 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
interpolation) (0.5.2)
Requirement already satisfied: scipy>=1.4.1 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
interpolation) (1.7.1)
Requirement already satisfied: numpy>=1.18.1 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
interpolation) (1.20.3)
Requirement already satisfied: numba>=0.47 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from
interpolation) (0.54.1)
Requirement already satisfied: setuptools in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages (from numba>=0.47-
>interpolation) (58.0.4)
Requirement already satisfied: llvmlite<0.38,>=0.37.0rc1 in /usr/share/miniconda3/envs/quantecon/lib/python3.8/site-packages
(from numba>=0.47->interpolation) (0.37.0)
```

```
Collecting package metadata (current_repodata.json): -
```

```
\
```

```
|
```

```
/
```

```
-
```

```
\
```

```
|
```

```
/
```

```
-
```

```
\
```

```
|
```

```
/
```

```
-
```

```
\
|
/
-
\
|
/
-
\
|
/
-
\
|
/
-
\
```

```
  done
Solving environment: /
```

```
-
\
|
/
-
\
|
/
-
\
|
/
-
\
```

```
  \
```

```
  |
```

```
  /
```

```
  -
```

```
  \
```

```
  |
```

```
  /
```

```
  done
```

```
## Package Plan ##

  environment location: /usr/share/miniconda3/envs/quantecon

  added / updated specs:
    - plotly
    - plotly-orca


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    plotly-5.6.0               |             py_0         6.9 MB  plotly
    plotly-orca-1.3.1          |                1        56.6 MB  plotly
    tenacity-8.0.1             |   py38h06a4308_0          38 KB
    ------------------------------------------------------------
                                           Total:        63.6 MB

The following NEW packages will be INSTALLED:

  plotly             plotly/noarch::plotly-5.6.0-py_0
  plotly-orca        plotly/linux-64::plotly-orca-1.3.1-1
  tenacity           pkgs/main/linux-64::tenacity-8.0.1-py38h06a4308_0


Downloading and Extracting Packages

tenacity-8.0.1       | 38 KB     |                                       |   0%
```

```
tenacity-8.0.1       | 38 KB     | ##################################### | 100%
```

```
plotly-orca-1.3.1    | 56.6 MB   |                                       |   0%
```

```
plotly-orca-1.3.1    | 56.6 MB   | ##################################### | 100%
plotly-orca-1.3.1    | 56.6 MB   | ##################################### | 100%

plotly-5.6.0         | 6.9 MB    |                                       |   0%
```

```
plotly-5.6.0         | 6.9 MB    | ##################################### | 100%
plotly-5.6.0         | 6.9 MB    | ##################################### | 100%
Preparing transaction: \
```

```
  |
```

```
  /
```

```
  -
```

```
  \
```

```
  |
```

```
  /
```

```
  done
Verifying transaction: \
```

```
  |
```

```
  /
```

```
  -
```

```
\
|
/
-
\
|
/
-
  done
Executing transaction: |
/
-
\
|
/
-
\
|
/
-
\
|
/
-
\
|
/
-
\
|
/
-
\
```

