University of Stavanger

**Faculty of Science and Technology**

# MASTER'S THESIS

Study program/ Specialization:

Industriell økonomi, investering og finans

Spring semester, 20..16..

[✓ Open]/ Restricted access

Writer:

Karl Stefan Afradi

Karl Stefan Afradi

(Writer's signature)

Faculty supervisor: Frank Asche

External supervisor(s):

Thesis title:

Salmon in a Global Perspective, a Lesson From Norwegian Regulations

Credits (ECTS): 30

Key words:
Salmon
Aquaculture
Incentives
Salmon lice
Regulations

Pages: ..108............

+ enclosure: …………

Stavanger, ..15/12-16.....
Date/year

Front page for master thesis
Faculty of Science and Technology

# Salmon Aquaculture in a Global Perspective, a Lesson from Norwegian Regulations

By

Karl Stefan Afradi

**Abstract:** The salmon aquaculture is experiencing rapid growth as the worlds need for sustainable sources of protein and fat increases. The Norwegian aquaculture industry is world leading. Over the years several regulatory schemes have been employed to internalize the industry's negative environmental. This thesis investigates how the industry will adapt to the suggestion of a collaborative incentive scheme. Results show that risk-averse actors will prefer to defect from collaboration, and that a single industry actor will have additionally decreased incentive to collaborate as a function of his relative smaller market size. Furthermore, incentive schemes like this seem only applicable to a coherent group of actors, and the probability of a single actor defecting can lead to opportunistic free rider behaviour in the group.

# TABLE OF CONTENTS

# 1 BACKGROUND

The natural life cycle of fish has developed over millions of years, and exists in a homeostasis within its natural environment. Aquaculture production is a man-made production process that imitates the natural life cycle of fish. The process is designed to be efficient as to extract as much value as possible. Several parts of the process are still carried out in the wild, and aquaculture it is therefore inherently susceptible to influence the environment.

As demand for farmed fish has increased, the industry has gradually employed more and more intense production techniques to combat and improve upon the conditions that limit their production volumes. The presence of several global and local environmental concerns can be traced back to fish farmers aggressively trying to increase production (Asche and Bjørndal, 2011). These efforts are usually attempts to change conditions and control parameters that have been proven to work after millions of years in trial and error by nature.

A leading global concern is that the growing aquaculture industry will threaten the sustainably of capture fisheries. This is because a sizeable amount of the input factors used in production of the feed utilized in aquaculture stems from wild fish stock. This is known as the 'fish meal trap' (Naylor et al, 2000). In short, the fish meal trap is what happens if increased production in aquaculture in turn puts increased pressure on the wild species used for fish meal and fish oil. This poses a problem when the authorities fail to implement the regulations needed to ensure the sustainability of wild fish stock, and when there is no viable substitute that can help to decrease the total demand for fish meal (Asche and Bjørndal, 2011).

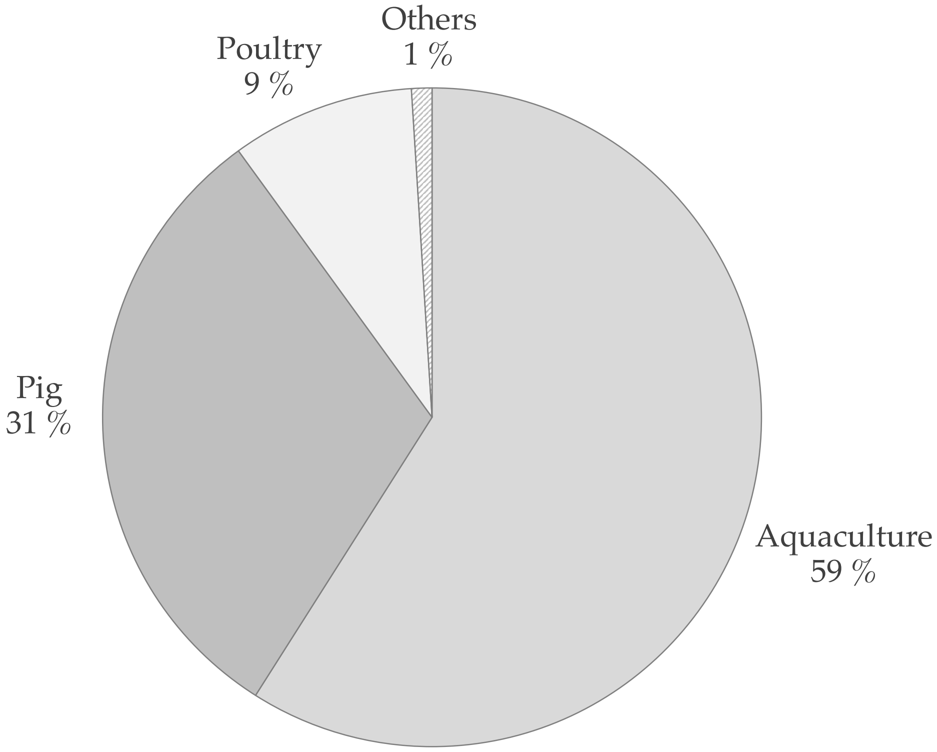


**Figure 1.1 Estimated total use of fish meal, 2008 (Asche and Bjørndal, 2011).**

Aquaculture and salmon farming can also pose a threat to local ecosystems in several ways. Issues can arise from pollution in various stages of the production process, as well as from interactions between farmed fish and the local environments.

Given a typical stakeholder regime often found in business, there are two main incentives for aligning operations with environmental concerns: it either impacts productivity and cash flow, and shows up on the bottom line directly; and/or there are regulations enforced by the authorities that internalize negative effect not accounted for in market prices (Asche and Bjørndal, 2011).

All industry externalities, including environmental issues, can be remedied by a combination of appropriate incentives and sufficient control of the production process. Salmon aquaculture is therefore theoretically sustainable. However, there is no guarantee that the required incentives will develop naturally from the market, and the importance of government involvement is therefore paramount to ensure sustainability.

The Norwegian government recognizes the importance of having a governing body act on behalf of the environment, and the nation's environmental concerns have to a large extent already been solved by the country's salmon aquaculture industry. This is due in no small part to how many of the industry's environmental externalities that also drive production costs (Ache and Bjørndal, 2011).

It is obviously important and desirable that any regulation address industry externalities in an efficient manner, while also allowing for economic sustainability and growth if possible. A recent proposal put forth by the Norwegian government is a revision of the current regulatory regime (Norwegian Ministry of Trade, Industry and Fisheries, 2015). By imposing a new collective accountability scheme with the intention to reward fish farmers for reducing their environmental impact, the government introduces a new mechanic that incentivises collaboration as an instrument for managing externalities inflicted onto local ecosystems.

While the upside offered by this incentive scheme is relatively straight forward, the possibility of maladaptation presents subtle economic and environmental consequences on the downside. The purpose of this thesis is to investigate the implications and dynamics of the proposed policy. This enquiry is approached by examining how the industry will choose to adapt to the proposition. To allow for a systematic examination the thesis is structured in the following way:

Section 2 provides an in depth look at the rationale and suggested policies as put forth in Meld St. 16. In section 3 a generic model is developed to mathematically portray the proposed incentive scheme. Section 4 describes a bio-economic model that allows for a quantitative analysis of the generic model. Using this framework, a simulation model that facilitates numeric examination of key parameters is developed and calibrated in section 5. The results of the investigation is used as a basis for a discussion on the appropriateness of the incentive scheme in section 6.

# 2 NORWEGIAN REGULATIONS

As put forth by the Norwegian government, Meld. St. 16 is founded on concepts of *regulatory predictability* and *environmental sustainability* (Norwegian Ministry of Trade, Industry and Fisheries, 2015).

The emphasis on *environmental sustainability* is founded on the UN's sustainability principle and its three pillars: the economical, the social and the environmental. Growth in the aquaculture sector is assumed to generate economic surplus and improve social welfare, and the economic and social sustainability are assumed to be solved by ways of modern capitalism and democracy. The premise is therefore that the government, on behalf of the greater society, must impose limits on what is considered acceptable environmental impact.

The emphasis on *regulatory predictability* stems from the economic rationale found in the works of Nobel prize winners Finn. E. Kydland and Edward C. Prescott (Nobel Media, 2014). The reasoning of Kydland and Prescott showed that if a governing body has several different instruments of enforcement available, and if the selection and implementation of these instruments does not follow any pattern of predictability, it will generate uncertainty in terms of future economic policy. Any third party would then be subjected to this uncertainty, which in turn would affect the way they adapt. Since this uncertainty is unnatural in that it is manmade, it will lead to maladaptation and subsequent loss of value (Kydland and Prescott, 1977).

In accordance with Kydland and Prescott the government's chief objective should be to provide predictability, and with that improve efficiency by limiting possible ways the industry can waste resources adapting to ever changing policies. Accordingly, the policies presented first and foremost seek to clarify which regulatory instruments the government can and cannot use to guide the industry towards sustainability.

