Quantitative Economics Workshop Paris

Prelude to Dynamic Programming

John Stachurski

September 2022

Introduction

Summary of this lecture:

- Linear equations
- Fixed point theory
- Infinite horizon job search

Operations on real numbers such as $|\cdot|$ and $\vee$ are applied to vectors element-by-element

Example.

$$a = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \quad \Longrightarrow \quad |a| = \begin{pmatrix} |a_1| \\ \vdots \\ |a_n| \end{pmatrix}$$

$$a \vee b = \begin{pmatrix} a_1 \vee b_1 \\ \vdots \\ a_n \vee b_n \end{pmatrix} \quad \text{and} \quad a \wedge b = \begin{pmatrix} a_1 \wedge b_1 \\ \vdots \\ a_n \wedge b_n \end{pmatrix}$$

etc.

Linear Equations

Given one-dimensional equation $x = ax + b$, we have

$$|a| < 1 \quad \implies \quad x^* = \frac{b}{1-a} = \sum_{k \geqslant 0} a^k b$$

How can we extend this beyond one dimension?

We define the **spectral radius** of square matrix A as

$$r(A) := \max\{|\lambda| : \lambda \text{ is an eigenvalue of } A\}$$

Key idea:

- $r(A) < 1$ is a generalization of $|a| < 1$

Neumann Series Lemma

Suppose b is a column vector in $\mathbb{R}^n$ and A is $n \times n$

Let I be the $n \times n$ identity matrix

Theorem. If $r(A) < 1$, then

1. $I - A$ is nonsingular,
2. the sum $\sum_{k \geqslant 0} A^k$ converges,
3. $(I - A)^{-1} = \sum_{k \geqslant 0} A^k$, and
4. the vector equation $x = Ax + b$ has the unique solution

$$x^* := (I - A)^{-1} b = \sum_{k \geqslant 0} A^k b$$

Intuitive idea: with $S := \sum_{k \geqslant 0} A^k$, we have

$$I + AS = I + A(I + A + \cdots) = I + A + A^2 + \cdots = S$$

Rearranging $I + AS = S$ gives $S = (I - A)^{-1}$

The equation $x = Ax + b$ is equivalent to $(I - A)x = b$

Unique solution is $x^* = (I - A)^{-1} b = Sb$, as claimed

Fixed Points

To solve more complex equations we use **fixed point theory**

Recall that, if S is any set then

- T is a **self-map** on S if T maps S into itself
- $x^* \in S$ is called a **fixed point** of T in S if $Tx^* = x^*$

Example. Every x in set S is fixed under the **identity map**

$$I \colon x \mapsto x$$

Example. If $S = \mathbb{N}$ and $Tx = x + 1$, then T has no fixed point

Example. If $S = \mathbb{R}$ and $Tx = x^2$, then T has fixed points at $0, 1$

Example. If $S = \mathbb{R}^n$ and $Tx = Ax + b$, then

$$r(A) < 1 \implies x^* := (I - A)^{-1}b \text{ is the unique f.p. of } T \text{ in } S$$

Example. If $S \subset \mathbb{R}$, $Tx = x \iff T$ meets the 45 degree line

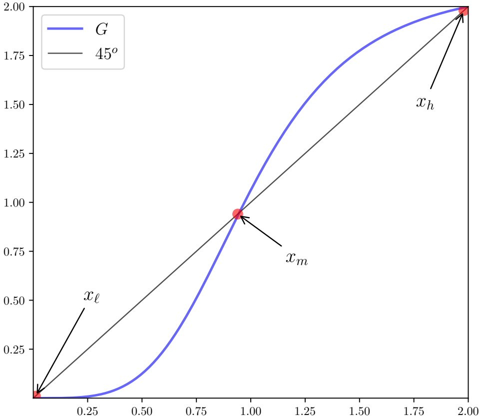


Figure: Graph and fixed points of $G\colon x \mapsto 2.125/(1 + x^{-4})$

Given self-map T on S, common to

- write Tx instead of $T(x)$ and
- call T an **operator** rather than a function

Key idea:

$$\text{solving equation } x = Tx \iff \text{finding fixed points of } T$$

Example. If $S = \mathbb{R}^n$ and $Tx = Ax + b$, then

$$x^* \text{ solves equation } x = Ax + b \iff x^* \text{ is a fixed point of } T$$

(But fixed point theory is mainly for nonlinear equations)

Point on notation:

- $T^2 = T \circ T$
- $T^3 = T \circ T \circ T$
- etc.