```
\
|
/
-
\
|
/
-
\
|
/
-
\
|
/
done
```

37.1. Introduction

This is a prolegomenon to another lecture Equilibrium Capital Structures with Incomplete Markets (BCG_incomplete_mkts.html) about a model with incomplete markets authored by Bisin, Clementi, and Gottardi [BCG18 (zreferences.html#id29)].

We adopt specifications of preferences and technologies very close to Bisin, Clemente, and Gottardi's but unlike them assume that there are complete markets in one-period Arrow securities.

This simplification of BCG's setup helps us by

- creating a benchmark economy to compare with outcomes in BCG's incomplete markets economy
- creating a good guess for initial values of some equilibrium objects to be computed in BCG's incomplete markets economy via an iterative algorithm
- illustrating classic complete markets outcomes that include
    - indeterminacy of consumers' portfolio choices
    - indeterminacy of firms' financial structures that underlies a Modigliani-Miller theorem [MM58 (zreferences.html#id30)]
- introducing `Big K, little k` issues in a simple context that will recur in the BCG incomplete markets environment

A Big K, little k analysis also played roles in this quantecon lecture (https://python.quantecon.org/cass_koopmans.html) as well as here (https://python.quantecon.org/rational_expectations.html) and here (dyn_stack.html).

37.1.1. Setup

The economy lasts for two periods, $t = 0, 1$.

There are two types of consumers named $i = 1, 2$.

A scalar random variable ε with probability density $g(\varepsilon)$ affects both

- the return in period 1 from investing $k \geq 0$ in physical capital in period 0.
- exogenous period 1 endowments of the consumption good for agents of types $i = 1$ and $i = 2$.

Type $i = 1$ and $i = 2$ agents' period 1 endowments are correlated with the return on physical capital in different ways.

We discuss two arrangements:

- a command economy in which a benevolent planner chooses k and allocates goods to the two types of consumers in each period and each random second period state
- a competitive equilibrium with markets in claims on physical capital and a complete set (possibly a continuum) of one-period Arrow securities that pay period 1 consumption goods contingent on the realization of random variable ε.

37.1.2. Endowments

There is a single consumption good in period $\mathbf{0}$ and at each random state $\boldsymbol{\varepsilon}$ in period $\mathbf{1}$.

Economy-wide endowments in periods $\mathbf{0}$ and $\mathbf{1}$ are

$$\begin{matrix} w_0 \\ w_1(\varepsilon) \text{ in state } \varepsilon \end{matrix}$$

Soon we'll explain how aggregate endowments are divided between type $i = 1$ and type $i = 2$ consumers.

We don't need to do that in order to describe a social planning problem.

37.1.3. Technology:

Where $\alpha \in (0,1)$ and $A > 0$

$$\begin{aligned} c_0^1 + c_0^2 + k &= w_0^1 + w_0^2 \\ c_1^1(\varepsilon) + c_1^2(\varepsilon) &= w_1^1(\varepsilon) + w_1^2(\varepsilon) + e^{\varepsilon} A k^{\alpha}, \quad k \geq 0 \end{aligned}$$

37.1.4. Preferences:

A consumer of type i orders period $\mathbf{0}$ consumption c_0^i and state ε, period $\mathbf{1}$ consumption $c_1^i(\varepsilon)$ by

$$u^i = u(c_0^i) + \beta \int u(c_1^i(\varepsilon)) g(\varepsilon) d\varepsilon, \quad i = 1, 2$$

$\beta \in (0,1)$ and the one-period utility function is

$$u(c) = \begin{cases} \frac{c^{1-\gamma}}{1-\gamma} & \text{if } \gamma \neq 1 \\ \log c & \text{if } \gamma = 1 \end{cases}$$

37.1.5. Parameterizations

Following BCG, we shall employ the following parameterizations:

$$\begin{aligned} \varepsilon &\sim \mathcal{N}(\mu, \sigma^2) \\ u(c) &= \frac{c^{1-\gamma}}{1-\gamma} \\ w_1^i(\varepsilon) &= e^{-\chi_i \mu - .5 \chi_i^2 \sigma^2 + \chi_i \varepsilon}, \quad \chi_i \in [0,1] \end{aligned}$$

Sometimes instead of asuming $\varepsilon \sim g(\varepsilon) = \mathcal{N}(0, \sigma^2)$, we'll assume that $g(\cdot)$ is a probability mass function that serves as a discrete approximation to a standardized normal density.

37.1.6. Pareto criterion and planning problem

The planner's objective function is

$$\text{obj} = \phi_1 u^1 + \phi_2 u^2, \quad \phi_i \geq 0, \quad \phi_1 + \phi_2 = 1$$

where $\phi_i \geq 0$ is a Pareto weight that the planner attaches to a consumer of type i.

We form the following Lagrangian for the planner's problem:

$$\begin{aligned} L = &\sum_{i=1}^{2} \phi_i \left[u(c_0^i) + \beta \int u(c_1^i(\varepsilon)) g(\varepsilon) d\varepsilon \right] \\ &+ \lambda_0 \left[w_0^1 + w_0^2 - k - c_0^1 - c_0^2 \right] \\ &+ \beta \int \lambda_1(\varepsilon) \left[w_1^1(\varepsilon) + w_1^2(\varepsilon) + e^{\varepsilon} A k^{\alpha} - c_1^1(\varepsilon) - c_1^2(\varepsilon) \right] g(\varepsilon) d\varepsilon \end{aligned}$$

First-order necessary optimality conditions for the planning problem are:

$$\begin{aligned} c_0^1: &\quad \phi_1 u'(c_0^1) - \lambda_0 = 0 \\ c_0^2: &\quad \phi_2 u'(c_0^2) - \lambda_0 = 0 \\ c_1^1(\varepsilon): &\quad \phi_1 \beta u'(c_1^1(\varepsilon)) g(\varepsilon) - \beta \lambda_1(\varepsilon) g(\varepsilon) = 0 \\ c_1^2(\varepsilon): &\quad \phi_2 \beta u'(c_1^2(\varepsilon)) g(\varepsilon) - \beta \lambda_1(\varepsilon) g(\varepsilon) = 0 \\ k: &\quad -\lambda_0 + \beta \alpha A k^{\alpha - 1} \int \lambda_1(\varepsilon) e^{\varepsilon} g(\varepsilon) d\varepsilon = 0 \end{aligned}$$

