CMiE Final Project

May 12, 2018

0.1 Final Project

Catherine Ledna

4-11-18

0.1.1 Overview

This project recreates the results of Gutrich & Howarth (2007), solving for optimal stand rotation of privately owned forests under private and social welfare functions that take into account the social value of carbon sequestration. The equations used in this paper are described below, and full documentation may be found at:

Gutrich, J. & Howarth, RB. 2007. Carbon sequestration and the optimal management of New Hampshire timber stands. *Ecological Economics* 62(3-4): 441-450.

The Github repository for this project may be found here: https://github.com/cledna/cmie-final.

Parameters for each equation are provided in Table 1 (gutrich_howarth_table1.csv). The model seeks to optimize NPV by choosing stand age, s, the number of years since a clear-cut harvest. NPV can be decomposed as:

$NPV_{social} = NPV_{timber} + NPV_{carbon}$.

NPV_{carbon} can be expressed as the product of the marginal benefit of carbon storage per year, $MB(t)$, the net uptake of carbon per year, $\Delta C(t)$, and the discount rate, expressed by $r(t)$. The specifics are denoted below:

$NPV_{carbon} = \sum_{t=0}^{\infty} MB(t)\Delta C(t) \prod_{i=1}^{t} 1/(1 + r(t))$

$MB(t)$ may be chosen depending on different expectations of climate change; they use the equation:

$MB_{low}(t) = 99.69 - (99.69 - 24.63)0.989^t$

$\Delta C(t)$ is given by:

$\Delta C(t) = C(t+1) - C(t)$ where $C(t)$ will be described in more detail later,
and $r(t) follows:

$r(t) = 0.0268 + (0.0434 - 0.0268)0.992^t$.

NPV_{timber} represents the forest owner's attempts to maximize the revenue stream of timber stands by sequential timber harvests, and can be given by a discounted function of price and volume:

$NPV_{timber} = \sum_{i=1}^{\infty} P(s_h, s_h * i)V(S_h) \prod_{t=1}^{S_h*i} 1/(1 + r(t))$

where s_h denotes the stand age at harvest, i denotes the harvest and $t = s_h * i$ denotes the date at which the ith harvest occurs.

Price is decomposed in to the prices of pole timber and saw timber, which are extracted from volume at harvest based on equations derived from the US Forest service. These equations, as well as the equations governing volume and carbon sequestration will be described in more detail in the code of this model and may also be found in the text of the paper.

The approach used for this project involves the use of Optim to recreate the optimization model, which the authors originally coded in Excel. In the original paper, the authors used a grid search strategy to determine private and social s_h in one-year increments ranging from 15 to 500 years. These results are described in Table 2 (gutrich_howarth_table2.csv) and are compared with results using the Optim package. The problem being explored is a deterministic optimization problem, but extensions might be to add stochastic elements.

0.1.2 Code

As a preliminary effort, I evaluate the reproducibility of Gutrich & Howarth's results for private timber owners, NPV_timber.

In [146]:
```
using Optim

# Parameter Values
# To limit the scope of this project, only parameters for white-red jack pine will b
# Parameters and definitions may be found in gutrich_howarth_table1.csv of this repo
# A more elegant solution would be to read in and assign these values from a
#  data frame or dictionary; due to time constraints they're being hand-coded.

const 0 = 431.
const 1 = .0066
const 2 = 4.47
const Ppole = 19.8
const Psaw = 121.
const 0 = 6.37
const 1 = 2.7
const 2 = 5.4
const 0 = 155.
```

```
const 1 = .0115
const Cdead0 = 20.5
const 0 = .045
const 1 = .258
const 2 = .358
const Csoil = 196.
const  = .268
const 1 = .234
const 2 = .234
const 3 = .357
const 4 = .357
const h1 = .208
const h2 = .5
const h3 = .114
const h4 = .149
const 01 = .006
const 02 = .0038
const 03 = .0062
const 04 = .0042
const 11 = .237
const 12 = .298
const 13 = .227
const 14 = .187

# Bounds on stand age at harvest
x_bounds = (15, 500)

# Time horizon (arbitrarily chosen, was not specified in paper)
const T = 500
```

```
WARNING: redefining constant T
```

Out[146]: 500

In [147]:

```
# Private Forest Owner Optimization
# The objective function to be optimized is NPV_timber

# For the private forest owner
# s = stand age in years
# t = time (1:1000), years
# The choice variable is stand age at harvest, x

# Define Functions
# Volume function
V = s -> begin
    if s >= 2
        return 0 * (1 - (1 - 1)^(s - 2))
```

```
    else
        return 0.
    end

end

# Proportion of saw timber as function of total timber volume
fsaw = s -> begin
    if (0*s / (s+1) - 2) < 0
        return 0
    elseif (0*s / (s + 1) - 2) > 1
        return 1
    else
        return (0*s / (s + 1) - 2)
    end

end

# Price Function at harvest
Price(x,t) = (Ppole * (1 - fsaw(x)) + Psaw*fsaw(x)) * 1.01^t

# Discount rate
r = t -> 0.0268 + (0.0434 - 0.0268) * 0.992^t

# Timber Payoff Function
NPV_timber = x -> begin
    # Compute number of harvests for stand age s given time period T
    # Assumes age of stand height at t=1 is 0

    nharvest = round(T / x, 0)
    val = 0.
    for i in 1:nharvest
        dr = 1.
        for t in 1:x*i
            dr *= 1/(1+r(t))
        end
        val += Price(x, x*i) * V(x) * dr
    end
    return val

end

# Optimization for private timber benefits
xout = optimize(x-> -NPV_timber(x),15, 500)
x_opt = Optim.minimizer(xout)[1]

print("Optimal rotation period under private ownership: $(round(x_opt))")
```

```
Optimal rotation period under private ownership: 37.0
```