Example. $Tx = Ax + b$ implies $T^2 x = A(Ax + b) + b$

Self-map T is called **globally stable** on S if

1. T has a unique fixed point x^* in S and
2. $T^k x \to x^*$ as $k \to \infty$ for all $x \in S$

Example. If $S = \mathbb{R}^n$ and $Tx = Ax + b$, then

$$T^k x = A^k x + A^{k-1} b + A^{k-2} b + \cdots + Ab + b \qquad (x \in S, k \in \mathbb{N})$$

If $r(A) < 1$, then $A^k x \to 0$ and $\sum_{i=0}^{k} A^i \to (I - A)^{-1}$, so

$$\lim_{k \to \infty} T^k x = \lim_{k \to \infty} \left[A^k x + \sum_{i=0}^{k} A^{i-1} b \right] = (I - A)^{-1} b = x^*$$

Example. Consider Solow–Swan growth dynamics

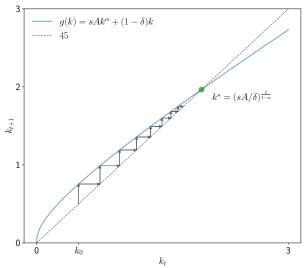
$$k_{t+1} = g(k_t) := sAk_t^{\alpha} + (1 - \delta)k_t, \qquad t = 0, 1, \ldots,$$

where

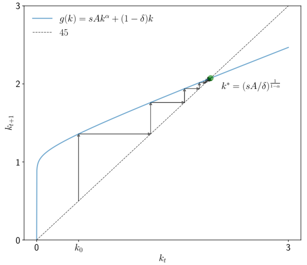
- k_t is capital stock per worker,
- $A, \alpha > 0$ are production parameters, $\alpha < 1$
- $s > 0$ is a savings rate, and
- $\delta \in (0, 1)$ is a rate of depreciation

Iterating with g from k_0 generates a time path for capital stock

The map g is globally stable on $(0, \infty)$



$g(k) = sAk^{\alpha} + (1-\delta)k$
45
$k^* = (sA/\delta)^{\frac{1}{1-\alpha}}$
k_{t+1}
k_t
k_0
0
1
2
3



$g(k) = sAk^{\alpha} + (1-\delta)k$
45
$k^* = (sA/\delta)^{\frac{1}{1-\alpha}}$
k_{t+1}
k_t
k_0
0
1
2
3

Note from last slide

- If g is flat near k^*, then $g(k) \approx k^*$ for k near k^*
- A flat function near the fixed point $\implies$ fast convergence

Conversely

- If g is close to the 45 degree line near k^*, then $g(k) \approx k$
- Close to 45 degree line means high persistence, slow convergence

Given a self-map T on S, we typically ask

- Does T have at least one fixed point on S (existence)?
- Does T have at most one fixed point on S (uniqueness)?
- How can we compute fixed points of T?

For the last question, we seek an algorithm

Then we investigate its properties

Successive Approximation

A natural algorithm for approximating the fixed point in S:

---

fix x_0 and $k = 0$
while *some stopping condition fails* **do**
  $x_{k+1} \leftarrow T x_k$
  $k \leftarrow k + 1$
end
return x_k

If T is globally stable on S, then $(x_k) = (T^k x_0)$ converges to x^*

$$\text{hence output} \approx x^*$$

The algorithm just described is called **successive approximation**

```python
import numpy as np
def successive_approx(T,                     # Operator (callable)
                      x_0,                   # Initial condition
                      tolerance=1e-6,        # Error tolerance
                      max_iter=10_000,       # Max iteration bound
                      print_step=25,         # Print at multiples
                      verbose=False):
    """
    Implements successive approximation by iterating on the operator T.

    """
    x = x_0
    error = np.inf
    k = 1
    while error > tolerance and k <= max_iter:
        x_new = T(x)
        error = np.max(np.abs(x_new - x))
        if verbose and k % print_step == 0:
            print(f"Completed iteration {k} with error {error}.")
        x = x_new
        k += 1
    if error > tolerance:
        print(f"Warning: Iteration hit upper bound {max_iter}.")
    elif verbose:
        print(f"Terminated successfully in {k} iterations.")
    return x
```