The first four equations imply that

$$\begin{aligned} \frac{u'(c_1^1(\varepsilon))}{u'(c_0^1))} = \frac{u'(c_1^2(\varepsilon))}{u'(c_0^2))} &= \frac{\lambda_1(\varepsilon)}{\lambda_0} \\ \frac{u'(c_0^1)}{u'(c_0^2)} = \frac{u'(c_1^1(\varepsilon))}{u'(c_1^2(\varepsilon))} &= \frac{\phi_2}{\phi_1} \end{aligned}$$

These together with the fifth first-order condition for the planner imply the following equation that determines an optimal choice of capital

$$1 = \beta \alpha A k^{\alpha - 1} \int \frac{u'(c_1^i(\varepsilon))}{u'(c_0^i)} e^{\varepsilon} g(\varepsilon) d\varepsilon$$

for $i = 1, 2$.

37.1.7. Helpful observations and bookkeeping

Evidently,

$$u'(c) = c^{-\gamma}$$

and

$$\frac{u'(c^1)}{u'(c^2)} = \left(\frac{c^1}{c^2}\right)^{-\gamma} = \frac{\phi_2}{\phi_1}$$

where it is to be understood that this equation holds for $c^1 = c_0^1$ and $c^2 = c_0^2$ and also for $c^1 = c^1(\varepsilon)$ and $c^2 = c^2(\varepsilon)$ for all ε.

With the same understanding, it follows that

$$\left(\frac{c^1}{c^2}\right) = \left(\frac{\phi_2}{\phi_1}\right)^{-\gamma^{-1}}$$

Let $c = c^1 + c^2$.

It follows from the preceding equation that

$$\begin{aligned} c^1 &= \eta c \\ c^2 &= (1-\eta)c \end{aligned}$$

where $\eta \in [0,1]$ is a function of ϕ_1 and γ.

Consequently, we can write the planner's first-order condition for k as

$$1 = \beta\alpha A k^{\alpha-1} \int \left(\frac{w_1(\varepsilon) + Ak^\alpha e^\varepsilon}{w_0 - k}\right)^{-\gamma} e^\varepsilon g(\varepsilon) d\varepsilon$$

which is one equation to be solved for $k \geq 0$.

Anticipating a `Big K, little k` idea widely used in macroeconomics, to be discussed in detail below, let K be the value of k that solves the preceding equation so that

$$1 = \beta\alpha A K^{\alpha-1} \int \left(\frac{w_1(\varepsilon) + AK^\alpha e^\varepsilon}{w_0 - K}\right)^{-\gamma} g(\varepsilon) e^\varepsilon d\varepsilon$$

The associated optimal consumption allocation is

$$\begin{aligned} C_0 &= w_0 - K \\ C_1(\varepsilon) &= w_1(\varepsilon) + AK^\alpha e^\varepsilon \\ c_0^1 &= \eta C_0 \\ c_0^2 &= (1-\eta) C_0 \\ c_1^1(\varepsilon) &= \eta C_1(\varepsilon) \\ c_1^2(\varepsilon) &= (1-\eta) C_1(\varepsilon) \end{aligned}$$

where $\eta \in [0,1]$ is the consumption share parameter mentioned above that is a function of the Pareto weight ϕ_1 and the utility curvature parameter γ.

37.1.7.1. Remarks

The relative Pareto weight parameter η does not appear in equation (37.1) that determines K.

Neither does it influence C_0 or $C_1(\varepsilon)$, which depend solely on K.

The role of η is to determine how to allocate total consumption between the two types of consumers.

Thus, the planner's choice of K does not interact with how it wants to allocate consumption.

37.2. Competitive equilibrium

We now describe a competitive equilibrium for an economy that has specifications of consumer preferences, technology, and aggregate endowments that are identical to those in the preceding planning problem.

While prices do not appear in the planning problem – only quantities do – prices play an important role in a competitive equilibrium.

To understand how the planning economy is related to a competitive equilibrium, we now turn to the `Big K, little k` distinction.

37.2.1. Measures of agents and firms

We follow BCG in assuming that there are unit measures of

- consumers of type $i = 1$
- consumers of type $i = 2$
- firms with access to the production technology that converts k units of time 0 good into $Ak^\alpha e^\varepsilon$ units of the time 1 good in random state ε

Thus, let $\omega \in [0,1]$ index a particular consumer of type i.

Then define Big C^i as

$$C^i = \int_0^1 c^i(\omega) d\omega$$

In the same spirit, let $\zeta \in [0,1]$ index a particular firm. Then define Big K as

$$K = \int_0^1 k(\zeta) d\zeta$$

The assumption that there are continua of our three types of agents plays an important role making each individual agent into a powerless **price taker**:

- an individual consumer chooses its own (infinesimal) part $c^i(\omega)$ of C^i taking prices as given
- an individual firm chooses its own (infinitesmimal) part $k(\zeta)$ of K taking prices as
- equilibrium prices depend on the `Big K, Big C` objects K and C

Nevertheless, in equilibrium, $K = k, C^i = c^i$

The assumption about measures of agents is thus a powerful device for making a host of competitive agents take as given equilibrium prices that are determined by the independent decisions of hosts of agents who behave just like they do.

37.2.1.1. Ownership

Consumers of type i own the following exogenous quantities of the consumption good in periods 0 and 1:

$$\begin{aligned} w_0^i, &\quad i = 1, 2 \\ w_1^i(\varepsilon) &\quad i = 1, 2 \end{aligned}$$

where

$$\begin{aligned} \sum_i w_0^i &= w_0 \\ \sum_i w_1^i(\varepsilon) &= w_1(\varepsilon) \end{aligned}$$