## 2.1 Current regulatory regime

In the regulatory paradigm enforced in Norway today, production of salmon and trout is regulated through licences that are granted by way of application. The application process is managed by Nærings- og Fiskeridepartmentet.

Each licence and location is at any time limited by the total active biomass allowed. Active biomass is defined as the total weight of fish in the sea, independently measured on both location and permit level, and is referred to as MTB.

*Location level MTB* is first and foremost a vehicle to manage local ecosystems. The farmer is required to document that the location is at a satisfactory environmental state between each production run. This is done to ensure that local natural homeostasis can be returned at any time. There are no suggestions to change the regulations that apply on a location level.

*Licence level MTB* is the primary method to regulate production volume. As production regulation by licence level has been proven quite successful compared to earlier systems, the government has explored different schemes for growth that can all be implemented within the current system. The government states that new regulations should look at new ways to internalize industry externalities.

Even though not predictable for the industry, the current regulatory regime gives a lot of flexibility in managing environmental considerations. Keeping the status quo does therefore not serve in the interest of facilitating industry growth. A constant growth policy would on the other hand be very predictable for the industry, but may prove difficult to enforce due to the wide range of different environmental conditions along the Norwegian cost.

## *2.2 The regional capacity rule*

Maintaining a level playing field for all actors in the industry, require a more granular instrument. The government therefore concludes that a third option, whereby a capacity rule governs industry growth by *regions* is the most suitable way to ensure industry predictability as well as environmental sustainability.

The proposed regional capacuty rule can be illustrated as a traffic light diagram where the consolidated state of one or more indicators is either red, green or yellow. The colour communicates the environmental status and/or development of a region.

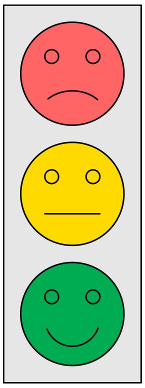


**Figure 2.1 - Traffic light diagram**

Increased production is only permitted if the indicators in that region allow it (green). If the indicators are not improving (yellow), or even falling behind (red), production licences will be either frozen or cut until next evaluation.

This system regulates the rate at which the industry can grow by considering if the environmental situation allows it. This is done in practise either by issuing new production licences, or by extending those already granted. The industry is assured predictability in how to be eligible for growth – both in terms of *when* the opportunity will arise, as well as the *criteria* upon which they are measured.

A central part of the system is that it is enforced on a regional basis. Everyone that holds a licence within the same region is given a collective interest in improving the state of the regional indicators, and is, by way of shared outcome, incentivized to cooperate.

## *2.3 Suggested implementation*

The suggested implementation of the above decision rule has three main elements that ensures industry predictability and environmental sustainability. They are: *the frequency* of which the industry is evaluated for adjustment; *the rate* of which the industry can be adjusted at each evaluation; and *the criteria* upon which adjustments are decided.

In determining the first, the government argues that the length between each evaluation must be adequate as to allow previous adjustments to take effect. Most consultative statements agree that intervals of two years would encompass the interest of most stakeholders (Ministry of Trade, Industry and Fisheries, 2015).

The government then argues a risk perspective to pinpoint the exact growth rate. Both economic and environmental risk, is in addition to the *frequency* of change, sensitive to the *rate* of change. Having decided on a frequency of two years, SINTEF research shows that the risk can be divided into three categories based in the rate of change in production volume: high (10 %), medium (6 %) and low (3 %) (Ministry of Trade, Industry and Fisheries, 2015).

If all regions keep within the green category grows at each period, the industry is predicted to reach its potential by the upside provided by the *high* risk-profile alone. On the downside, the industry would be severely negatively impacted if production output is crippled by numerous regions suffering from environmental difficulties.

Although growth would be suboptimal even in the best-case scenarios using moderate and low risk profiles, the downside risk and output volatility can be mitigated by the more conservative rates of change. The government concludes that a policy with a moderate risk profile would best align the industry’s incentives with considerations of environmental sustainability.

Finally, an appropriate selection of the indicators that drive the regional capacity rule is crucial to provide intended incentives, and the fact that something is important to the environment does not make it suitable for policy inclusion by default. A meaningful indicator should therefore exhibit the following properties:

- The indicator must be a reliable index to determine environmental status and development;
- There must be a proven correlation between the indicator and the production output in a region;
- There must be a realistic way for the industry to positively influence the indicator.

Out of several externalities evaluated, only the impact of sea lice is considered to inherit all the properties above, and the increased mortality rate in wild fish due to increased sea lice populations is considered an efficient indicator of measurement.

Havforskningsinstituttet has estimated how lice infections increase the mortality risk in wild fish (Ministry of Trade, Industry and Fisheries, 2015). Their findings can be categorized into risk categories based on the resulting increase in mortality rate in wild fish.

| Low | Medium | High |
|---|---|---|
| Less than 10% increase in mortality rate | Between 10-30% increase in mortality rate | Above 30% increase in mortality rate |

**Figure 2.2 Thresholds of sea lice impact on wild fisheries (Ministry of Trade, Industry and Fisheries, 2015).**

## *2.4 Policy summary*

In summary, Meld. St. 16 proposes a system that regulates production volume by measuring the environmental impact of salmon lice by regions every two years. If wild fish experience less that 10 % increase in mortality rate due to lice infection, farmers in that region are rewarded with a 6 % increase in allowed production licence. If the number of fatalities increase to between 10 % and 30 %, the current production volume is kept at status quo, and if the number rises to above 30 % the production licence is cut by 6 %.

# 3 INCENTIVES

Any externality that impacts costs or is otherwise accounted for in the market will usually be rectified by the industry, and, as stated in the introduction, any rational fish farmer would want to lower the levels of sea lice if there are economic incentives internal to the production process for doing so.

The current regulatory scheme in Norway require individual salmon farmers to keep the amount of sea lice per fish below some threshold on a per *location basis*. This policy suggests that it is not profitable to treat salmon on the margin today[1], and also that it is reasonable to believe that all incentives for managing sea lice are currently enforced externally through regulations.

In the new proposition outlined in Meld. St. 16 incentives are elevated to a group level in a scheme in which a collection of rules and incentives attempts to either compensate or punish farmers on a *regional basis* for their efforts towards sea lice management.

To develop a better understanding of the implications of the new scheme, a simple economic model will first be used to examine how the industry will adapt to the incentives on a group level. As the metric that regulates the incentive scheme is influenced on a per actor level, the model is then expanded to investigate the dynamics of collaboration within the group.

Insights into the incentive scheme can be found by analysing the question *'Does the benefits obtained outweigh the costs?'* In this setting the question can be further divided into three separate assessments: (i) *'Does the additional cost of managing sea lice stand in proportion to the benefits obtained from getting increases in production grants?'* (ii) *'Does the additional cost of managing sea lice stand in proportion to the benefits obtained from avoiding a production cut?'*, and (iii) *'Is managing sea lice at the current production volume profitable?'*.

---

[1] Although this is not given any further consideration in this thesis, this represents an interesting insight into how environmental externalities can be internalized by industry advancements.

Clearly, all questions apply on both individual and group level, but as cooperation is needed to achieve a beneficial outcome, the incentive scheme is first analysed on a group level.

## 3.1 Group level

The investigation is generalized by assuming that all actors are average in terms of being equally competent in all manners of conduct and subject to the same conditions[2], i.e. they get the same net present value from one single licence. This implies that if something is profitable for the group, it is also profitable for the average licence holder.

The three assessments (i), (ii) and (iii) can be expressed in terms of mathematical equations, and group setting the first two assessments, (i) and (ii), becomes:

$$(1+g)NPV - \Delta PV_{C_l} > NPV \tag{1}$$

$$NPV - \Delta PV_{C_l} > (1-g)NPV \tag{2}$$

Here $NPV$ is the current net present value of operations, $\Delta PV_{C_l}$ is change in cost associated with managing sea lice levels[3], and $g$ is used to denote the adjustment rate.

---

[2] Although there exists some evidence suggesting that lice spread between locations, lice is here treated as a regional phenomena (!REF NOT FOUND)

[3] Although $\Delta PV_{C_l}$ is different for a reduction scenario compared to a growth scenario, they can be treated as equal for mathematical convenience without loss in logic.

Equation ( 1 ) shows the requirement for the scheme to incentivise growth, and equation ( 2 ) the requirement to incentivise the industry to avoid reduction[4]. Both equations simplify into:

$$(g)NPV > \Delta PV_{C_l} \tag{3}$$

and further

$$g > \frac{\Delta PV_{C_l}}{NPV} \tag{4}$$

These equations show that the incentive scheme is only successful in motivating fish farmers towards lowering the current level of sea lice if the increase in net present value provided by the adjustment rate exceeds the additional cost of managing sea lice on a group level. Put in terms of the traffic light diagram from Meld. St. 16: in order to incentivise fish farmers to change from either red to yellow or from yellow to green, the adjustment rate must be greater than the ratio between the additional cost of managing salmon lice and the current net present value ( 4 ).