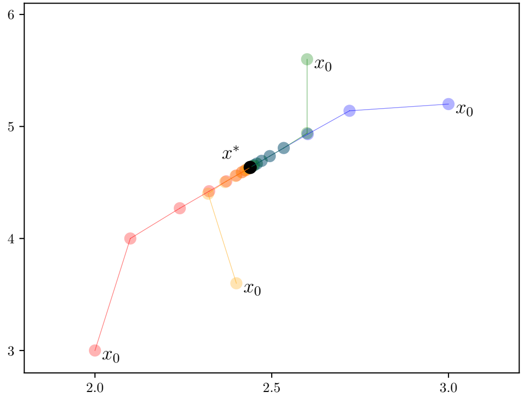


Figure: Successive approximation from different initial conditions

Newton's Method

Let h be a differentiable real-valued function on $(a, b) \subset \mathbb{R}$

We seek a **root** of h, which is an x^* such that $h(x^*) = 0$

We start with guess x_0 and then update it

To do this we use $h(x_1) \approx h(x_0) + h'(x_0)(x_1 - x_0)$

Setting the RHS $= 0$ and solving for x_1 gives

$$x_1 = x_0 - \frac{h(x_0)}{h'(x_0)}$$

Continuing in the same way, we set

$$x_{k+1} = q(x_k) \quad \text{where} \quad q(x) := x - \frac{h(x)}{h'(x)},$$

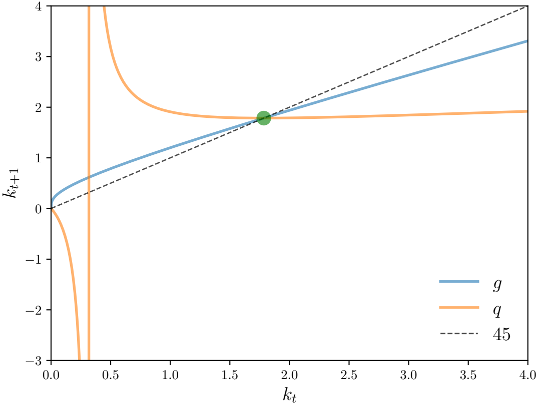


Figure: Successive approximation vs Newton's method

Comments:

- The map q is flat close to the fixed point k^*
- Hence Newton's method converges quickly <u>near</u> k^*
- But Newton's method is not globally convergent
- Successive approximation is slower but more robust

Key ideas

- There is almost always a trade-off between robustness and speed
- Speed requires assumptions, and assumptions can fail

Newton's method extends naturally to **multiple dimensions**

When h is a map from $S \subset \mathbb{R}^n$ to itself, we use

$$x_{k+1} = x_k - [J(x_k)]^{-1} h(x_k)$$

Here $J_h(x_k) :=$ the Jacobian of h evaluated at x_k

Comments

- Typically faster but less robust
- Matrix operations can be parallelized
- Automatic differentiation can be helpful

Job Search

A model of job search created by **John J. McCall**

We model the decision problem of an unemployed worker

Job search depends on

- current and likely future wage offers
- impatience, and
- the availability of unemployment compensation

We begin with a very simple version of the McCall model

(Later we consider extensions)

Set Up

An agent begins working life at time $t = 0$ without employment

Receives a new job offer paying wage w_t at each date t

She has two choices:

1. **accept** the offer and work permanently at w_t or
2. **reject** the offer, receive unemployment compensation c, and reconsider next period

Assume $\{w_t\}$ is $\overset{\text{IID}}{\sim}$ φ, where

- $\mathsf{W} \subset \mathbb{R}_+$ is a finite set of wage outcomes and
- $\varphi \in \mathcal{D}(\mathsf{W})$

The agent cares about the future but is **impatient**

Impatience is parameterized by a **time discount factor** $\beta \in (0, 1)$

- Present value of a next-period payoff of y dollars is βy

Trade off:

- $\beta < 1$ indicating some impatience
- hence the agent will be tempted to accept reasonable offers, rather than always waiting for a better one
- The key question is how long to wait

The worker who aims to maximize

$$\mathbb{E} \sum_{t=0}^{\infty} \beta^t Y_t, \quad Y_t \in \{c, W_t\} \text{ is earnings at time } t \tag{1}$$

- $\{W_t\} \stackrel{\text{IID}}{\sim} \varphi$ for $\varphi \in \mathcal{D}(\mathsf{W})$
- $\mathsf{W} \subset \mathbb{R}_+$ with $|\mathsf{W}| < \infty$
- c and β are positive and $\beta < 1$
- jobs are <u>permanent</u>

What is max EPV of each option when lifetime is infinite?