Consumers also own shares in a firm that operates the technology for converting nonnegative amounts of the time 0 consumption good one-for-one into a capital good k that produces $Ak^\alpha e^\varepsilon$ units of the time 1 consumption good in time 1 state ε.

Consumers of types $i = 1, 2$ are endowed with θ_0^i shares of a firm and

$$\theta_0^1 + \theta_0^2 = 1$$

37.2.1.2. Asset markets

At time 0, consumers trade the following assets with other consumers and with firms:

- equities (also known as stocks) issued by firms
- one-period Arrow securities that pay one unit of consumption at time 1 when the shock ε assumes a particular value

Later, we'll allow the firm to issue bonds too, but not now.

37.2.2. Objects appearing in a competitive equilibrium

Let

- $a^i(\varepsilon)$ be consumer i 's purchases of claims on time 1 consumption in state ε
- $q(\varepsilon)$ be a pricing kernel for one-period Arrow securities
- $\theta_0^i \geq 0$ be consumer i's intial share of the firm, $\sum_i \theta_0^i = 1$
- θ^i be the fraction of a firm's shares purchased by consumer i at time $t = 0$
- V be the value of the representative firm
- $\tilde{V}$ be the value of equity issued by the representative firm
- K, C_0 be two scalars and $C_1(\varepsilon)$ a function that we use to construct a guess about an equilibrium pricing kernel for Arrow securities

We proceed to describe constrained optimum problems faced by consumers and a representative firm in a competitive equilibrium.

37.2.3. A representative firm's problem

A representative firm takes Arrow security prices $q(\varepsilon)$ as given.

The firm purchases capital $k \geq 0$ from consumers at time 0 and finances itself by issuing equity at time 0.

The firm produces time 1 goods $Ak^\alpha e^\varepsilon$ in state ε and pays all of these `earnings` to owners of its equity.

The value of a firm's equity at time 0 can be computed by multiplying its state-contingent earnings by their Arrow securities prices and then adding over all contingencies:

$$\tilde{V} = \int Ak^\alpha e^\varepsilon q(\varepsilon) d\varepsilon$$

Owners of a firm want it to choose k to maximize

$$V = -k + \int Ak^\alpha e^\varepsilon q(\varepsilon) d\varepsilon$$

The firm's first-order necessary condition for an optimal k is

$$-1 + \alpha A k^{\alpha - 1} \int e^\varepsilon q(\varepsilon) d\varepsilon = 0$$

The time 0 value of a representative firm is

$$V = -k + \tilde{V}$$

The right side equals the value of equity minus the cost of the time 0 goods that it purchases and uses as capital.

37.2.4. A consumer's problem

We now pose a consumer's problem in a competitive equilibrium.

As a price taker, each consumer faces a given Arrow securities pricing kernel $q(\varepsilon)$, a given value of a firm V that has chosen capital stock k, a price of equity $\tilde{V}$, and prospective next period random dividends $Ak^\alpha e^\varepsilon$.

If we evaluate consumer i's time 1 budget constraint at zero consumption $c_1^i(\varepsilon) = 0$ and solve for $-a^i(\varepsilon)$ we obtain

$$-\bar{a}^i(\varepsilon;\theta^i) = w_1^i(\varepsilon) + \theta^i Ak^\alpha e^\varepsilon$$

The quantity $-\bar{a}^i(\varepsilon;\theta^i)$ is the maximum amount that it is feasible for consumer i to repay to his Arrow security creditors at time 1 in state ε.

Notice that $-\bar{a}^i(\varepsilon;\theta^i)$ defined in (37.2) depends on

- his endowment $w_1^i(\varepsilon)$ at time 1 in state ε
- his share θ^i of a representative firm's dividends

These constitute two sources of **collateral** that back the consumer's issues of Arrow securities that pay off in state ε

Consumer i chooses a scalar c_0^i and a function $c_1^i(\varepsilon)$ to maximize

$$u(c_0^i) + \beta \int u(c_1^i(\varepsilon))g(\varepsilon)d\varepsilon$$

subject to time 0 and time 1 budget constraints

$$\begin{aligned} c_0^i &\leq w_0^i + \theta_0^i V - \int q(\varepsilon)a^i(\varepsilon)d\varepsilon - \theta^i\tilde{V} \\ c_1^i(\varepsilon) &\leq w_1^i(\varepsilon) + \theta^i Ak^\alpha e^\varepsilon + a^i(\varepsilon) \end{aligned}$$

Attach Lagrange multiplier λ_0^i to the budget constraint at time 0 and scaled Lagrange multiplier $\beta\lambda_1^i(\varepsilon)g(\varepsilon)$ to the budget constraint at time 1 and state ε, then form the Lagrangian

$$\begin{aligned} L^i &= u(c_0^i) + \beta \int u(c_1^i(\varepsilon))g(\varepsilon)d\varepsilon \\ &\quad + \lambda_0^i[w_0^i + \theta_0^i - \int q(\varepsilon)a^i(\varepsilon)d\varepsilon - \theta^i\tilde{V} - c_0^i] \\ &\quad + \beta \int \lambda_1^i(\varepsilon)[w_1^i(\varepsilon) + \theta^i Ak^\alpha e^\varepsilon + a^i(\varepsilon) - c_1^i(\varepsilon)]g(\varepsilon)d\varepsilon \end{aligned}$$

Off corners, first-order necessary conditions for an optimum with respect to $c_0^i, c_1^i(\varepsilon),$ and $a^i(\varepsilon)$ are

$$\begin{aligned} c_0^i: &\quad u'(c_0^i) - \lambda_0^i = 0 \\ c_1^i(\varepsilon): &\quad \beta u'(c_1^i(\varepsilon))g(\varepsilon) - \beta\lambda_1^i(\varepsilon)g(\varepsilon) = 0 \\ a^i(\varepsilon): &\quad -\lambda_0^i q(\varepsilon) + \beta\lambda_1^i(\varepsilon) = 0 \end{aligned}$$

