# ECON 607 Assignment 8

Andrew Girgis

November 20, 2023

# Contents

# List of Figures

# 1 Numerical Optimization

## 1.1 Solving constrained optimization problem analytically

$$v^s = \max_{p,b} \frac{b}{b+s}(p-c)$$

subject to

$$v^b = \frac{s}{b+s}(u-p)$$

We will perform the Lagrangian.

$$\mathcal{L} = \frac{b}{b+s}(p+c) - \lambda(\frac{s}{b+s}(u-p) - v^b)$$

First we find the FOC's:

$$\frac{\partial \mathcal{L}}{\partial p} = (\frac{b}{b+s})(\frac{\lambda s}{b+s}) = 0 \tag{1}$$

$$\frac{\partial \mathcal{L}}{\partial b} = \frac{s}{(b+s)^2}(p-c) + \frac{\lambda s}{(b+s)^2}(u-p) = 0 \tag{2}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \frac{s}{b+s}(u-p) - v^b = 0 \tag{3}$$

Looking at the first foc in equation 1 we can simplify:

$$\implies \lambda s = -b$$
$$\implies b = -\lambda s \tag{4}$$

Looking at the second foc in equation 2 we can simplify:

$$\implies \frac{s(p-c) + \lambda s(u-p)}{(b+s)^2} = 0$$
$$\implies sp - sc + \lambda su - \lambda sp = 0$$

We know that $\lambda s = -b$ in equation 1 therefore;

$$\implies sp - sc + \lambda su - (-b)p = 0$$
$$\implies sp - sc + \lambda su + bp = 0$$
$$\implies sp + bp = sc - \lambda su$$
$$\implies p(s+b) = s(c - \lambda u)$$
$$\implies p = \frac{s(c-\lambda u)}{(s+b)}$$
$$\implies p = \frac{s(c-\lambda u)}{s - \lambda s}$$

$$\implies p = \frac{c - \lambda u}{1 - \lambda} \tag{5}$$

Looking at the Third foc in equation 3 we find:

$$\implies v^b = \frac{s}{b+s}(u - p)$$

We know that $b = -\lambda s$ in equation 4 therefore;

$$\implies v^b = \frac{s(u-p)}{s - \lambda s}$$

$$\implies v^b = \frac{u-p}{1-\lambda}$$

$$\implies v^b(1-\lambda) = u - p$$

We know that $p = \frac{c-\lambda u}{1-\lambda}$ in equation 5 therefore;

$$\implies v^b(1-\lambda) = u - \frac{c - \lambda u}{1-\lambda}$$

After simplification of lambda we get;

$$\implies \lambda = 1 - \sqrt{\frac{u-c}{v^b}} \tag{6}$$

Plugging equation 6 into equation 5 we get;

$$\implies p = \frac{c - (1 + \sqrt{\frac{u-c}{v^b}})u}{1 - 1 + \sqrt{\frac{u-c}{v^b}}}$$

Therefore optimal p is;

$$\implies p^* = \frac{c - (1 + \sqrt{\frac{u-c}{v^b}})u}{\sqrt{\frac{u-c}{v^b}}} \tag{7}$$

Plugging equation 6 into equation 4 we get optimal b is;

$$\implies b^* = -(1 - \sqrt{\frac{u-c}{v^b}})s \tag{8}$$

## 1.2 Solution using computer program

Optimal allocation of goods:
Quantity of b: 2.00000020 units
Quantity of p: 0.72222220 units
Optimal utility: 0.37037037
See code used in Section 2

### 1.3 Comparison between computer program answer versus exact solution

When plugging $s = 1, u = 1, c = \frac{1}{6}, v^b = \frac{5}{54}$ into equation 7 and equation 8 we get;

$$\implies p^* = \frac{\frac{1}{6} - (1 + \sqrt{\frac{1-\frac{1}{6}}{\frac{5}{54}}})}{\sqrt{\frac{1-\frac{1}{6}}{\frac{5}{54}}}}$$

$$\implies p^* = 0.7222$$

And

$$\implies b^* = -(1 - \sqrt{\frac{1-\frac{1}{6}}{\frac{5}{54}}})$$

$$\implies b^* = 2$$

when plugging $p^* = 0.7222$ and $b^* = 2$ into $v^s = \frac{b}{b+s}(p - c)$ we get;

$$\implies v^s = \frac{2}{2+1}(0.7222 - \frac{1}{6})$$

$$\implies v^s = 0.370370226$$

In our computer program we obtained the answers;
Optimal quantity of b: 2.00000020 units
Optimal quantity of p: 0.72222220 units
Optimal utility: 0.37037037

Therefore, it is clear that the exact numbers and those provided by the computer program are not precisely the same. The number generated by the computer for the optimal 'b' has a difference 0.00000020 greater than the exact answer. Additionally, the computer's value for the optimal 'p' stops at the eighth decimal place, whereas the exact answer continues as a continuous decimal. There is a slight difference in the optimal utility calculated by the computer compared to the real answer. These differences arise from the inherent nature of computer arithmetic, where floating-point numbers introduce rounding errors, impacting calculations and resulting in very small deviations from the exact answer. While this error is minimal and may not significantly affect output, especially in the case of utility, it is crucial to note that in more precise applications, these deviations can have more substantial impacts. Therefore, it is essential to understand whether it is necessary to account for rounding errors in advance or not.

