LECTURE 4 – MARCH 27, 2024

LINEAR SYSTEMS

GIVEN:

- A, $m \times n$ MATRIX
- x, $n \times 1$ UNKNOWN MATRIX
- b, $m \times 1$ MATRIX

WE WANT TO SOLVE

$$A \cdot x = b \iff \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

BY THE ROUCHÉ–CAPELLI THEOREM, SOLVABILITY IS EQUIV. TO

$$\text{rank } A = \text{rank } [A|b]$$

[ ]:
```python
Ab = np.concatenate((A, b), axis=1)
rA = np.linalg.matrix_rank(A)
rAb = np.linalg.matrix_rank(Ab)
rA == rAb
```

A IS SQUARE

IF $\text{rank } A = \text{rank } [A|b] = m = n$, WE HAVE A UNIQUE SOLUTION

$$A \cdot x = b \iff A^{-1} \cdot A \cdot x = A^{-1} \cdot b \iff x = A^{-1} \cdot b$$

IDENTITY MATRIX

[ ]:
```python
# Solve the system
Ainv = np.linalg.inv(A)
x = Ainv @ b
x
```

ONE PERIOD MARKETS

CONSIDER A ONE PERIOD BETWEEN $t=0$ (TODAY) AND $T=1$ (TOMORROW). WE ARE UNCERTAIN ON THE FUTURE STATE OF THE MARKET THAT CAN BELONG TO A FINITE SET

STATES → $\Omega = \{\omega_1, \omega_2, \dots, \omega_m\}$

PROBABILITY DISTRIBUTION → $P = (p_1, p_2, \dots, p_m)$

IN THE MARKET THERE ARE A RISK-FREE BOND B AND n RISKY STOCKS $S^1, \dots, S^n$.

THE (RANDOM) PAYOFFS AT TIME $T=1$ CAN BE ORGANIZED IN A MATRIX

$T=1$ PAYOFFS

$$
M_1 = \begin{array}{c} \\ \\ \\ \\ \\ \end{array}
\begin{array}{ccccc|cc}
\text{BOND} & \text{STOCK1} & \cdots & \text{STOCK}n & & \text{STATES} & \text{PROBABILITIES} \\
B_1 & S_1^1(\omega_1) & \cdots & S_1^n(\omega_1) & & \omega_1 & p_1 \\
B_1 & S_1^1(\omega_2) & \cdots & S_1^n(\omega_2) & & \omega_2 & p_2 \\
\vdots & \vdots & & \vdots & & \vdots & \vdots \\
B_1 & S_1^1(\omega_m) & \cdots & S_1^n(\omega_m) & & \omega_m & p_m
\end{array}
$$

THE PRICES AT TIME $t=0$ CAN BE ORGANIZED IN A VECTOR

$t=0$ PRICES

$$
\begin{array}{cccc}
& \text{BOND} & \text{STOCK1} & & \text{STOCK}n \\
M_0 = [& B_0 & S_0^1 & \cdots & S_0^n]
\end{array}
$$

A <u>PORTFOLIO</u> IS A VECTOR

$$\vartheta = [\beta \quad \alpha_1 \quad \cdots \quad \alpha_n]^T$$

WHERE β AND α_i ARE THE UNITS OF BOND AND STOCK TO BUY (IF POSITIVE) OR SHORT-SELL (IF NEGATIVE).

THE PORTFOLIO θ WILL HAVE A PAYOFF AT TIME $T=1$ AND A PRICE A TIME $t=0$

$T=1$ PAYOFF $\quad V_1^\theta = M_1 \cdot \theta$ RANDOM VARIABLE

$t=0$ PRICE $\quad V_0^\theta = M_0 \cdot \theta$ NUMBER

NO-ARBITRAGE: WE CANNOT FIND A PORTFOLIO θ SUCH THAT

(a) $\underbrace{V_0^\theta \leq 0}_{\text{WE ARE PAID TODAY OR WE DON'T PAY ANITHING}}$ AND $\underbrace{V_1^\theta \geq 0}_{\text{WE DON'T PAY AND HAVE CHANCE TO RECEIVE SOMETHING TOMORROW}}$ (WITH $>$ IN AT LEAST A STATE)

(b) $\underbrace{V_0^\theta < 0}_{\text{WE ARE PAID TODAY}}$ AND $\underbrace{V_1^\theta = 0}_{\text{WE DON'T PAY ANITHING TOMORROW}}$

A CONTINGENT CLAIM IS A CONTRACT WHOSE PAYOFF C_1 AT TIME $T=1$ DEPENDS ON THE STATES IN Ω: IT IS A RANDOM VARIABLE.