These equations imply that consumer i adjusts its consumption plan to satisfy

$$q(\varepsilon) = \beta\left(\frac{u'(c_1^i(\varepsilon))}{u'(c_0^i)}\right)g(\varepsilon)$$

To deduce a restriction on equilibrium prices, we solve the period 1 budget constraint to express $a^i(\varepsilon)$ as

$$a^i(\varepsilon) = c_1^i(\varepsilon) - w_1^i(\varepsilon) - \theta^i Ak^\alpha e^\varepsilon$$

then substitute the expression on the right side into the time 0 budget constraint and rearrange to get the single intertemporal budget constraint

$$w_0^i + \theta_0^i V + \int w_1^i(\varepsilon)q(\varepsilon)d\varepsilon + \theta^i\left[Ak^\alpha \int e^\varepsilon q(\varepsilon)d\varepsilon - \tilde{V}\right] \geq c_0^i + \int c_1^i(\varepsilon)q(\varepsilon)d\varepsilon$$

The right side of inequality (37.4) is the present value of consumer i's consumption while the left side is the present value of consumer i's endowment when consumer i buys θ^i shares of equity.

From inequality (37.4), we deduce two findings.

1. No arbitrage profits condition:

Unless

$$\tilde{V} = Ak^\alpha \int e^\varepsilon q(\varepsilon)d\varepsilon$$

an **arbitrage** opportunity would be open.

If

$$\tilde{V} > Ak^\alpha \int e^\varepsilon q(\varepsilon)d\varepsilon$$

the consumer could afford an arbitrarily high present value of consumption by setting θ^i to an arbitrarily large **negative** number.

If

$$\tilde{V} < Ak^\alpha \int e^\varepsilon q(\varepsilon)d\varepsilon$$

the consumer could afford an arbitrarily high present value of consumption by setting θ^i to be arbitrarily large **positive** number.

Since resources are finite, there can exist no such arbitrage opportunity in a competitive equilibrium.

Therefore, it must be true that the following no arbitrage condition prevails:

$$\tilde{V} = \int A k^{\alpha} e^{\varepsilon} q(\varepsilon; K) d\varepsilon$$

Equation (37.6) asserts that the value of equity equals the value of the state-contingent dividends $Ak^{\alpha}e^{\varepsilon}$ evaluated at the Arrow security prices $q(\varepsilon; K)$ that we have expressed as a function of K.

We'll say more about this equation later.

2. Indeterminacy of portfolio

When the no-arbitrage pricing equation (37.6) prevails, a consumer of type i's choice θ^i of equity is indeterminate.

Consumer of type i can offset any choice of θ^i by setting an appropriate schedule $a^i(\varepsilon)$ for purchasing state-contingent securities.

37.2.5. Computing competitive equilibrium prices and quantities

Having computed an allocation that solves the planning problem, we can readily compute a competitive equilibrium via the following steps that, as we'll see, relies heavily on the `Big K, little k, Big C, little c` logic mentioned earlier:

- a competitive equilbrium allocation equals the allocation chosen by the planner
- competitive equilibrium prices and the value of a firm's equity are encoded in shadow prices from the planning problem that depend on Big K and Big C.

To substantiate that this procedure is valid, we proceed as follows.

With K in hand, we make the following guess for competitive equilibrium Arrow securities prices

$$q(\varepsilon; K) = \beta \left(\frac{u'\left(w_1(\varepsilon) + AK^{\alpha}e^{\varepsilon}\right)}{u'(w_0 - K)} \right)^{-\gamma}$$

To confirm the guess, we begin by considering its consequences for the firm's choice of k.

With Arrow securities prices (37.7), the firm's first-order necessary condition for choosing k becomes

$$-1 + \alpha A k^{\alpha - 1} \int e^{\varepsilon} q(\varepsilon; K) d\varepsilon = 0$$

which can be verified to be satisfied if the firm sets

$$k = K$$

because by setting $k = K$ equation (37.8) becomes equivalent with the planner's first-order condition (37.1) for setting K.

To pose a consumer's problem in a competitive equilibrium, we require not only the above guess for the Arrow securities pricing kernel $q(\varepsilon)$ but the value of equity $\tilde{V}$:

$$\tilde{V} = \int A K^{\alpha} e^{\varepsilon} q(\varepsilon; K) d\varepsilon$$

Let $\tilde{V}$ be the value of equity implied by Arrow securities price function (37.7) and formula (37.9).

At the Arrow securities prices $q(\varepsilon)$ given by (37.7) and equity value $\tilde{V}$ given by (37.9), consumer $i = 1, 2$ choose consumption allocations and portolios that satisfy the first-order necessary conditions

$$\beta \left(\frac{u'(c_1^i(\varepsilon))}{u'(c_0^i)} \right) g(\varepsilon) = q(\varepsilon; K)$$

It can be verified directly that the following choices satisfy these equations

$$\begin{aligned} c_0^1 + c_0^2 &= C_0 = w_0 - K \\ c_0^1(\varepsilon) + c_0^2(\varepsilon) &= C_1(\varepsilon) = w_1(\varepsilon) + Ak^{\alpha}e^{\varepsilon} \\ \frac{c_1^2(\varepsilon)}{c_1^1(\varepsilon)} &= \frac{c_0^2}{c_0^1} = \frac{1-\eta}{\eta} \end{aligned}$$

for an $\eta \in (0, 1)$ that depends on consumers' endowments $[w_0^1, w_0^2, w_1^1(\varepsilon), w_1^2(\varepsilon), \theta_0^1, \theta_0^2]$.

Remark: Multiple arrangements of endowments $[w_0^1, w_0^2, w_1^1(\varepsilon), w_1^2(\varepsilon), \theta_0^1, \theta_0^2]$ associated with the same distribution of wealth η. Can you explain why? **Hint:** Think about the portfolio indeterminacy finding above.