What if we **accept** $w \in \mathsf{W}$ now?

$$\text{EPV} \;= \text{stopping value} \;= w + \beta w + \beta^2 w + \cdots = \frac{w}{1-\beta}$$

What if we **reject**?

$$\begin{aligned} \text{EPV} \;&= \text{continuation value} \\ &= \text{EPV of } \underline{\text{optimal}} \text{ choice in each subsequent period} \end{aligned}$$

But what are optimal choices?!

Calculating optimal choice requires knowing optimal choice!

The Value Function

Let $v^*(w) :=$ max lifetime EPV given wage offer w

We call v^* the **value function**

Suppose that we know v^*

Then the (maximum) **continuation value** is

$$h^* := c + \beta \sum_{w' \in \mathsf{W}} v^*(w') \varphi(w')$$

$$= \text{ max EPV conditional on decision to continue}$$

The optimal choice is then

$$\mathbb{1}\left\{\text{stopping value} \geqslant \text{continuation value}\right\} = \mathbb{1}\left\{\frac{w}{1-\beta}, \, h^*\right\}$$

But how can we calculate v^*?

Key idea: We can use the Bellman equation to solve for v^*

Theorem. The value function v^* satisfies the **Bellman equation**

$$v^*(w) = \max\left\{\frac{w}{1-\beta}, c + \beta \sum_{w' \in \mathsf{W}} v^*(w')\varphi(w')\right\} \qquad (w \in \mathsf{W})$$

Intuition:

- If accept, get $w/(1-\beta)$
- If reject and then choose optimally, get max continuation value
- Max value today is max of these alternatives

So how can we use the Bellman equation

$$v^*(w) = \max\left\{\frac{w}{1-\beta}, \, c + \beta \sum_{w' \in \mathsf{W}} v^*(w')\varphi(w')\right\} \qquad (w \in \mathsf{W})$$

to solve for v^*?

Contractions

Let

- U be a nonempty subset of $\mathbb{R}^n$,
- $\|\cdot\|$ be a norm on $\mathbb{R}^n$, and
- T be a self-map on U

T is called a **contraction** on U with respect to $\|\cdot\|$ if

$$\exists\, \lambda < 1 \text{ such that } \|Tu - Tv\| \leqslant \lambda \|u - v\| \quad \text{for all} \quad u, v \in U$$

Example. $Tx = ax + b$ is a contraction on $\mathbb{R}$ with respect to $|\cdot|$ if and only if $|a| < 1$

Indeed,

$$|Tx - Ty| = |ax + b - ay - b| = |a||x - y|$$

Banach's Contraction Mapping Theorem

Theorem If

1. U is closed in $\mathbb{R}^n$ and
2. T is a contraction of modulus λ on U with respect to some norm $\|\cdot\|$ on $\mathbb{R}^n$,

then T has a unique fixed point u^* in U and

$$\|T^n u - u^*\| \leqslant \lambda^n \|u - u^*\| \quad \text{for all } n \in \mathbb{N} \text{ and } u \in U$$

In particular, T is globally stable on U

Proof: See the DP text

Let's now return to the job search problem

Recall that that the value function v^* solves the Bellman equation

That is,

$$v^*(w) = \max\left\{\frac{w}{1-\beta}, c + \beta \sum_{w' \in \mathsf{W}} v^*(w')\varphi(w')\right\} \qquad (w \in \mathsf{W})$$

The infinite-horizon **continuation value** is defined as

$$h^* := c + \beta \sum_{w'} v^*(w')\varphi(w')$$

Key question: how to solve for v^*?