PROBLEM: HOW DO WE FIND ITS PRICE C_0?

NO-ARBITRAGE PRICE THROUGH REPLICATION:

IF WE CAN FIND A PORTFOLIO θ THAT REPLICATES C_1

$T=1$ PAYOFF $\quad V_1^\theta = M_1 \cdot \theta = C_1$

THEN WE TAKE C_0 EQUAL TO THE PRICE OF SUCH PORTFOLIO

$t=0$ PRICE $\quad V_0^\theta = M_0 \cdot \theta = \boxed{C_0}$ ← TO FIND!

EXAMPLE: ONE PERIOD BINOMIAL MARKET

CONSIDER A TWO STATES MARKET OVER TWO DATES: $t=0$ (TODAY) AND $T=1$ (TOMORROW), COMPOSED BY A RISK-FREE BOND AND A RISKY STOCK

$T=1$ PAYOFFS

$$\begin{array}{cccc} & \text{BOND} & \text{STOCK} & \text{STATES} \\ M_1 = & \underbrace{\begin{matrix} 11 \\ 11 \end{matrix}}_{= B_1} & \underbrace{\begin{matrix} 20 \\ 5 \end{matrix}}_{= S_1} & \begin{matrix} u & \text{"UP"} \\ d & \text{"DOWN"} \end{matrix} \end{array}$$

$t=0$ PRICES

$$\begin{array}{ccc} & \text{BOND} & \text{STOCK} \\ M_0 = & \underbrace{10}_{= B_0} & \underbrace{10}_{= S_1} \end{array}$$

$\boxed{\text{IMPORTANT: THIS MARKET IS ARBITRAGE FREE.}}$

A CALL OPTION ON THE STOCK WITH MATURITY $T=1$ AND STRIKE PRICE $K=10$ IS A CONTRACT THAT GIVES THE HOLDER THE RIGHT BUT NOT THE OBLIGATION TO BUY THE STOCK AT TIME $T=1$ FOR THE PRICE K.

$T=1$ PAYOFF

$$C_1 = \max(S_1 - K, 0) = \begin{array}{c|c} & \text{STATES} \\ \begin{bmatrix} 10 \\ 0 \end{bmatrix} & \begin{matrix} u \\ d \end{matrix} \end{array}$$

REPLICATION PROBLEM: CAN WE FIND A PORFOLIO θ OF BOND AND STOCK THAT REPLICATES C_1?

WE NEED TO SOLVE

$$V_1^\theta = M_1 \cdot \theta = \begin{bmatrix} 11 & 20 \\ 11 & 5 \end{bmatrix} \cdot \begin{bmatrix} \beta \\ \alpha \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \end{bmatrix} = C_1$$

$\beta \to$ UNITS OF BOND

$\alpha \to$ UNITS OF STOCK

```
[ ]: import numpy as np

     # Build the market
     # Bond
     B1 = np.array([11, 11]) # Payoff
     B0 = 10 # Price
     # Stock
     S1 = np.array([20, 5]) # Payoff
     S0 = 10 # Price
```

```
[ ]: # Payoffs at time T = 1
     M1 = np.array([B1, S1]).T
     M1
```

```
[ ]: # Prices at time t = 0
     M0 = np.array([B0, S0]).reshape(1,2)
     M0
```

```
[ ]: # Payoff of the call option at time T=1
     K = 10 # Strike price
     C1 = np.maximum(S1 - K, 0)
     C1 = C1.reshape(2,1)
     C1
```