The third question (iii) quantifies the requirements for operations to be profitable with the cost of managing sea lice at the current level of production explicitly stated. It is here implied that the 'yellow' state in the traffic light diagram is equivalent to this neutral

---

[4] A linear relationship between increasing production output, the net present value and the cost of managing lice is assumed for mathematical convenience.

position, and if no one collaborates the state will go to ‘red’. The scenario is analysed by the examining the following equation:

$$NPV = PV_R - PV_C > (1-g)NPV + PV_{C_l} \tag{5}$$

which simplifies into

$$(g)NPV > PV_{C_l} \tag{6}$$

where

$$PV_C = PV_{C_l} + PV_{C_i}$$

Here $PV_R$ and $PV_C$ is the net present value of revenue and cost. $PV_C$ is the sum of all costs, and is explicitly highlighted by sea lice related costs, $PV_{C_l}$, and all other costs, $PV_{C_i}$.

The obvious interpretation of ( 6 ) is that the cost of managing sea lice must not exceed a limit where operations are no longer profitable at the group level.

## *3.2 Individual level*

The analysis is now expanded to investigate the dynamics of when several actors interact in a region, and the previous assumption that all licence holders are average, allows for analysis into how relative size impacts the cooperation dynamics. This is done by representing the size of an actor by the number of licences he holds, i.e. weighting an actor relative to his share of licences in a region.

With these assumptions in place, the inherit design of the incentive scheme permits for an interesting two-actor setup whereby the first actor represents some single licence holder, and the second actor represents the rest of the licence holders in a region[5]. The notation must be slightly modified to account for this.

Given an actor $i$ that holds an amount of licences $L_i$, the assumptions above allow for a linear relationship between the amount of licences an actor holds, and the resulting benefits and costs, i.e. for actor $i$ equation ( 5 ) can be written as:

$$L_i NPV = L_i PV_R - L_i PV_C > (1-g) L_i NPV + L_i PV_{C_l}$$

The 'average licence holder assumption' in the group level analysis has the subtle implication of requiring all licence holders to carry a negative benefit of sea lice that stands in proportion to their share of licences. This restriction is now relieved by allowing actors to neglect their share of the total cost sea lice management. By assuming lice to be a regional phenomenon that is increasing in the total amount of biomass available (Torrisen et. al. 2013), controlling the increased sea lice pressure in a region essentially becomes a zero-sum game. If one licence holder neglects his share, somebody else must cover the slack to achieve the same levels of sea lice for a given amount of licences in a region.

---

[5] This analysis does not include the effect of a licence holder having licences in other regions, which essentially gives the licence holder opportunity to hedge the effects of his actions. Nevertheless, the model is adequate in its portrayal of the dynamics to provide representative insight.

Even though this applies to all incentive scenarios described in the start of section 3, that is, assessment of growth (i), the assessment of avoiding reduction (ii), and the assessment of the status quo (iii), the underlying dynamics are the same when licence holders are allowed to misbehave by not taking responsibility for regional sea lice levels. Only scenario (iii) is analysed in a multi actor setup, but the results carry over nonetheless.

A multi-actor variation of scenario (iii) can be formalized as a coordination problem, that can be represented in a classic *coordination game* from game theory. In a coordination game the players (licence holders) can independently choose strategies in their pursuit of some mutual gains, but these gains can only be obtained if they make mutually consistent decisions by either cooperating, $C$, or defecting, $D$ (Cooper, 2009).

A typical coordination game can be represented by a payoff matrix that has the generic form shown in figure 3.2. Here the row player (A) has his payoff on the first line in each cell, and the column player (B) on the second. All coordination games require that equation ( 7 ) holds for the row player, and by symmetry ( 8 ) for the column player.

$$A > B \text{ and } D > C \tag{7}$$

$$a > c \text{ and } d > b \tag{8}$$

| $^{A}/_{B}$ | $C$ | $D$ |
|---|---|---|
| $C$ | $A,$<br>$a$ | $C,$<br>$c$ |
| $D$ | $B,$<br>$b$ | $D,$<br>$d$ |

**Figure 3.2 Generic coordination game**

Implementing formula ( 5 ) into a multi-actor environment yields the payoff matrix shown in figure 3.3[6].

---

[6] It is important to note that the term (1-g) in figure 3.3, shows how the *cumulative effort* required to avoid a non-beneficial outcome, must somehow be equal to the *total effort* exercised if all actors cooperate.

| $^A/_B$ | $C$ | $D$ |
|---|---|---|
| $C$ | $L_ANPV,$<br>$L_BNPV$ | $L_ANPV - L_BPV_{C_l},$<br>$L_BNPV + L_BPV_{C_l}$ |
| $D$ | $L_ANPV + L_APV_{C_l},$<br>$L_BNPV - L_APV_{C_l}$ | $(1-g)L_ANPV + L_APV_{C_l},$<br>$(1-g)L_BNPV + L_BPV_{C_l}$ |

**Table 3.2 Cooperative incentive scheme for status quo**

From the criteria given by the coordination game in equation ( 7 ) and ( 8 ), we get the two relationships shown in ( 9 ) and ( 10 ). Note that the notation $L_{-i}$ that is used to describe 'the other' actor.

$$L_iNPV > L_iNPV + L_iPV_{C_l} \qquad (9)$$

$$L_iNPV - L_{-i}PV_{C_l} > (1-g)L_iNPV + L_iPV_{C_l} \qquad (10)$$

Equation ( 9 ) simplifies into

$$PV_{C_l} < 0 \qquad (11)$$

Equation ( 11 ) conveys that sea lice is an externality that costs the licence holder money to manage, and that there would be no point in forcing internalization by regulation if 'farming' sea lice were profitable.

From equation ( 10 ) we get that

$$gL_iNPV > (L_i + L_{-i})PV_{C_l}$$

and further

$$\frac{g}{1+\frac{L_{-i}}{L_i}}NPV > PV_{C_l} \tag{12}$$

Equation ( 12 ) provides valuable insight, and when compared to the group level equation ( 6 ), it shows that the incentive to cooperate, $(g)NPV$, is reduced by an additional factor of $\left(1+\frac{L_{-i}}{L_i}\right)^{-1}$ for <u>all</u> actors in a region, and in addition that the reduction is larger for actors holding a smaller share of the licences in the region, i.e. $\frac{L_{-i}}{L_i}$ gets larger[7].

Having developed and evaluated the conditions of the game shown in figure 3.2, allows for some further examination of the properties given by game theory. If the requirements that is given by a coordination game holds, here equations ( 11 ) and ( 12 ), it becomes apparent that the additional relationship, $A > B \geq D > C$ holds for the row player, and by symmetry also for the column player. This form is a special case of a coordination game called a 'Trust game', or the 'Stag Hunt'[8], and is often used to describe social and economic cooperation.

---

[7] This reduction factor can also be derived from assessment (i) 'growth' and (ii) 'reduction', but its presence in scenario (iii) proves its existence and serves as a good basis for further discussion.

[8] The French philosopher Jean-Jacques Rosseau coined the canonical 'Stag Hunt'-game by describing a situation where two individuals go hunting. They can individually and without knowing what the other

All Stag Hunt-games have two pure strategy Nash equilibria[9] where one is payoff-dominant and the other is risk-dominant. This means that while all players prefer the efficient strategy (both actors cooperating) in terms of total payoff, the inefficient strategy (both actors defecting) is less risky in terms of variance depending on what the other player chooses.

The payoff structure clearly shows that it is possible for an actor to move costs onto the others by not collaborating. Although it is possible for all actors to benefit in terms of total economic surplus while also limiting their environmental footprint, as evident in figure 3.2, an actor could have several motives for risk aversion and choose the risk-dominant strategy by defecting.

# 4 HARVEST MODEL

To further analyse the implications of equation ( 12 ) its variables must be investigated. The relative size of an actor compared to the market, $\frac{L_{-i}}{L_i}$, is given at any time. So is the adjustment rate which is determined by the authorities.

---

selects choose to hunt either stag or hare. While an individual can get a less satiating hare by themselves, a successful stag hunting requires cooperation (Shor, 2006).

[9] In a n>2-player game a Nash-equilibrium is defined as the set of strategies choices where no player can benefit from changing their strategy given the other players choice (Nash, 1950).

The purpose of this section then becomes to explain the theory behind a harvest model that can be used to determine the other two variables in equation ( 12 ), namely the value of harvest, and the cost of lice. This will be used to provide a framework upon which a simulation model can be run to facilitate numeric insights into how the industry will adapt.

## *4.1 Optimal Harvesting of Farmed Fish*

Net present value is a collection of discounted cash flows that comprise both variable income and variable cost. A basic functional representation of salmon biology during the grow-out phase[10] is needed to further explore the underlying dynamics of the cash flows in salon aquaculture.