This matches the result presented in Gutrich & Howarth (2007). (gutrich_howarth_table2.csv)

0.1.3 Carbon Sequestration

I next extend the model to include net social benefits from carbon storage. I had to make some assumptions here because some things were not stated in the paper.

```
In [148]: # Carbon Storage Functions
          # Carbon stored in living wood
          C_live = t -> 0 * (1 - (1 - 1)^t)

          # Decomposition
          # Define harvest function for harvest years (if t in x*i for i=1:nharvests)
          # Assumes harvest occurred in year 0
          H =(t,s) -> begin
              nharvest = round(T / s, 0)
              Ts = vcat(0,[s*i for i=1:nharvest])
              if t in Ts
                  return V(s)
              else
                  return 0
              end

          end

          # Decomposition function for years when harvest occurs
          D= (t,s) -> begin
              if H(t,s) == 0
                  return 0
              else
                  return C_live(t) * *H(t,s)
              end
          end


          # Other functions are dependent on their prior-year values
          # Carbon stored in dead wood
          Cdead = (t,s) -> begin
              if t==0
                  return Cdead0
              elseif t==1
                  return (1 - 0)*(Cdead0 + 1 * C_live(t-1)^2 + D(t-1,s))
              else
                  return (1 - 0)*(Cdead(t-1,s) + 1 * C_live(t-1)^2 + D(t-1,s))
              end
          end
```

```
# Carbon stored in 4 categories of wood products
# I am making the assumption that carbon storage in period 0 is harvested volume*fra
# This is not stated in the paper
Cprod1 = (t,s) -> begin
    if t==0
        return 1*h1*H(0,s)
    elseif t==1
        return (1 - 01) * (H(0,s) + 11*1*h1*H(0,s))
    else
        return (1 - 01) * (Cprod1(t-1,s) + 11*1*h1*H(t-1,s))
    end

end

Cprod2 = (t,s) -> begin
    if t==0
        return 2*h2*H(0,s)
    elseif t==1
        return (1 - 02) * (H(0,s) + 12*2*h2*H(0,s))
    else
        return (1 - 02) * (Cprod2(t-1,s) + 12*2*h2*H(t-1,s))
    end

end

Cprod3 = (t,s) -> begin
    if t==0
        return 3*h3*H(0,s)
    elseif t==1
        return (1 - 03) * (H(0,s) + 13*3*h3*H(0,s))
    else
        return (1 - 03) * (Cprod3(t-1,s) + 13*3*h3*H(t-1,s))
    end

end

Cprod4 = (t,s) -> begin
    if t==0
        return 4*h4*H(0,s)
    elseif t==1
        return (1 - 04) * (H(0,s) + 14*4*h4*H(0,s))
    else
        return (1 - 04) * (Cprod4(t-1,s) + 14*4*h4*H(t-1,s))
    end

end
```

```
# C function
C = (t, s) -> begin
    return C_live(t) + Cdead(t,s) + Cprod1(t,s) + Cprod2(t,s) + Cprod3(t,s) + Cprod4

end

# DeltaC function
deltaC = (t,s) -> begin
    if t+1 >T
        return 0.
    else
        return C(t+1,s) - C(t,s)
    end

end

# Marginal benefit function
MB = t -> 99.69 - (99.69 - 24.63)*0.989^t

NPV_timber = x -> begin
    # Compute number of harvests for stand age x given time period T
    # Assumes age of stand height at t=1 is 0

    nharvest = round(T / x, 0)
    val = 0.
    for i in 1:nharvest
        dr = 1.
        for t in 1:x*i
            dr *= 1/(1+r(t))
        end
        val += Price(x, x*i) * V(x) * dr
    end
    return val

end
```

Out[148]: (::#677) (generic function with 1 method)

In [152]:
```
# NPV Carbon
NPV_carbon = x -> begin

    val = 0.
    for t=0:T

        dr = 1.
        for i = 1:t
          dr *= 1/(1+r(i))
```

```julia
            end

            val += MB(t) * deltaC(t,x)* dr


        end
        return val
    end
```

Out[152]: (::#689) (generic function with 1 method)

In [154]:
```julia
# NPV Total
NPV_total = x -> NPV_carbon(x) + NPV_timber(x)

xout3 = optimize(x -> -NPV_total(x), 50, 55)
x_opt3 = Optim.minimizer(xout3)

print("Optimal rotation period under combined benefit: $(round(x_opt3))")
```

```
Optimal rotation period under combined benefit: 55.0
```

Unfortunately, I was unable to verify the results of Gutrich and Howarth (2007), due to the excessively long optimization time caused by the amount of recursivity in the carbon storage functions that I defined. I attempted to mitigate this somewhat by presenting a very restricted version of this optimization in the section above. This project would benefit from additional coding time to think of a way around the recursive element, or a way to restructure the optimization.

These results do not recreate what is presented in the paper. This may be because of differences in carbon storage function zero period values, other issues with time-steps, or as-yet unidentified errors. In addition, this approach suffers from performance issues due to the recursive functions defined for carbon storage functions.

Possible extensions of this project would be to explore adding stochastic shocks, Monte Carlo analysis, and techniques to create a more generic algorithm to solve this model.