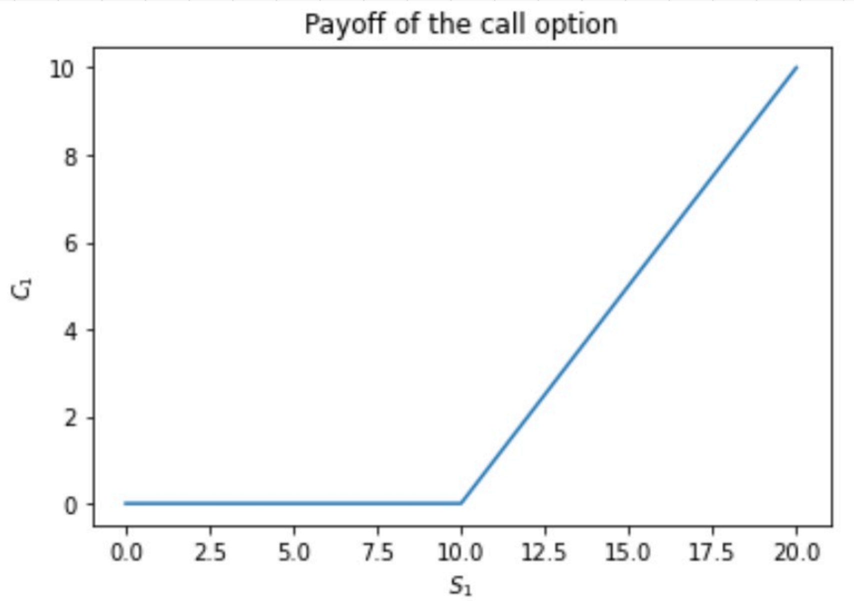
WE PLOT THE CONTRACT FUNCTION OF THE CALL

$$y = \max\{x - K, 0\}$$

[ ]:
```
# Plot the payoff of the call
import matplotlib.pyplot as plt
```

[ ]:
```
# Tabulate the function for x = 0, 1, ..., 20
x = np.arange(21)    <- WE CAN ALSO USE np.linspace()
y = np.maximum(x - K, 0)
```

[ ]:
```
plt.title('Payoff of the call option')
plt.xlabel('$S_1$')
plt.ylabel('$C_1$')
plt.plot(x, y)
```



THE MARKET IS COMPLETE IF rank(M1) = ℓ (ITS ROWS).

[ ]:
```python
# Verify that this is a complete market
r = np.linalg.matrix_rank(M1)
rows, cols = M1.shape
if r == rows:
    print('The market is complete')
else:
    print('The market is incomplete')

if r == rows and r == cols:
    print('No redundant assets')
```

REPLICATION OF THE CALL: WE NEED TO SOLVE

$$M_1 \cdot \theta = C_1 \iff \theta = M_1^{-1} \cdot C_1$$

SQUARE AND WITH FULL RANK

[ ]:
```python
# Replication of the payoff of the call
M1inv = np.linalg.inv(M1)
theta = M1inv @ C1
theta
```

NO-ARBITRAGE PRICE OF THE CALL

$$C_0 = M_0 \cdot \theta$$

[ ]:
```python
# No-arbitrage price of the call
C0 = M0 @ theta
C0 = C0[0, 0]   # <- EXTRACTS THE VALUE FROM THE 1x1 MATRIX
C0
```

IN A COMPLETE MARKET WE CAN REPLICATE EVERY CONTINGENT CLAIM: CHOOSING A SUITABLE ϑ WE CAN REACH EVERY CONTINGENT CLAIM $C1 \in \mathbb{R}^2$.

TO SEE THIS, WE GENERATE 1000 RANDOM PORTFOLIOS WITH INDEPENDENT AND UNIFORM DISTRIBUTION ON $[-2,2]$. WE USE THE NumPy RANDOM GENERATOR: IT ALLOWS TO SAMPLE FROM KNOWN DISTRIBUTIONS.

REMARK: IF YOU USE NumPy, USE ITS RANDOM GENERATOR. EVERYTHING IS OPTIMIZED FOR ARRAYS!!

[ ]:
```
# Create the random generator with NumPy
from numpy.random import default_rng
rng = default_rng(100)
```
↖ SEED