Bjørndal (1988) shows an analytic method for determining the optimal harvest time for a yearclass. The heuristic approach found in Asche and Bjørndal (2011)[11] proves a good fit for the purpose of this thesis, and is used as a foundation in describing some the underlying biological processes in this exposition.

The grow out-phase starts when new smolts are released into the sea. All fish released at the same time is of the same age and belong to the same yearclass. The total biomass of

---

[10] Short footnote on how the grow-out phase fits to the biological cycle in salmon aquaculture

[11] Extensions of this model is provided by eg Arnarson (1992), Heaps (1994) and Guttormsen (2008).

a yearclass, $B(t)$, can at any time be expressed as the number of fish alive, $N(t)$, multiplied by the average weight of a fish belonging to that yearclass, $w(t)$.

$$B(t) = N(t)w(t) \tag{13}$$

The number of fish in a yearclass, $N(t)$, will decline due to natural mortality. There is evidence for that the natural mortality rate fluctuate over time, both in wild and farmed fish (Asche and Bjørndal, 2011).

As the yearclass age, the growth rate will eventually start to decline, i.e. $w'(t) < 0$. Assuming adequate access to feed and a stable environment, this process follows a predictable pattern. However, in fish farming it is realistic to expect variable growth phases depending on seasonal changes in the environment (Asche and Bjørndal 2011).

To obtain the gross biomass value of a yearclass, denoted $V(t)$, price is multiplied by quantity. The price is a function of weight, where larger fish is typically valued above smaller fish.

By examining the figure above, it is apparent that even though the individual fish reaches its peak weight at time $\tilde{t}$, the maximal gross biomass value of a yearclass is reached earlier as some fish will perish during the period. This point in time is the optimal harvest time, and is denoted $t^*$.

A fish farmer is therefore presented with two alternatives: he can either harvest the fish and put the money in the bank to collect interest, or he can let the fish grow and defer harvesting to a later point in time. Leaving variable costs aside for now, it is evident that the farmer creates value by keeping fish in the sea while the gross biomass value increases at a rate higher than comparable interest rates. Asche and Bjørndal (2011) puts the above into two harvesting rules:

1. Do not harvest the fish if the capital in the form of fish gives a better return on investment than one can obtain from a bank.
2. Harvest the fish if the capital in the form of fish gives a lower return on investment than one can obtain from a bank.

The optimal harvest model is built upon the two harvesting rules above. First a zero-cost model with biomass growth and natural mortality is developed. As feeding represents the largest share of variable cost during the grow out phase it is later introduced into the model (Guttormsen, 2002).

The biological process is modelled as an adapted Beverton-Holt model[12], where the time variable $t$ also indicate the age of the yearclass. It is defined on the interval $t \in [0, T]$, where $T$ can be interpreted as either the time of sexual maturity or the species maximal lifetime. At time $t = 0$ an amount, $R$, new recruits are released into sea pens.

---

[12] The Beverton–Holt model is a classic discrete-time population model which gives the expected number of individuals in the next generation as a function of the number of individuals in the previous generation. (Beverton and Holt, 1957)

$$N(0) = R \tag{14}$$

$$\frac{dN}{dt} \equiv N'(t) = -M(t)\, N(t) \tag{15}$$

$$N(t) = Re^{-\int_0^t M(u)du} \tag{16}$$

Assuming the natural mortality rate is constant over time i.e. $M(t) = M$, the equations ( 15 ) and ( 16 ) can be simplified into:

$$\frac{dN}{dt} \equiv N'(t) = -MN(t) \tag{17}$$

$$N(t) = Re^{-M(t)} \tag{18}$$

All fish are assumed to be equal by simplifying away any stochastic variations. This implies that each fish represents the statistical average in terms of weight, natural mortality, vulnerability to salmon lice, etc. The total biomass for the entire yearclass can then at any time be defined as:

$$B(t) = N(t)\, w(t) = Re^{-M(t)}\, w(t) \tag{19}$$

Before turning the equation ( 19 ) into an optimization problem, two important assumptions are made to appropriately limit the investigative scope. *First*, all investments necessary for a farm to be operational are considered sunk. In explicit terms this means that all CapEx[13] considerations that would normally influence a business are not considered to be relevant. This includes new investments and their depreciation. *Second*, only the variable OpEx[14] needed to get a credible portrayal of cash flow from fish farming are included in the model (Asche and Bjørndal, 2011).

To properly account for the time value of money the biomass value is discounted. Using the discount rate $r$, an exponential discount function in the form of $e^{-rt}$ is employed. This exponential form is well behaved in optimization problems, and is a suitable approximation to the more commonly used discrete function $(1+r)^{-t}$ for small values of $r$.

Before including any variable cost, the present value of a yearclass takes the functional form of:

$$V(t)e^{-rt} = p(w)\,B(t)\,e^{-rt} \tag{20}$$

A farmer seeking to maximize his profits then faces the following optimization problem:

---

[13] Capital expenditure, or CapEx, are funds used by a company to acquire or upgrade physical assets such as property, industrial buildings or equipment. It is often used to undertake new projects or investments by the firm. (Investopedia, 2016)

[14] An operating expense, operating expenditure, operational expense, operational expenditure or Opex is an ongoing cost for running a product, business, or system. (Wikipedia, 2016)

$$\max_{\{0\leq t\leq T\}} \pi(t) = V(t)e^{-rt} \tag{21}$$

where the first-order conditions are given by:

$$\pi'(t) = V'(t)e^{-rt} - \mathrm{r}V(t)e^{-rt} = 0 \tag{22}$$

and the optimal harvesting time must then, in addition to any second-order conditions, satisfy:

$$V'(t^*) = rV(t^*) \tag{23}$$

Equation ( 23 ) is a mathematical representation of the harvesting rule. By interpretation it says that the optimal time for harvest is when the marginal increase in value from fish growing in the sea, equals the opportunity cost in the form of economic rent obtainable elsewhere, usually in the form of interest rate from a bank. It can be rewritten to display the relationship between the marginal revenue (left side), and the marginal cost (right side):

$$\frac{p'(w)}{p(w)}w'(t) + \frac{w'(t^*)}{w(t^*)} = r + M \tag{24}$$

The marginal revenue curve varies over time. Increasing either the growth rate or the price, would increase the optimal harvesting time, and shift the marginal revenue curve upwards. In the zero-cost case the marginal cost curve is horizontal because the factors

$r + M$ are constant. Increasing either the discount rate or the mortality rate would decrease the optimal harvesting time, and cause an upwards shift in the marginal cost curve.

Next, variable costs in the form of feeding costs are introduced. The relationship between the weight change (growth), $w'(t)$, and feed quantity, $F(t)$, is determined by the *feed conversion ratio* $f_t$.

$$f_t = \frac{F(t)}{w'(t)} \tag{25}$$

The total feed quantity varies over time:

$$F(t)\, N(t) = f_t\, w'(t) R e^{-Mt} \tag{26}$$

By integrating equation ( 26 ) and multiplying by the unit cost of feed, and a discount factor, one obtains the present value of feeding costs:

$$C_f \int_0^t F(u)\, R e^{-Mu} e^{-ru} du \tag{27}$$

which gives the new maximization problem:

$$\max_{\{0 \le t \le T\}} \pi(t) = V(t) e^{-rt} - C_f \int_0^t F(u)\, R e^{-(M+r)u} du \tag{28}$$

and the first-order conditions are now given by:

$$\pi'^{(t)} = V'^{(t)e^{-rt}} - \mathrm{r}V(t)e^{-rt} - C_f F(t)Re^{-(M+r)t} = 0 \qquad (29)$$

The new marginal revenue and marginal cost curves then takes the functional form of:

$$\frac{p'(w)}{p(w)}w'(t^*) + \frac{w'(t^*)}{w(t^*)} = r + M + \frac{C_f F(t^*)}{p(w)w(t^*)} \qquad (30)$$

By comparing equation ( 24 ) to equation ( 30 ), one observes that a term is now added to the left side onto the marginal cost, and that this term is sensitive in feed cost as well as the feed conversion ratio.

## *4.2 The Added Cost of Salmon Lice*

When infected by parasitic sea lice a host will typically respond with reduced appetite and growth. Wounds left by lice can also increase the hosts sensitivity to secondary infections (Costello, 2006; Johnson, 2004; Tully, 2002). While the impact on biomass development can come from influencing both the growth rate and the mortality rate, sea lice rarely impact host mortality directly (Pike, 1999).

Although there is little research done on the economic impact of lice, Mustafa, Randkaduw and Campbell (2001) found that reduced fish growth accounted for the greatest financial loss by as much as 200g per fish per cycle. Costello (2009) found that the "most significant costs of sea lice where control is successful in preventing pathogenicity, are treatment costs, reduced fish growth, and reduced food conversion efficiency."