### 1.4 Matlab Plots

Figure 1: Optimal price as a function of $v^b$

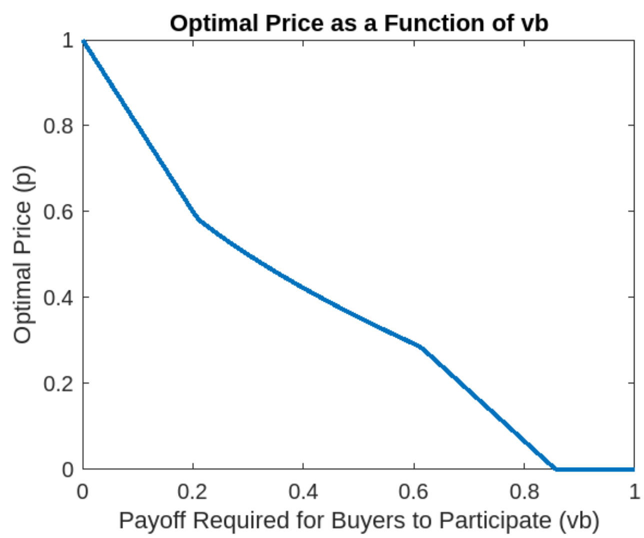
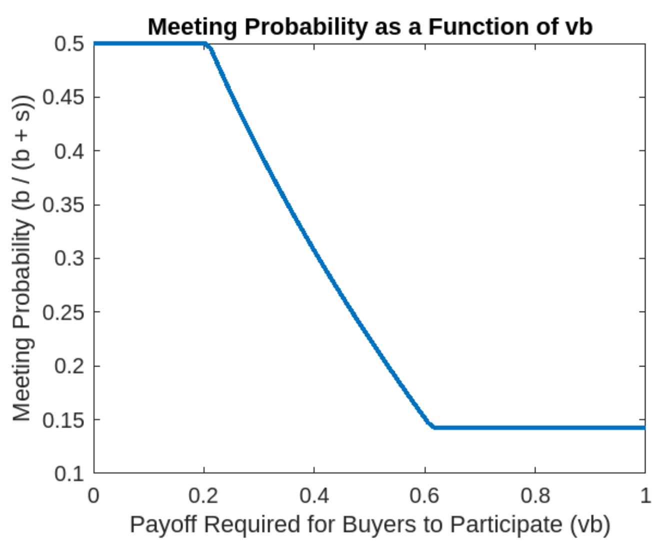



Figure 2: Meeting probability as a function of $v^b$



The decrease in both the optimal price ($p^*$) and the meeting probability ($\frac{b}{b+s}$) as the payoff required for buyers to participate in the market ($v_b$) increases can be explained by the economic intuition behind the optimization problem.

The constraint reflects the buyer's decision to participate in the market, which depends on the payoff from participating ($\frac{b}{b+s}(p-c)$) being greater than or equal to the payoff from not participating ($v_b$). As you increase $v_b$, buyers require a higher payoff from participating to outweigh the value of their outside option. This leads to a lower optimal price ($p$).

Simultaneously, as $v_b$ increases, the constraint becomes more binding, reducing the feasible set for $p$ and $b$. This constraint forces a trade-off between $p$ and $b$. The decrease in $p$ and $b$ is a result of finding a solution that satisfies the constraint while maximizing the objective function.

Therefore, increasing the payoff required for buyers to participate ($v_b$) makes it more difficult for buyers to find the market attractive, leading to a lower optimal price and a lower measure of participating buyers. This relationship is reflected in the figure 1 and figure 2.

## 2 Code

```
% Define some parameters

s = 1;
% Where s is the fixed measure of participating sellers
u = 1;
%Where u is the the buyers utility from consuming the good
c = 1/6;
%Where c is the sellers cost
v_b = 5/54;
%Where u_b is a constraint, value to a buyer of not participating
%(i.e., their outside option)

%

%x = [b, p];

% Define the utility function
utility = @(x) -x(1)/(x(1) + s) * (x(2) - c);


% Initial guess for the decision variables (x and y)
x0 = [14, .5];  % Initial guess
%The initial guess can be any *reasonable* guess at what the decision
%variables could be within the bounds

% Define lower and upper bounds for the decision variables
lb = [0, c];  % Non-negativity constraints
ub = [inf, u];      % upper bound of b is infinity


% Create an options structure (optional)
options = optimoptions('fmincon', 'Display', 'iter');

% Define the inequality constraint
nonlcon = @(x)deal([], s/(x(1) + s) * (u - x(2)) - vb); % Budget constraint

% Use fmincon to solve the utility maximization problem
[x, fval, exitflag, output, lambda] = fmincon(utility, x0, [], [], [], [], lb, ub, nonlcon,

% Display the results
fprintf('Optimal allocation of goods:\n');
fprintf('Quantity of b: %.8f units\n', x(1));
fprintf('Quantity of p: %.8f units\n', x(2));
```

```
fprintf('Optimal utility: %.8f\n', -fval);  % Negative sign due to maximization
```