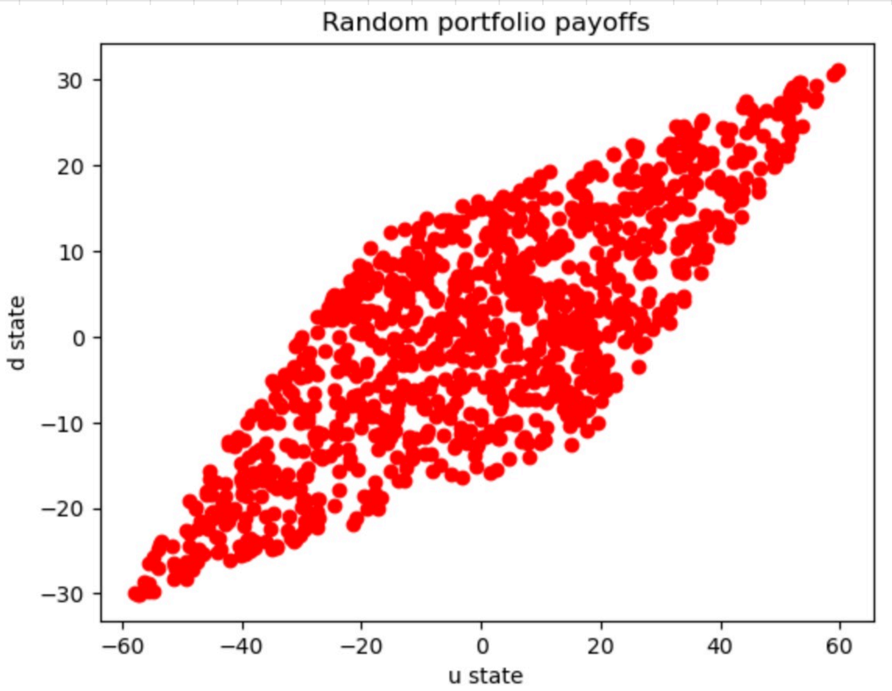
[ ]:
```
# Generate the portfolios payoffs
n = 1000
b = rng.uniform(low=-2, high=2, size=n)
a = rng.uniform(low=-2, high=2, size=n)

# Create a list of payoffs
payoffs = []
for i in range(n):
    payoffs.append(b[i] * B1 + a[i] * S1)

# Transform the list in an array
payoffs = np.array(payoffs)
```

```
[]: # Plot the portfolio payoffs
    plt.clf()
    plt.title('Random portfolio payoffs')
    plt.xlabel('u state')
    plt.ylabel('d state')
    plt.scatter(payoffs[:, 0], payoffs[:, 1], c='red')
```

EXERCISE: REPEATE USING

```
a = rng.normal(size = n)
b = rng.normal(size = n)
```

SAMPLES FROM TWO INDEPENDENT STANDARD NORMALS



HOMEWORK

GIVEN THE MARKET

T = 1
PAYOFFS

$$
M1 = \begin{array}{cccc} B1 & S1\_1 & S2\_1 & S3\_1 \\ \end{array}
\begin{bmatrix} 10 & 30 & 15 & 60 \\ 10 & 20 & 5 & 40 \\ 10 & 10 & 0 & 20 \end{bmatrix}
$$

t = 0
PRICES

$$
M0 = \begin{array}{cccc} B0 & S1\_0 & S2\_0 & S3\_0 \\ \end{array}
\begin{bmatrix} 10 & 20 & 6 & 40 \end{bmatrix}
$$

RECALL TO IMPORT numpy AND itertools

① USE THE FUNCTION non_redundant() (LAST HOMEWORK) TO EXTRACT A SUBMARKET WITH 3 NON-REDUNDANT ASSETS M1s (PAYOFFS) WITH CORRESPONDING PRICE VECTOR M0s (PRICES) OBTAINED SELECTING A SUBSET OF COLUMNS OF M0.

② VERIFY THAT THE NEW MARKET (M0s, M1s) IS COMPLETE.

③ CREATE AN ARRAY FOR THE VALUE OF THE STRIKE PRICE K RANGING IN $[\overset{0}{0}, \overset{1}{1}, \dots, \overset{20}{20}]$ ← INDICES AND FOR EACH VALUE OF K[i] CONSIDER A CALL OPTION ON S1_1 WITH PAYOFF

$$C1 = np.maximum(S1\_1 - K[i], 0).$$

FOR EACH i COMPUTE:

ⓐ THE REPLICATING PORTFOLIO

ⓑ THE NO-ARBITRAGE PRICE OF THE CALL

ⓒ INSERT THE NO-ARBITRAGE PRICE IN A LIST

④ PLOT THE GRAPH OF THE PRICE AS A FUNCTION OF K (K ON THE x AXIS, C0 ON THE y AXIS).