Abolofia, Asche and Wilen (2016) employs an adapted harvesting model that builds on the models of Bjørndal (1988), Arnason (1992), Heaps (1993), and Guttormsen (2008) to show the economic impact of sea lice on farm profits. The model accounts for the impact of salmon lice by allowing biomass growth $\dot{B}$ to be a function of a time varying level of lice per fish $L(t)$. In addition, the costly chemical delousing treatments are accounted for in the discounted revenues from a single rotation:

$$\begin{aligned} \Pi(T) = P(T) \cdot \left( B_0 + \int_0^T \dot{B}(t, L(t)) \cdot dt \right) \cdot e^{-rT} \\ - C_f \int_0^T FCR \cdot \dot{B}(t, L(t)) \cdot e^{-rt} \cdot dt - C_r \sum_{n=1}^{N} e^{-rT_n} \end{aligned} \tag{31}$$

Here $T$ is the harvest time, $P(T)$ the price per kg fish, $B_0$ the initial lice-free biomass and $\dot{B}(\cdot)$ the biomass growth.

By comparing its discounted net revenues to another identical production cycle free of sea lice, the model allows for economic quantification of an infestation of sea lice on a production cycle by comparing its discounted net revenues to another identical production cycle free of lice.

$$\Pi(T)^{nolice} - \Pi(T)^{lice} = \underbrace{e^{-rT} \cdot P(T) \sum_{i=1}^{I} \int_{t_{i-1}}^{t_i} \left(\dot{B}(t,0) - \dot{B}\big(t, L(t)\big)\right) dt}_{Revenue\ loss}$$
$$- \underbrace{C_f \cdot FCR \sum_{i=1}^{I} \int_{t_{i-1}}^{t_i} \left(\dot{B}(t,0) - \dot{B}(t, L(t))\right) e^{-rt} \cdot dt}_{Feed\ cost\ savings} + \underbrace{C_r \sum_{n=1}^{N} e^{-rT_n}}_{Treatment\ cost} \tag{32}$$

The above model shows that a lice infestation will impact the revenues of a fish farm in the form of 'Revenue Loss', 'Feed cost savings' and 'Treatment cost'.

# 5 NUMERIC SIMULATION

The theory and models shown in section 4 are in this section implemented into a stylized 'real world setting', which purpose is attempting to identify the maladaptation patterns predicted in section 3. This is done by simulating a selection of biological and economic processes in different scenarios. The simulation model is written in MATLAB, and the full source code is included in appendix A.

The proposition described in section 2 suggests that the industry's regulatory potential for growth should be governed by environmental considerations, and further that the aquaculture sector should solve their environmental externalities by way of incentivized collaboration. The specification of the simulation model should capture this by allowing for scenarios with and without the shared responsibility scheme, as well as simulations where only some licence holders choose to collaborate.

Although the proposition suggests the use of an adjustment rate as the instrument for both positive and negative incentives, it is beyond the scope of this thesis to investigate at which monetary level the suggested adjustment rate might provide proper incentive strength for the industry to adjust levels of sea lice. The model does however provide opportunity to compare the relative economic and environmental impact between two scenarios, i.e. one scenario where sea lice is regulated by putting thresholds on each location like today, and another scenario where sea lice levels is governed only by a shared responsibility scheme like the one proposed in Meld. St. 16.

The key insights provided in section 3 depicts a relationship between several variables, notably in equation ( 12 ). The specification of the simulation model should capture these dynamics by allowing for different scenarios where the licence holders have different ratios of the total market. However, the simulation specifications should not allow for a licence operator to hold any advantage compared to others in terms of extracting value from each licence, i.e. deviating from the assumption made in section 3.

Finally, the specification must include modelling some aspects of the exogenous environment in order for the model to give meaningful insights into the dynamics of how fish farms interact with each other, and affect the environment. This includes a model for sea lice populations.

## 5.1 Model Calibration and Implementation

Several of the input functions and parameters in the model are externally given, while others must be determined through different approaches. The functions and parameters are grouped into three categories below.

### 5.1.1 Regulatory

Licence level MTB is the Norwegian authority's primary method of regulating production volume, and it is therefore an essential metric in the proposed regulation. Although the impact as seen in section 3 has more to do with the ratio of licences between actors, the implementation restricts the maximal allowed biomass to 780 ton for each license as a baseline. The model has a method that checks if a licence breaches the MTB limit, and subsequently force an offending licence operator to do a 'MTB harvest' to bring biomass volume back within requirements.

The model uses three locations per licence, and the licence operator cycles between the locations so that there is always one inactive[15]. In addition, the model ensures that all locations are inactive for a minimum two months after a production cycle (Akvakulturforskriften, 2008, §40).

---

[15] This is a realistic representation of how aquaculture is conducted in Norway today (Asche and Bjørndal, 2011).

The application of the adjustment rate is governed by the three states of the regional capacity rule, and the states are determined based on the relative increase in mortality rate found in wild fish. As argued in the start of this section, the purpose of this thesis is not to determine whether the adjustment rates give proper monetary incentives at any given time, but rather if they serve the purpose of incentivising collaboration. The adjustment rate is therefore modelled implicitly by running several scenarios where licence holders are either complying or defecting with the sea lice regulations, and measuring the relative impact on both the licence holders and the environment.

The implementation of sea lice regulating efforts then functions in two ways. In scenarios where the farming is governed by the current regulations, the levels of sea lice are strictly limited at 0.1 lice per fish[16] (Forskrift om bekjempelse av lakselus i akvakulturanlegg, 2012, §7) by having operators implement lice regulating efforts on the location if it rises above this threshold. In scenarios where the new regional capacity rule is employed, a process monitors the mortality rate of wild fish that requires collaborating operators to expend lice managing efforts towards keeping the increase in mortality rate below a threshold of 30%[17], while defecting operators are not required to utilize any lice regulating efforts.

---

[16] The model does not distinguish between male and female lice.

[17] This is in accordance with the neutral level 'yellow', used the regional capacity rule described in section 2.

### *5.1.2 Biological*

In addition to the wild fish in found in farming installations, the simulated region is modelled to have a population of sea lice inhabiting its ecosystem. The simulation is not developed as a venture into accurately describing the sea lice life cycle and biology, but rather to provide adequate representation for the purposes of this model.

The population of sea lice is therefore modelled using a simple formula that converges to an upper limit of saturation. The upper limit is a function of the ratio between total biomass and the number of lice in the region, and is set sufficiently high to be of consequence in the model. More importantly, the derivative of the function is in addition to being a function of the sea lice population itself, also a function of the regional ratio between sea lice and farmed salmon.

By employing this function, the sea lice population will trace the regional development of farmed salmon, and consequently give a response in growth rate for sea lice at all amounts of farmed biomass, either up or down. At the start of the model, before introducing any farmed fish into the ecosystem, the sea lice population is modelled to exist in a homeostasis within the environment. This serves as a baseline for comparing the impact of farmed fish on sea lice levels, and subsequently, as a baseline for the increased impact of sea lice on wild fish mortality due to the presence aquaculture.

As the framework for a biological growth model of salmon has already been developed in section 4.1, only a function for the weight curves, and some parameters are needed for a direct implementation into the simulation model. Weight curves vary depending on species, location and the environment amongst other factors. Asche and Bjørndal (2011) provides a polynomial that has been estimated on empirical data from observations of farmed salmon growth, as well as an anecdotal mortality rate which gives approximately 10% yearly reduction in yearclass population.

In addition to the weight curve, a simple relationship between sea lice and farmed salmon weight development is devised. This parameter is calibrated with the sole purpose of giving noticeable effect the model, and is set so that the presence of each 0.1 lice per fish results in a 10% decrease in weight over a year. This leads to loss in revenue.

### *5.1.3 Economic*

After implementing methods to manage and handle sea lice and its impact, as well as cycling of the production runs, the only process in the model left to the licence operator is to determine the optimal harvest time for each yearclass. This is done by having the model extrapolate biomass data into the future, and the estimated weight and growth data is calculated using the growth function described earlier. The function then has the licence operator execute an 'optimal harvest' to empty the pen and put the location into the cycle where it waits for a new yearclass.