37.2.6. Modigliani-Miller theorem

We now allow a firm to issue both bonds and equity.

Payouts from equity and bonds, respectively, are

$$\begin{aligned} d^e(k, b; \varepsilon) &= \max\left\{e^{\varepsilon}Ak^{\alpha} - b, 0\right\} \\ d^b(k, b; \varepsilon) &= \min\left\{\frac{e^{\varepsilon}Ak^{\alpha}}{b}, 1\right\} \end{aligned}$$

Thus, one unit of the bond pays one unit of consumption at time 1 in state ε if $Ak^{\alpha}e^{\varepsilon} - b \geq 0$, which is true when $\varepsilon \geq \varepsilon^* = \log \frac{b}{Ak^{\alpha}}$, and pays $\frac{Ak^{\alpha}e^{\varepsilon}}{b}$ units of time 1 consumption in state ε when $\varepsilon < \varepsilon^*$.

The value of the firm is now the sum of equity plus the value of bonds, which we denote

$$\tilde{V} + bp(k, b)$$

where $p(k, b)$ is the price of one unit of the bond when a firm with k units of physical capital issues b bonds.

We continue to assume that there are complete markets in Arrow securities with pricing kernel $q(\varepsilon)$.

A version of the no-arbitrage-in-equilibrium argument that we presented earlier implies that the value of equity and the price of bonds are

$$\tilde{V} = Ak^\alpha \int_{\varepsilon^*}^{\infty} e^\varepsilon q(\varepsilon) d\varepsilon - b \int_{\varepsilon^*}^{\infty} q(\varepsilon) d\varepsilon$$
$$p(k, b) = \frac{Ak^\alpha}{b} \int_{-\infty}^{\varepsilon^*} e^\varepsilon q(\varepsilon) d\varepsilon + \int_{\varepsilon^*}^{\infty} q(\varepsilon) d\varepsilon$$

Consequently, the value of the firm is

$$\tilde{V} + p(k, b)b = Ak^\alpha \int_{-\infty}^{\infty} e^\varepsilon q(\varepsilon) d\varepsilon,$$

which is the same expression that we obtained above when we assumed that the firm issued only equity.

We thus obtain a version of the celebrated Modigliani-Miller theorem [MM58 (zreferences.html#id30)] about firms' finance:

Modigliani-Miller theorem:

- The value of a firm is independent the mix of equity and bonds that it uses to finance its physical capital.
- The firms's decision about how much physical capital to purchase does not depend on whether it finances those purchases by issuing bonds or equity
- The firm's choice of whether to finance itself by issuing equity or bonds is indeterminant

Please note the role of the assumption of complete markets in Arrow securities in substantiating these claims.

In Equilibrium Capital Structures with Incomplete Markets (BCG_incomplete_mkts.html), we will assume that markets are (very) incomplete – we'll shut down markets in almost all Arrow securities.

That will pull the rug from underneath the Modigliani-Miller theorem.

37.3. Code

We create a class object `BCG_complete_markets` to compute equilibrium allocations of the complete market BCG model given a list of parameter values.

It consists of 4 functions that do the following things:

- `opt_k` computes the planner's optimal capital K
  - First, create a grid for capital.
  - Then for each value of capital stock in the grid, compute the left side of the planner's first-order necessary condition for k, that is,

$$\beta \alpha A K^{\alpha-1} \int \left(\frac{w_1(\varepsilon) + AK^\alpha e^\varepsilon}{w_0 - K} \right)^{-\gamma} e^\varepsilon g(\varepsilon) d\varepsilon - 1 = 0$$

  - Find k that solves this equation.
- `q` computes Arrow security prices as a function of the productivity shock ε and capital K:

$$q(\varepsilon; K) = \beta \left(\frac{u'(w_1(\varepsilon) + AK^\alpha e^\varepsilon)}{u'(w_0 - K)} \right)$$

- `v` solves for the firm value given capital k:

$$V = -k + \int Ak^\alpha e^\varepsilon q(\varepsilon; K) d\varepsilon$$

- `opt_c` computes optimal consumptions c_0^i, and $c^i(\varepsilon)$:
  - The function first computes weight η using the budget constraint for agent 1:

$$w_0^1 + \theta_0^1 V + \int w_1^1(\varepsilon) q(\varepsilon) d\varepsilon = c_0^1 + \int c_1^1(\varepsilon) q(\varepsilon) d\varepsilon = \eta \left(C_0 + \int C_1(\varepsilon) q(\varepsilon) d\varepsilon \right)$$

    where

$$C_0 = w_0 - K$$
$$C_1(\varepsilon) = w_1(\varepsilon) + AK^\alpha e^\varepsilon$$

  - It computes consumption for each agent as

$$c_0^1 = \eta C_0$$
$$c_0^2 = (1 - \eta) C_0$$
$$c_1^1(\varepsilon) = \eta C_1(\varepsilon)$$
$$c_1^2(\varepsilon) = (1 - \eta) C_1(\varepsilon)$$

The list of parameters includes:

- χ_1, χ_2: Correlation parameters for agents 1 and 2. Default values are 0 and 0.9, respectively.
- w_0^1, w_0^2: Initial endowments. Default values are 1.
- θ_0^1, θ_0^2: Consumers' initial shares of a representative firm. Default values are 0.5.
- ψ: CRRA risk parameter. Default value is 3.
- α: Returns to scale production function parameter. Default value is 0.6.
- A: Productivity of technology. Default value is 2.5.

- μ, σ: Mean and standard deviation of the log of the shock. Default values are -0.025 and 0.4, respectively.
- β: time preference discount factor. Default value is .96.
- `nb_points_integ` : number of points used for integration through Gauss-Hermite quadrature: default value is 10