EXERCISE: ONE PERIOD TRINOMIAL MARKET MODEL

CONSIDER THE MARKET ON 3 STATES $\Omega = \{u, m, d\}$:

T = 1
PAYOFFS

$$M1 = \begin{array}{cc} \text{BOND} & \text{STOCK} \\ \left[\begin{array}{c} 11 \\ 11 \\ 11 \end{array}\right. & \left.\begin{array}{c} 20 \\ 10 \\ 5 \end{array}\right] \end{array} \quad \begin{array}{c} \text{STATES} \\ u \\ m \\ d \end{array}$$

WE DO NOT NEED PRICES IN THIS EXERCISE.

[ ]:

```
import numpy as np

# Bond
B1 = np.array([11, 11, 11])    # Payoff
# Stock
S1 = np.array([20, 10, 5])    # Payoff

# Payoffs at time T = 1
M1 = np.array([B1, S1]).T
M1
```

[ ]:

```
# Verify that this is an incomplete market
r = np.linalg.matrix_rank(M1)
rows, cols = M1.shape
if r == rows:
    print('The market is complete')
else:
    print('The market is incomplete')
```

IN AN INCOMPLETE MARKET THERE ARE CONTINGENT CLAIMS WHOSE PAYOFF CANNOT BE REPLICATED.

WE GENERATE AT RANDOM SOME REPLICABLE PAYOFFS.

```
[ ]: # Create the random generator with NumPy
     from numpy.random import default_rng
     rng = default_rng(100)
```

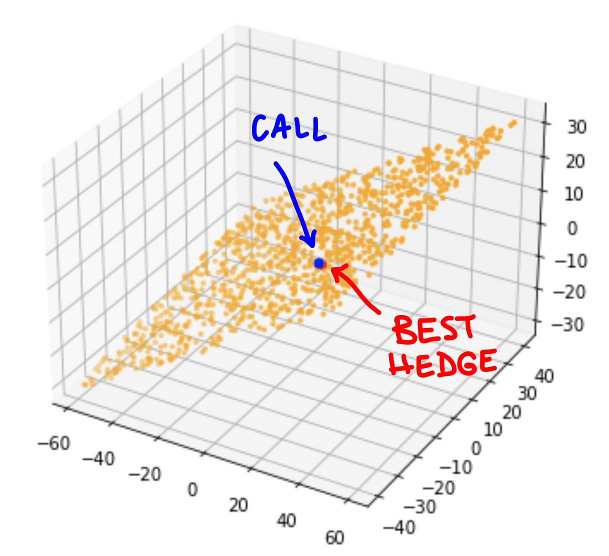
SEED

```
[ ]: # Generate the portfolio payoffs
     n = 1000
     b = rng.uniform(low=-2, high=2, size=n)
     a = rng.uniform(low=-2, high=2, size=n)

     # Create a list of payoffs
     payoffs = []
     for i in range(n):
         payoffs.append(b[i] * B1 + a[i] * S1)

     payoffs = np.array(payoffs)
```

THE REPLICATED PAYOFFS ARE POINTS IN THE 3D SPACE

```
[ ]: # Scatter plot in the 3D space
     import matplotlib.pyplot as plt
     from mpl_toolkits.mplot3d import Axes3D

     # Plot the replication portfolio space
     fig = plt.figure(figsize=(10, 6))
     ax = fig.add_subplot(111, projection='3d')
     ax.scatter(payoffs[:, 0], payoffs[:, 1], payoffs[:, 2])
```

THE REPLICATION SPACE IS A PLANE FROM THE ORIGIN IN THE 3D SPACE: IT IS A LINEAR SUBSPACE OF $\mathbb{R}^3$!



THE CALL OPTION ON S1 WITH STRIKE $K = 10$ IS NOT REPLICABLE: ITS PAYOFF IS A POINT OUTSIDE THE PLANE. WE CAN SEARCH A "BEST HEDGE" COMPUTING THE PORTFOLIO WHOSE PAYOFF ON THE PLANE MINIMIZES THE MEAN SQUARED ERROR:

$$\min_{\theta} \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \Big(\underbrace{V_1^\theta(\omega)}_{\text{PAYOFF OF THE PORTFOLIO } \theta} - C_1(\omega) \Big)^2$$

PORTFOLIOS

RECALLING THAT $V_1^\theta = M_1 \cdot \theta$.

```
[]: # Payoff of the call
     K = 10
     C1 = np.maximum(S1 - K, 0)
```

TO MINIMIZE

```
[]: # Ordinary least-square regression
     reg = np.linalg.lstsq(M1, C1, rcond=-1)
     theta_best = reg[0]
     theta_best
```