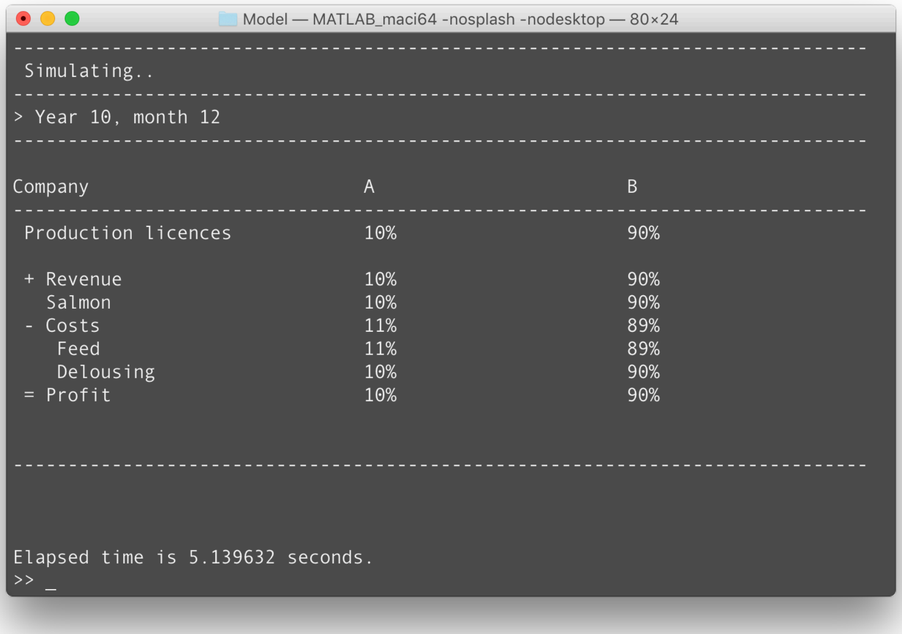
This approach allows for a very granular control of the future information that is put into the harvesting decision, i.e. that a fish farmer cannot predict an outbreak of sea lice in the future, but can assume that the impact of sea lice on biomass generating revenue follows from the current levels of sea lice.

All monetary values are normalized and based on the price of a single kilogram of salmon which is for convenience purposes set to ‘1’. Utilizing the correct normalized prices for farmed salmon meat, salmon feed or sea lice treatment is not necessary of this model to indicate any maladaptation patterns in the incentive scheme. The values are therefore set sufficiently high as to enable easy detection of any such patters, but also appropriately low for the sake of realism.

## *5.2 Model output and results*

In all simulations, the effect of the incentive scheme is investigated by giving licence holder ‘A’ a relative share of licences equal to 10%, while licence holder ‘B’, representing the rest of the industry, is given the remaining 90% of licences. The resulting consequences for the group can be investigated by looking at the collective surplus in each simulation. All simulations are run over a 10-year period, with monthly simulation intervals. The actors are compared relative to eachother.

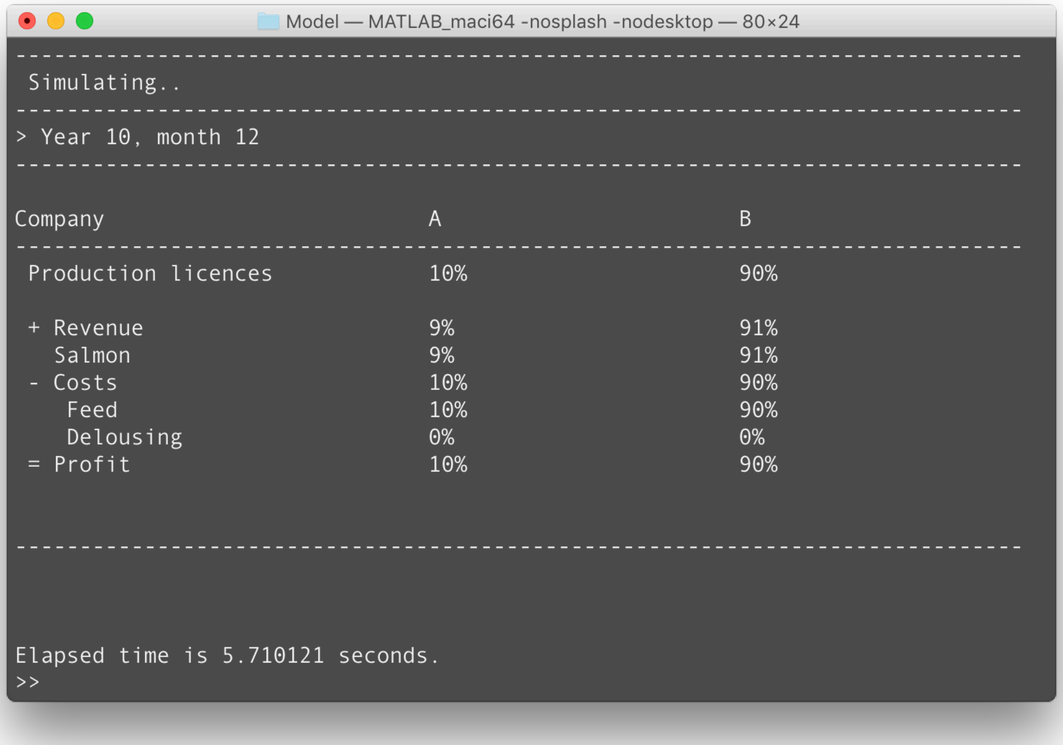
First, the model is configured so that all licence operators comply with sea lice regulations enforced on a location level only, this is similar to what is enforced under the current regulatory regime. This is similar to what a scenario with full cooperation under the regional capacity rule would look like. Although model does not capture it, if full cooperation were to be achieved over the two year evaluation period with the resulting decrease in lice, both the environment as well as the total economic surplus would benefit.



```
-------------------------------------------------------------------------------
 Simulating..
-------------------------------------------------------------------------------
> Year 10, month 12
-------------------------------------------------------------------------------

Company                         A                       B
-------------------------------------------------------------------------------
 Production licences            10%                     90%

 + Revenue                      10%                     90%
   Salmon                       10%                     90%
 - Costs                        11%                     89%
    Feed                        11%                     89%
    Delousing                   10%                     90%
 = Profit                       10%                     90%


-------------------------------------------------------------------------------



Elapsed time is 5.139632 seconds.
>> _
```

**Figure 5.1 - Scenario 1 - Cooperation**

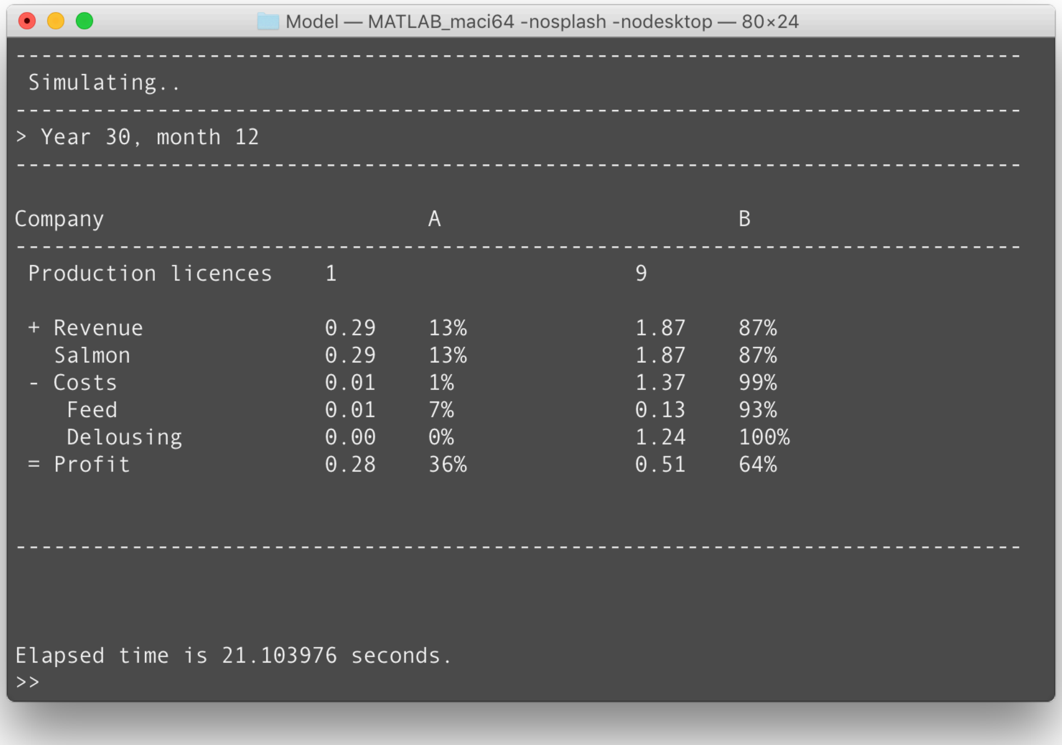
Second a scenario is shown where there is no enforcement on a location level, and neither actor ‘A’ nor ‘B’.



```
 ----------------------------------------------------------------------------
  Simulating..
 ----------------------------------------------------------------------------
 > Year 10, month 12
 ----------------------------------------------------------------------------

 Company                        A                       B
 ----------------------------------------------------------------------------
  Production licences           10%                     90%

  + Revenue                     9%                      91%
    Salmon                      9%                      91%
  - Costs                       10%                     90%
     Feed                       10%                     90%
     Delousing                  0%                      0%
  = Profit                      10%                     90%


 ----------------------------------------------------------------------------



 Elapsed time is 5.710121 seconds.
 >>
```