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
from numba import njit, prange
from quantecon.optimize import root_finding
%matplotlib inline
```

```
#=========== Class: BCG for complete markets ===========#
class BCG_complete_markets:

    # init method or constructor
    def __init__(self,
                 χ1 = 0,
                 χ2 = 0.9,
                 w10 = 1,
                 w20 = 1,
                 θ10 = 0.5,
                 θ20 = 0.5,
                 ψ = 3,
                 α = 0.6,
                 A = 2.5,
                 μ = -0.025,
                 σ = 0.4,
                 β = 0.96,
                 nb_points_integ = 10):

        #=========== Setup ===========#
        # Risk parameters
        self.χ1 = χ1
        self.χ2 = χ2

        # Other parameters
        self.ψ = ψ
        self.α = α
        self.A = A
        self.μ = μ
        self.σ = σ
        self.β = β

        # Utility
        self.u = lambda c: (c**(1-ψ)) / (1-ψ)

        # Production
        self.f = njit(lambda k: A * (k ** α))
        self.Y = lambda ϵ, k: np.exp(ϵ) * self.f(k)

        # Initial endowments
        self.w10 = w10
        self.w20 = w20
        self.w0 = w10 + w20

        # Initial holdings
        self.θ10 = θ10
        self.θ20 = θ20

        # Endowments at t=1
        w11 = njit(lambda ϵ: np.exp(-χ1*μ - 0.5*(χ1**2)*(σ**2) + χ1*ϵ))
        w21 = njit(lambda ϵ: np.exp(-χ2*μ - 0.5*(χ2**2)*(σ**2) + χ2*ϵ))
        self.w11 = w11
        self.w21 = w21

        self.w1 = njit(lambda ϵ: w11(ϵ) + w21(ϵ))

        # Normal PDF
        self.g = lambda x: norm.pdf(x, loc=μ, scale=σ)

        # Integration
        x, self.weights = np.polynomial.hermite.hermgauss(nb_points_integ)
        self.points_integral = np.sqrt(2) * σ * x + μ

        self.k_foc = k_foc_factory(self)

    #=========== Optimal k ===========#
    # Function: solve for optimal k
    def opt_k(self, plot=False):
        w0 = self.w0

        # Grid for k
        kgrid = np.linspace(1e-4, w0-1e-4, 100)

        # get FONC values for each k in the grid
        kfoc_list = [];
        for k in kgrid:
            kfoc = self.k_foc(k, self.χ1, self.χ2)
            kfoc_list.append(kfoc)

        # Plot FONC for k
        if plot:
            fig, ax = plt.subplots(figsize=(8,7))
            ax.plot(kgrid, kfoc_list, color='blue', label=r'FONC for k')
            ax.axhline(0, color='red', linestyle='--')
            ax.legend()
            ax.set_xlabel(r'k')
            plt.show()

        # Find k that solves the FONC
        kk = root_finding.newton_secant(self.k_foc, 1e-2, args=(self.χ1, self.χ2)).root

        return kk

    #=========== Arrow security price ===========#
    # Function: Compute Arrow security price
    def q(self,ϵ,k):
        β = self.β
        ψ = self.ψ
        w0 = self.w0
        w1 = self.w1
        fk = self.f(k)
        g = self.g

        return β * ((w1(ϵ) + np.exp(ϵ)*fk) / (w0 - k))**(-ψ)


    #=========== Firm value V ===========#
    # Function: compute firm value V
    def V(self, k):
        q = self.q
        fk = self.f(k)
        weights = self.weights
        integ = lambda ϵ: np.exp(ϵ) * fk * q(ϵ, k)

        return -k + np.sum(weights * integ(self.points_integral)) / np.sqrt(np.pi)

    #=========== Optimal c ===========#
    # Function: Compute optimal consumption choices c
    def opt_c(self, k=None, plot=False):
        w1 = self.w1
        w0 = self.w0
        w10 = self.w10
        w11 = self.w11
        θ10 = self.θ10
        Y = self.Y
        q = self.q
        V = self.V
        weights = self.weights

        if k is None:
            k = self.opt_k()
```

```
        # Solve for the ratio of consumption η from the intertemporal B.C.
        fk = self.f(k)

        c1 = lambda ϵ: (w1(ϵ) + np.exp(ϵ)*fk)*q(ϵ,k)
        denom = np.sum(weights * c1(self.points_integral)) / np.sqrt(np.pi) + (w0 - k)

        w11q = lambda ϵ: w11(ϵ)*q(ϵ,k)
        num = w10 + θ10 * V(k) + np.sum(weights * w11q(self.points_integral)) / np.sqrt(np.pi)

        η = num / denom

        # Consumption choices
        c10 = η * (w0 - k)
        c20 = (1-η) * (w0 - k)
        c11 = lambda ϵ: η * (w1(ϵ)+Y(ϵ,k))
        c21 = lambda ϵ: (1-η) * (w1(ϵ)+Y(ϵ,k))

        return c10, c20, c11, c21


def k_foc_factory(model):
    ψ = model.ψ
    f = model.f
    β = model.β
    α = model.α
    A = model.A
    ψ = model.ψ
    w0 = model.w0
    μ = model.μ
    σ = model.σ

    weights = model.weights
    points_integral = model.points_integral

    w11 = njit(lambda ϵ, χ1, : np.exp(-χ1*μ - 0.5*(χ1**2)*(σ**2) + χ1*ϵ))
    w21 = njit(lambda ϵ, χ2: np.exp(-χ2*μ - 0.5*(χ2**2)*(σ**2) + χ2*ϵ))
    w1 = njit(lambda ϵ, χ1, χ2: w11(ϵ, χ1) + w21(ϵ, χ2))

    @njit
    def integrand(ϵ, χ1, χ2, k=1e-4):
        fk = f(k)
        return (w1(ϵ, χ1, χ2) + np.exp(ϵ) * fk) ** (-ψ) * np.exp(ϵ)

    @njit
    def k_foc(k, χ1, χ2):
        int_k = np.sum(weights * integrand(points_integral, χ1, χ2, k=k)) / np.sqrt(np.pi)

        mul = β * α * A * k ** (α - 1) / ((w0 - k) ** (-ψ))
        val = mul * int_k - 1

        return val

    return k_foc
```

