ABM and environmental policy

A mini-Poseidon model in NetLogo

Nicolas Payette

September 19, 2023

What's the plan?

- A little bit about the **Poseidon** fisheries ABM
- A whole lot about building **Poseidon in NetLogo**!

Why model fisheries?

- It's big!
  - 96.4 million tonnes of fish caught in 2018
  - employing 59.51 million people
  - 17% of total animal protein consumed globally
- It's in trouble!
  - 34.2% of stocks are overfished

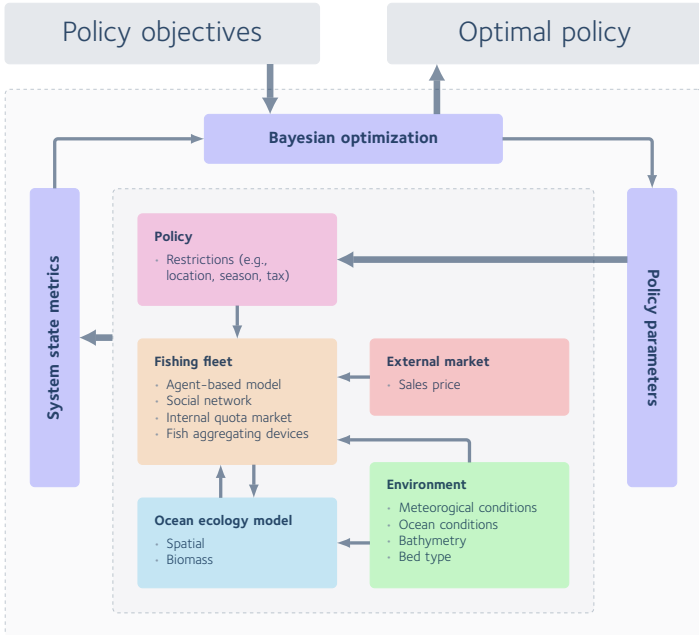
Why use ABM to do it?

- They're inherently spatial;
- they involve complex interactions,
- between smart, adaptive agents.

Source: `http://www.fao.org/3/ca9231en/CA9231EN.pdf`

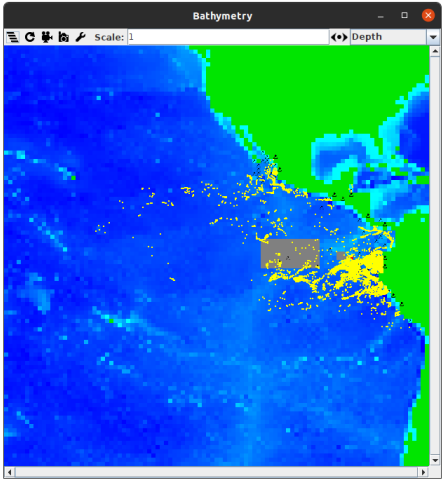
Poseidon

Source: Ricardo André Frantz, `https://w.wiki/wzx`



Poseidon applications

- Conceptual models
- US West Coast Groundfish
- Indonesian Deep water snapper grouper fishery
- Eastern Pacific Tuna Management



What can we put in a minimal fisheries model?

Agents:

- A port,
- fishers,
- and fish!

Source: https://www.tourismeilesdelamadeleine.com/fr/decouvrir-les-iles/les-iles/ile-de-grande-entree

Some policy to test:

- The size of a marine protected area (MPA)

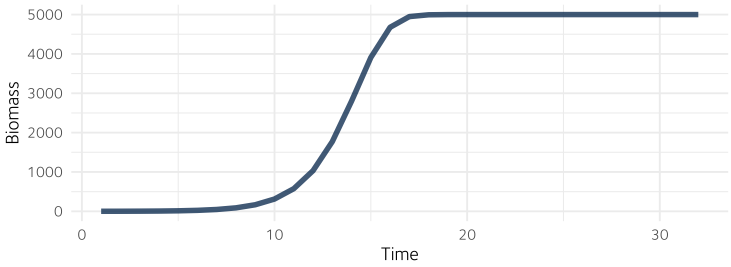
Fisher agents in Poseidon use the **“Explore, exploit, imitate”** behaviour algorithm:

- Should I go exploring?
  - If yes, pick a random spot not too far from my favourite spot (**EXPLORE**)
  - If not, pick a friend and ask: is my favourite spot better than my friend’s favourite spot?
    - If it is, go to my favourite spot (**EXPLOIT**)
    - If it isn’t, go to my friend’s favourite spot (**IMITATE**)
- If the spot I went to was better than my favourite spot, it becomes my new favourite!

A bit of biology

We do not simulate individual fish. We keep track of the total biomass and apply yearly **logistic growth**.

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$



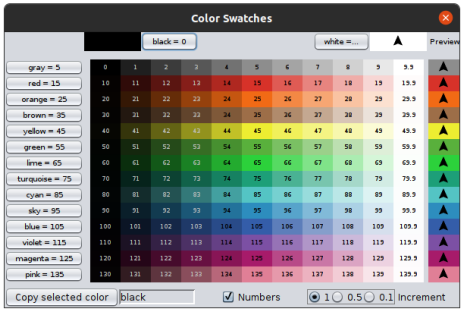
Pierre-François Verhulst (1804–1849).
Source: `https://w.wiki/x2o`.