**Figure 5.2 - Scenario 4 – No cooperation**

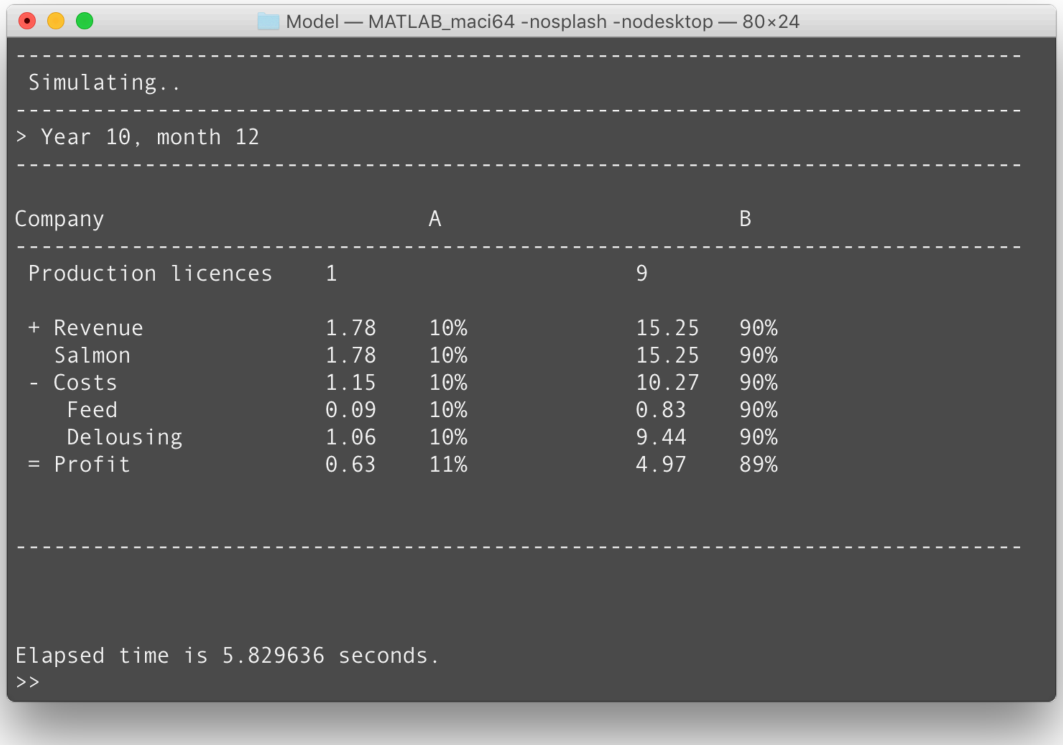
Below a scenario where ‘A’ defects and forces costs over on ‘B’.



```
--------------------------------------------------------------------------------
 Simulating..
--------------------------------------------------------------------------------
> Year 30, month 12
--------------------------------------------------------------------------------

Company                         A                       B
--------------------------------------------------------------------------------
 Production licences    1                       9

 + Revenue              0.29    13%             1.87    87%
   Salmon               0.29    13%             1.87    87%
 - Costs                0.01    1%              1.37    99%
    Feed                0.01    7%              0.13    93%
    Delousing           0.00    0%              1.24    100%
 = Profit               0.28    36%             0.51    64%


--------------------------------------------------------------------------------



Elapsed time is 21.103976 seconds.
>>
```

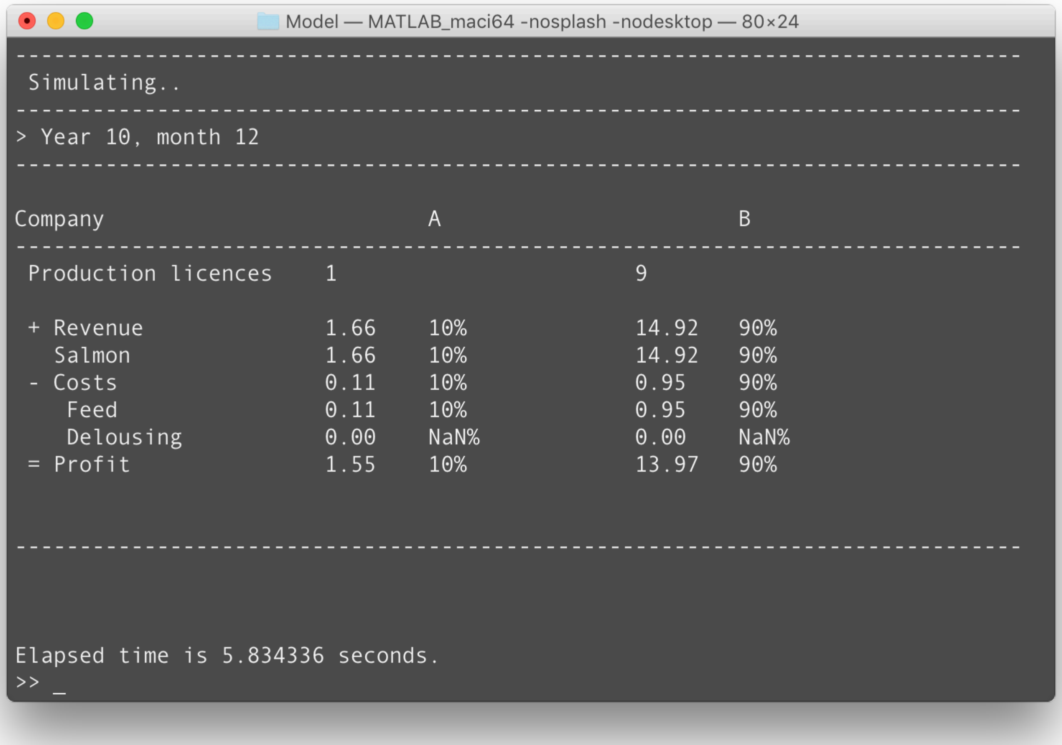
**Figure 5.3 - A defects**

Last, two scenarios are shown where both ‘A’ and ‘B’ are collaborating. The first one is including the presence of lice, and the second one is without it. Notice how feed costs go down, but treatment costs go up.



```
--------------------------------------------------------------------------------
 Simulating..
--------------------------------------------------------------------------------
> Year 10, month 12
--------------------------------------------------------------------------------

Company                         A                       B
--------------------------------------------------------------------------------
 Production licences    1                       9

 + Revenue              1.78    10%             15.25   90%
   Salmon               1.78    10%             15.25   90%
 - Costs                1.15    10%             10.27   90%
    Feed                0.09    10%             0.83    90%
    Delousing           1.06    10%             9.44    90%
 = Profit               0.63    11%             4.97    89%


--------------------------------------------------------------------------------



Elapsed time is 5.829636 seconds.
>>
```

Figure 5.4 - Scenario 5 – Lice, collaboration



```
--------------------------------------------------------------------------------
 Simulating..
--------------------------------------------------------------------------------
> Year 10, month 12
--------------------------------------------------------------------------------

Company                         A                       B
--------------------------------------------------------------------------------
 Production licences     1                       9

 + Revenue               1.66     10%            14.92    90%
   Salmon                1.66     10%            14.92    90%
 - Costs                 0.11     10%            0.95     90%
    Feed                 0.11     10%            0.95     90%
    Delousing            0.00     NaN%           0.00     NaN%
 = Profit                1.55     10%            13.97    90%



--------------------------------------------------------------------------------




Elapsed time is 5.834336 seconds.
>> _
```

**Figure 5.5 - No lice, collaboration**

# 6 CONCLUSION

It is possible to further quantify and fine-tune the simulation model from section 5. This can be done in several ways, i.e. by examining the aquaculture industry to improve the model dynamics; by employing more realistic biological models, esp. for sea lice development; or, by updating the monetary input variables to reflect more accurate market values. Either way, the experimental results show the presence of free rider behaviour, and that in order for the incentive scheme to have effect, rules must be enforced on a per actor basis.

This is underpinned by the theoretical solution in section 3. It shows that if the incentive scheme provides individual actors with the ability to neglect their share of collaboration, even though it does not force externalities onto others, then there is no change in fact that the underlying subtleties allows for opportunistic free rider behaviour, but only as long as the benefits of achievement outcome applies to the entire group collectively regardless of effort and participation.

The proposition identifies three elements that comprises the risk profile of the regional capacity rule. These elements are the adjustment rate; the possible frequency of the adjustments; and, the selection of indicators that governs the application of the adjustment rate. As both the adjustment rate and its applicable frequency are to be considered fixed for the sake of industry predictability, only the selection of indicators that governs the application of adjustment rate are left available as an instrument for the authorities. But tuning the indicators will not have any effect as long as the incentive scheme allows for opportunistic behaviour.

It is worth noting that if the incentive scheme should work on a *group level*, it will only hold true as long as the benefit obtained from the adjustment rate is sufficient to compensate for the additional sea lice treatment costs. If it breaks down, i.e. if the adjustment rate is too low, the game changes and most of the monetary incentives for cooperation is lost.