We introduce the **Bellman operator**, defined at $v \in \mathbb{R}^{\mathsf{W}}$ by

$$(Tv)(w) = \max\left\{\frac{w}{1-\beta}, c + \beta \sum_{w' \in \mathsf{W}} v(w')\varphi(w')\right\} \qquad (w \in \mathsf{W})$$

By construction, $Tv = v \iff v$ solves the Bellman equation

Let $\mathcal{V} := \mathbb{R}^{\mathsf{W}}_{+}$

Proposition. T is a contraction $\mathcal{V}$ with respect to $\|\cdot\|_\infty$

In the proof, we use the elementary bound

$$|\alpha \vee x - \alpha \vee y| \leqslant |x - y| \qquad (\alpha, x, y \in \mathbb{R})$$

Fixing f, g in $\mathcal{V}$ fix any $w \in \mathsf{W}$, we have

$$|(Tf)(w) - (Tg)(w)| \leqslant \left| \beta \sum_{w'} f(w')\varphi(w') - \beta \sum_{w'} g(w')\varphi(w') \right|$$

$$= \beta \left| \sum_{w'} [f(w') - g(w')]\varphi(w') \right|$$

Applying the triangle inequality,

$$|(Tf)(w) - (Tg)(w)| \leqslant \beta \sum_{w'} |f(w') - g(w')|\varphi(w') \leqslant \beta \|f - g\|_\infty$$

$$\therefore \quad \|Tf - Tg\|_\infty \leqslant \beta \|f - g\|_\infty$$

Recall: The optimal decision at any given time, facing current wage draw $w \in \mathsf{W}$, is

$$\mathbb{1}\left\{\frac{w}{1-\beta} \geqslant h^*\right\}$$

Let's try to write this in the language of dynamic programming

Dynamic programming centers around the problem of finding optimal **policies**

Optimal Policies

In general, for a dynamic program, choices consist of a sequence $(A_t)_{t \geqslant 0}$

- specifies how the agent acts at each t

Since agents are not clairvoyant, so we assume that A_t cannot depend on future events

In other words, for some function σ_t,

$$A_t = \sigma_t(X_t, A_{t-1}, X_{t-1}, A_{t-2}, X_{t-2}, \ldots A_0, X_0)$$

In dynamic programming, σ_t is called a **policy function**

Key idea Design the state such that X_t is

- sufficient to determine the optimal current action
- but not so large as to be unmanagable

- Finding the state is an art!

Example. Recall retailer who chooses stock orders and prices in each period

What to include in the current state?

- level of current inventories
- interest rates and inflation?
- the rate at which inventories have changed?
- competitors prices?

So suppose state X_t determines the current action A_t

Then we can write $A_t = \sigma(X_t)$ for some function σ

Note that we dropped the time subscript on σ

No loss of generality: can include time in the current state

- i.e., expand X_t to $\hat{X}_t = (t, X_t)$

Depends on the problem at hand

- For the job search model with finite horizon, the date matters
- For the infinite horizon version of the problem, however, the agent always looks forward toward an infinite horizon

For job search model,

- state $=$ current wage offer and
- possible actions are accept (1) or reject (0)

A policy is a map σ from W to $\{0, 1\}$

Let Σ be the set of all such maps

For each $v \in \mathcal{V}$, let us define a **v-greedy policy** to be a $\sigma \in \Sigma$ satisfying

$$\sigma(w) = \mathbb{1}\left\{ \frac{w}{1-\beta} \geqslant c + \beta \sum_{w' \in \mathsf{W}} v(w')\varphi(w') \right\} \quad \text{for all } w \in \mathsf{W}$$

Accepts iff $w/(1-\beta) \geqslant$ continuation value computed using v

Optimal choice:

- agent should adopt a v^*-greedy policy
- Sometimes called **Bellman's principle of optimality**

We can also express a v^*-greedy policy via

$$\sigma^*(w) = \mathbb{1}\{w \geqslant w^*\} \quad \text{where} \quad w^* := (1-\beta)h^* \tag{2}$$

The term w^* in (2) is called the **reservation wage**

- Same ideas as before, different language
- We prove optimality more carefully later

Computation

Since T is globally stable on $\mathcal{V}$, we can compute an approximate optimal policy by

1. applying successive approximation on T to compute v^*
2. calculate a v^*-greedy policy

In dynamic programming, this approach is called **value function iteration**

input $v_0 \in \mathcal{V}$, an initial guess of v^*
input τ, a tolerance level for error
$\varepsilon \leftarrow \tau + 1$
$k \leftarrow 0$
while $\varepsilon > \tau$ **do**
    for $w \in \mathsf{W}$ **do**
        $v_{k+1}(w) \leftarrow (Tv_k)(w)$
    end
    $\varepsilon \leftarrow \|v_k - v_{k+1}\|_\infty$
    $k \leftarrow k + 1$
end
Compute a v_k-greedy policy σ
return σ