Get the model skeleton from GitHub

https://github.com/
nicolaspayette/mini-poseidon

Recoloring patches

```
to recolor-patches
  ask patches [
    set pcolor scale-color blue (biomass / 2) carrying-capacity 0
  ]
end
```



Explore, exploit, imitate...

```
to pick-destination ; fisher procedure
  ifelse random-float 1 < exploration-probability [
    ; explore:
    let r 1 + random-poisson exploration-radius
    set trip-destination [ one-of fishable-patches in-radius r ] of favourite-destination
  ] [
    let other-fisher one-of other fishers
    let their-profits [ profits-at-favourite-destination ] of other-fisher
    ifelse profits-at-favourite-destination >= their-profits [
      ; exploit:
      set trip-destination favourite-destination
    ] [
      ; imitate
      set trip-destination [ favourite-destination ] of other-fisher
    ]
  ]
  set current-destination trip-destination
end
```

Let's go!

```
to go
  ask fishers [
    set trip-costs trip-costs + hourly-costs
    ifelse patch-here = current-destination [
      ifelse any? ports-here [ dock ] [ fish ]
    ] [
      face current-destination
      forward speed
    ]
  ]
  update-biology
  tick
end
```

Fishing!

```
to fish ; fisher procedure
  set pcolor red
  let biomass-caught biomass * catchability
  set biomass biomass - biomass-caught
  set biomass-in-hold biomass-in-hold + biomass-caught
  set current-destination [ patch-here ] of one-of ports
end
```

Docking at the port

```
to dock ; fisher procedure
  let revenues biomass-in-hold * price-of-fish
  set biomass-in-hold 0
  let profits revenues - trip-costs
  set trip-costs 0
  set bank-balance bank-balance + profits
  (ifelse
    trip-destination = favourite-destination [
      set profits-at-favourite-destination profits
    ]
    profits > profits-at-favourite-destination [
      set favourite-destination trip-destination
      set profits-at-favourite-destination profits
    ]
  )
  pick-destination
end
```

Updating the biology

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$

```
to update-biology
  diffuse biomass diffusion-rate
  recolor-patches
  if ticks mod 365 = 0 [
    ask patches [
      set biomass biomass + (
        growth-rate * biomass * (1 - (biomass / carrying-capacity))
      )
    ]
  ]
end
```

Adding a protected area after one year

```
to setup-mpa
  set-default-shape xs "x"
  ask patches with [
    pxcor >= min-mpa-x and
    pxcor <= max-mpa-x and
    pycor >= min-mpa-y and
    pycor <= max-mpa-y and
    not any? ports-here
  ] [
    sprout-xs 1 [ set color cyan ]
  ]
  set fishable-patches fishable-patches with [ not any? xs-here ]
end
```

Modify go

```
to go
  if ticks = 365 * 24 [ setup-mpa ]
  ask fishers [
    set trip-costs trip-costs + hourly-costs
    ifelse patch-here = current-destination [
      ifelse any? ports-here [ dock ] [ fish ]
    ] [
      face current-destination
      forward speed
    ]
  ]
  update-biology
  tick
end
```

Modify dock

```
to dock ; fisher procedure
  let revenues biomass-in-hold * price-of-fish
  set biomass-in-hold 0
  let profits revenues - trip-costs
  set trip-costs 0
  set bank-balance bank-balance + profits
  (ifelse
    trip-destination = favourite-destination [
      set profits-at-favourite-destination profits
    ]
    profits > profits-at-favourite-destination [
      set favourite-destination trip-destination
      set profits-at-favourite-destination profits
    ]
  )
  if [ any? xs-here ] of favourite-destination [
    set favourite-destination min-one-of fishable-patches [
      distance [ favourite-destination ] of myself
    ]
  ]
  pick-destination
end
```

Scenario

We have observed the real-world fishery for one year.

Fishers ended up with a mean bank balance of £775,000.

We want to:

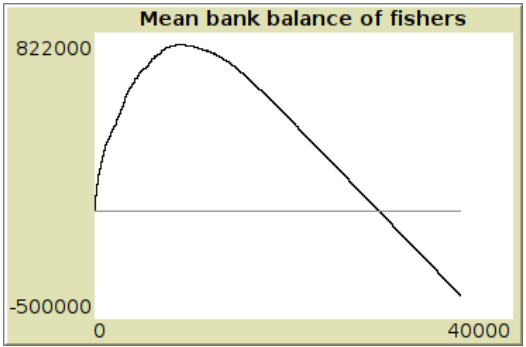
- Use this information to estimate the EEI algorithm parameters.
- Simulate the fishery for three more years under the "business as usual" scenario.
- Figure out the best location for an MPA.

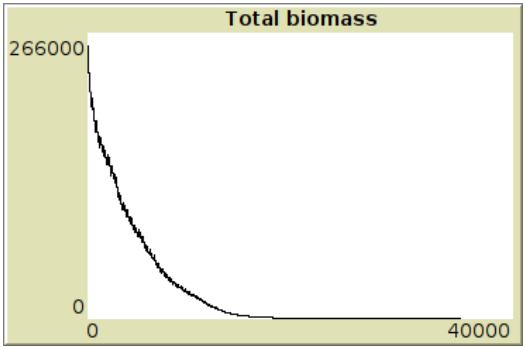
Estimating EEI parameters

Loading BehaviorSearch results automatically

```
to load-best [ prefix ]
  let rows csv:from-file word prefix ".finalCheckedBests.csv"
  let headers item 0 rows
  let data first sort-by [ [r1 r2] -> last r1 > last r2 ] but-first rows
  foreach range length headers [ i ->
    if last item i headers = "*" [
      run (word "set " but-last item i headers " " item i data)
    ]
  ]
end
```

Not looking good for the fishery...





Searching for the optimal MPA