37.3.1. Examples

Below we provide some examples of how to use `BCG_complete markets`.

37.3.1.1. 1st example

In the first example, we set up instances of BCG complete markets models.

We can use either default parameter values or set parameter values as we want.

The two instances of the BCG complete markets model, `mdl1` and `mdl2`, represent the model with default parameter settings and with agent 2's income correlation altered to be $\chi_2 = -0.9$, respectively.

```
# Example: BCG model for complete markets
mdl1 = BCG_complete_markets()
mdl2 = BCG_complete_markets(χ2=-0.9)
```

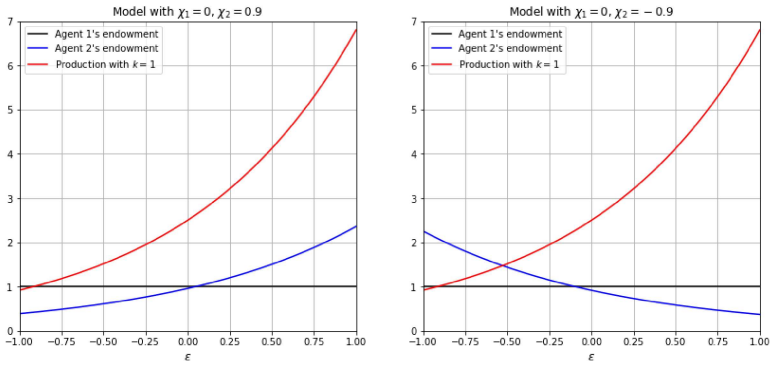
Let's plot the agents' time-1 endowments with respect to shocks to see the difference in the two models:

```
#==== Figure 1: HH endowments and firm productivity ====#
# Realizations of innovation from -3 to 3
epsgrid = np.linspace(-1,1,1000)


fig, ax = plt.subplots(1,2,figsize=(14,6))
ax[0].plot(epsgrid, mdl1.w11(epsgrid), color='black', label='Agent 1\'s endowment')
ax[0].plot(epsgrid, mdl1.w21(epsgrid), color='blue', label='Agent 2\'s endowment')
ax[0].plot(epsgrid, mdl1.Y(epsgrid,1), color='red', label=r'Production with $k=1$')
ax[0].set_xlim([-1,1])
ax[0].set_ylim([0,7])
ax[0].set_xlabel(r'$\epsilon$',fontsize=12)
ax[0].set_title(r'Model with $\chi_1 = 0$, $\chi_2 = 0.9$')
ax[0].legend()
ax[0].grid()

ax[1].plot(epsgrid, mdl2.w11(epsgrid), color='black', label='Agent 1\'s endowment')
ax[1].plot(epsgrid, mdl2.w21(epsgrid), color='blue', label='Agent 2\'s endowment')
ax[1].plot(epsgrid, mdl2.Y(epsgrid,1), color='red', label=r'Production with $k=1$')
ax[1].set_xlim([-1,1])
ax[1].set_ylim([0,7])
ax[1].set_xlabel(r'$\epsilon$',fontsize=12)
ax[1].set_title(r'Model with $\chi_1 = 0$, $\chi_2 = -0.9$')
ax[1].legend()
ax[1].grid()

plt.show()
```



Let's also compare the optimal capital stock, k, and optimal time-0 consumption of agent 2, c_0^2, for the two models:

```
# Print optimal k
kk_1 = mdl1.opt_k()
kk_2 = mdl2.opt_k()

print('The optimal k for model 1: {:.5f}'.format(kk_1))
print('The optimal k for model 2: {:.5f}'.format(kk_2))

# Print optimal time-0 consumption for agent 2
c20_1 = mdl1.opt_c(k=kk_1)[1]
c20_2 = mdl2.opt_c(k=kk_2)[1]

print('The optimal c20 for model 1: {:.5f}'.format(c20_1))
print('The optimal c20 for model 2: {:.5f}'.format(c20_2))
```

```
The optimal k for model 1: 0.14235
The optimal k for model 2: 0.13791
```

```
The optimal c20 for model 1: 0.90205
The optimal c20 for model 2: 0.92862
```

37.3.1.2. 2nd example

In the second example, we illustrate how the optimal choice of k is influenced by the correlation parameter χ_i.

We will need to install the `plotly` package for 3D illustration. See https://plotly.com/python/getting-started/ (https://plotly.com/python/getting-started/) for further instructions.

```
# Mesh grid of χ
N = 30
χ1grid, χ2grid = np.meshgrid(np.linspace(-1,1,N),
                             np.linspace(-1,1,N))

k_foc = k_foc_factory(mdl1)

# Create grid for k
kgrid = np.zeros_like(χ1grid)

w0 = mdl1.w0

@njit(parallel=True)
def fill_k_grid(kgrid):
    # Loop: Compute optimal k and
    for i in prange(N):
        for j in prange(N):
            X1 = χ1grid[i, j]
            X2 = χ2grid[i, j]
            k = root_finding.newton_secant(k_foc, 1e-2, args=(X1, X2)).root
            kgrid[i, j] = k
```

```
%%time
fill_k_grid(kgrid)
```

```
CPU times: user 2.94 s, sys: 15.4 ms, total: 2.95 s
Wall time: 2.94 s
```

```
%%time
# Second-run
fill_k_grid(kgrid)
```

```
CPU times: user 21 ms, sys: 31 µs, total: 21.1 ms
Wall time: 12.6 ms
```

```
#=== Example: Plot optimal k with different correlations ===#

from IPython.display import Image
# Import plotly
import plotly.graph_objs as go

# Plot optimal k
fig = go.Figure(data=[go.Surface(x=χ1grid, y=χ2grid, z=kgrid)])
fig.update_layout(scene = dict(xaxis_title='x - χ1',
                               yaxis_title='y - χ2',
                               zaxis_title='z - k',
                               aspectratio=dict(x=1,y=1,z=1)))
fig.update_layout(width=500,
                  height=500,
                  margin=dict(l=50, r=50, b=65, t=90))
fig.update_layout(scene_camera=dict(eye=dict(x=2, y=-2, z=1.5)))

# Export to PNG file
Image(fig.to_image(format="png"))
# fig.show() will provide interactive plot when running
# notebook locally
```