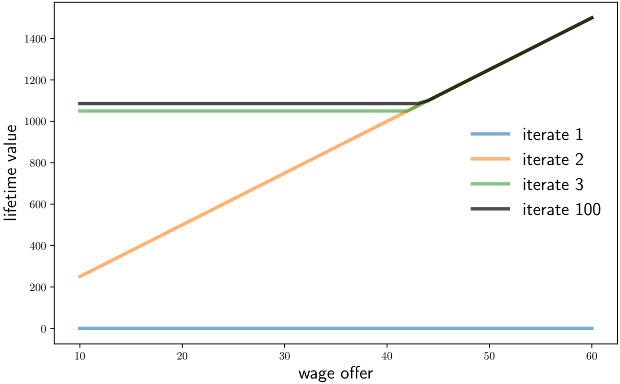


Figure: A sequence of iterates of the Bellman operator

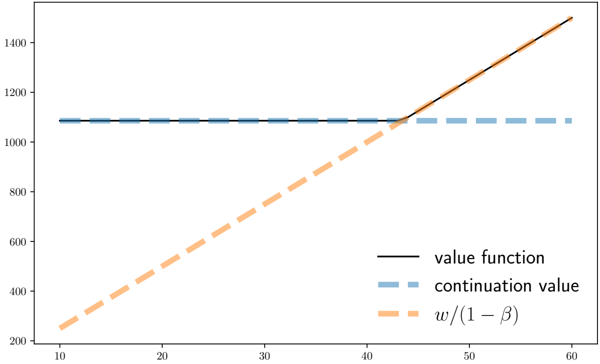


Figure: The approximate value function for job search

Computing the Continuation Value Directly

We used a standard dynamic programming approach to solve this problem

Sometimes we can find more efficient ways to solve particular problems

For the infinite horizon job search problem, a more efficient way exists

The idea is to compute the continuation value directly

This shifts the problem from n-dimensional to one-dimensional

Method: Recall that

$$v^*(w) = \max\left\{\frac{w}{1-\beta},\, c + \beta \sum_{w'} v^*(w')\varphi(w')\right\} \qquad (w \in \mathsf{W})$$

Using the definition of h^*, we can write

$$v^*(w') = \max\left\{w'/(1-\beta),\, h^*\right\} \qquad (w' \in \mathsf{W})$$

Take expectations, multiply by β and add c to obtain

$$h^* = c + \beta \sum_{w'} \max\left\{\frac{w'}{1-\beta},\, h^*\right\}\varphi(w')$$

How to find h^* from the equation

$$h^* = c + \beta \sum_{w'} \max \left\{ \frac{w'}{1-\beta}, h^* \right\} \varphi(w') \tag{3}$$

We introduce the map $g \colon \mathbb{R}_+ \to \mathbb{R}_+$ defined by

$$g(h) = c + \beta \sum_{w'} \max \left\{ \frac{w'}{1-\beta}, h \right\} \varphi(w')$$

By construction, h^* solves (3) if and only if h^* is a fixed point of g

Ex. Show that g is a contraction map on $\mathbb{R}_+$

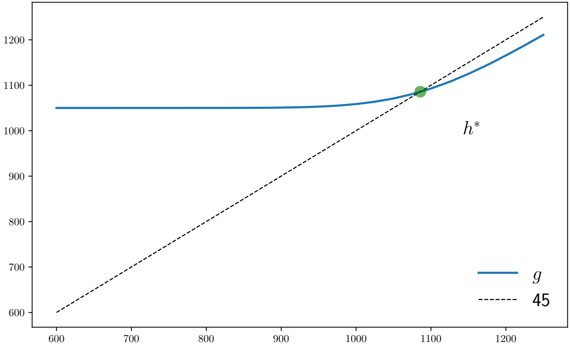


Figure: Computing the continuation value as the fixed point of g

New algorithm:

1. Compute h^* via successive approximation on g

    - Iteration in $\mathbb{R}$, not $\mathbb{R}^n$

2. Optimal policy is

$$\sigma^*(w) = \mathbb{1}\left\{\frac{w}{1-\beta} \geqslant h^*\right\}$$