In addition, by utilizing a fixed adjustment rate Meld. St. 16 indirectly assumes that a fixed adjustment rate is somehow applicable for growth at any level of production. While this increases the incentive strength if production is subject to economies of scale, it also implies the existence of an upper limit of incentive effectiveness if production is subject to diseconomies of scale, and, by extension, raises the questions of where this interval is defined.

In conclusion, this thesis provides evidence to believe that the proposition is presented with the false belief or logical fallacy that the group level incentives offered in Meld. St. 16 are also applicable at an individual level.

# 7 ACKNOWLEDGMENTS

I would like to thank those of whom have made this thesis possible. My girlfriend Julie for her unwavering support, my supervisor Frank for his valuable insights and agreeable nature, and my boss Eric for his understanding and graciousness in providing much needed time off.

# 8 LITERATURE CITED

Asche, F. and Bjørndal, T. (2011). *The Economics of Salmon Aquaculture (2nd Edition).* Chichester: Wiley-Blackwell

Naylor, Rosamond L,, Goldburg, Rebecca J,, Primavera, Jurgenne H., Kautsky, N., Beveridge, Malcolm C. M., Clay, J., Folke, C., Lubchenco, J., Mooney, H., and Troell, M. *Effect of aquaculture on world fish supplies,* Nature 405, 1017-1024 (29 June 2000), doi:10.1038/35016500

Ministry of Trade, Industry and Fisheries. (2015). *Forutsigbar og miljømessig bærekraftig vekst i norsk lakse- og ørretoppdrett*. Meld. St. 16 (2014-2015). https://www.regjeringen.no/contentassets/6d27616f18af458aa930f4db9492fbe5/no/pdfs/stm201420150016000dddpdfs.pdf

Nobel Media AB. (2014). *The Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 2004* http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2004/

Kydland, F., & Prescott, E. (1977). *Rules Rather than Discretion: The Inconsistency of Optimal Plans. Journal of Political Economy, 85*(3), 473-491

Torrissen, O., Jones, S., Asche, F., Guttormsen, A., Skilbrei, O.T., Nilsen, F., Horsberg, T.E., Jackson, D., 2013. *Salmon lice - impact on wild salmonids and salmon aquaculture.* Journal of Fish Diseases 36, 171–194

Cooper, R. (2009). *Coordination Games*. doi:10.1017/CBO9780511609428

Shor, M. (2006). *Stag Hunt, Dictionary of Game Theory Terms, Game Theory .net*. http://www.gametheory.net/dictionary/games/StagHunt.html

Nash, J. (1951). *Non-Cooperative Games. Annals of Mathematics, 54*(2), second series, 286-295. doi:10.2307/1969529

Bjørndal, T. (1988). *Optimal Harvesting of Farmed Fish. Marine Resource Economics, 5*(2), 139-159

Arnason, R. (1992) *Optimal Feeding Schedules and Harvesting Time in Aquaculture, Marine Resource Economics 7*(1), 15–35.

Heaps, T. 1995. *Density Dependent Growth and the Culling of Farmed Fish. Marine Resource Economics 10*(3): 285-298.

*Guttormsen, A.G. (2008) Faustman in the Sea: Optimal Rotation in Aquaculture. Marine Resource Economics, 23, 401-410.*

Guttormsen, A. G. (2002) *Input Factor Substitutability in Salmon Aquaculture. Marine Resource Economics*, 17, 91-102.

Beverton, R. J. H.; Holt, S. J. (1957), *On the Dynamics of Exploited Fish Populations, Fishery Investigations Series II Volume XIX*, Ministry of Agriculture, Fisheries and Food

Investopedia. (2006). *Capital Expenditure (CAPEX).* http://www.investopedia.com/terms/c/capitalexpenditure.asp

Wikipedia. (2006). *Operating Expense*. https://en.wikipedia.org/wiki/Operating_expense

Costello, M. J. (2006). *Ecology of sea lice parasitic on farmed and wild fish. Trends in Parasitology, 22*(10), 475-483.

Johnson, S. C., Treasurer, J. W., Bravo, S., Nagasawa, K., & Kabata, Z. (2004). *A review of the impact of parasitic copepods on marine aquaculture. Zoological Studies, 43*(2), 229-243.

Tully, O., & Nolan, D. T. (2002). *A review of the population biology and host-parasite interactions of the sea louse Lepeophtheirus salmonis (Copepoda : Caligidae)*. Parasitology, 124, S165-S182.

Pike, A. W., & Wadsworth, S. L. (1999). *Sealice on salmonids: Their biology and control. Advances in Parasitology*, Vol 44, 44, 233-337.

Mustafa, A., Rankaduwa, W., & Campbell, P. (2001). *Estimating the cost of sea lice to salmon aquaculture in eastern Canada. Canadian Veterinary Journal-Revue Veterinaire Canadienne, 42*(1), 54-56.

Costello, M. J. (2009). *The global economic cost of sea lice to the salmonid farming industry. Journal of Fish Diseases, 32*(1), 115-118.

Abolofia, J., Asche, F. & Wilen, J. E. (2016) The Cost of Lice: Quantifying the Impacts of Parasitic Sea Lice on Farmed Salmon

Forskrift om drift av akvakulturanlegg (akvakulturdriftsforskriften)

Forskrift om bekjempelse av lakselus i akvakulturanlegg

# 9 APPENDIX A – MATLAB MODEL

The source code is provided ‘as-is’, complete with last minute modifications, and the reader must be warned that it is of an experimental nature that might require the instructions of the author to run with the correct parameters and settings.

```
initialize



global modelDebug;

global plotting;

global eventLog;



global tickRate;

global simulationLength;



global environment;

global regulation;

global fishery;



global t;



tic % Start timer
```

```
    %% Loop through simulation period

    for t = 1:simulationLength*tickRate


        %environment = updateLice(environment);

        %environment = updateSalmonWild(environment);




        %% Loop through all fisheries

        for company = 1:length(fishery)




                fishery(company) = operate(fishery(company),'checkLiceLevels');

                fishery(company) = operate(fishery(company),'updateData');

                fishery(company) = operate(fishery(company),'checkOptimalHarvestTimes');

                fishery(company) = operate(fishery(company),'checkMTB');

                fishery(company) = operate(fishery(company),'checkForEmptyLocations');


                %environment = updateSalmonFarmed(environment,aggregate(fishery(company),t,'Biomass'),aggregate(fishery(company),t,'Population'));
```

```
       end    % ==> End fisheries loop

       logger % Dumps info to screen

   end % ==> End simulation loop

   toc % End timer

   %% -------------------------------------------------------------------------------------
-------------------
   classdef companyObject
   %% -------------------------------------------------------------------------------------
-------------------

       properties
              licence;
              companyName;
       end
```

```
   %% -------------------------------------------------------------------------------------
-------------------

    methods

        %% Class constructor
        %% ---------------------------------------------------------------------------
----------------------
        function obj = companyObject(companyName)
            obj.companyName = companyName;
        end
        %% ---------------------------------------------------------------------------
----------------------


        %% Adds licence to company
        %% ---------------------------------------------------------------------------
----------------------
        function obj = addLicence(obj)
            if length(obj.licence) == 0
                obj.licence                                                    =
licenceObject(obj.companyName,length(obj.licence)+1);
            else
```

```
                    obj.licence(length(obj.licence)+1)                    =
licenceObject(obj.companyName,length(obj.licence)+1);

                end

            end

            %% --------------------------------------------------------------------------
---------------------



            %% Obtains aggregate data from all company licences

            %% --------------------------------------------------------------------------
---------------------

            function output = aggregate(obj,time,dataType)

                output = 0;

                for i = 1:1:length(obj.licence)

                    output = output + aggregate(obj.licence(i),time,dataType);

                end

            end



            %% Cascade some operating function down to all the licenses

            %% --------------------------------------------------------------------------
---------------------

            function obj = operate(obj,operation)

                for i = 1:1:length(obj.licence)
```

```
                                if strcmp(operation,'checkForEmptyLocations')
                                        obj.licence(i)  =  operate(obj.licence(i),operation);  %
Escalate to licence level
                                elseif strcmp(operation,'checkOptimalHarvestTimes')
                                        obj.licence(i)  =  operate(obj.licence(i),operation);  %
Escalate to licence level
                                elseif strcmp(operation,'checkMTB')
                                        obj.licence(i)  =  operate(obj.licence(i),operation);  %
Escalate to licence level
                                elseif strcmp(operation,'updateData')
                                        obj.licence(i)  =  operate(obj.licence(i),operation);  %
Escalate to licence level
                                elseif strcmp(operation,'checkLiceLevels')
                                        obj.licence(i)  =  operate(obj.licence(i),operation);  %
Escalate to licence level
                                end
                        end
                end

        end
```

```
   %% -------------------------------------------------------------------------------------
-------------------

   end % End classdef



   %% -------------------------------------------------------------------------------------
-------------------

   classdef licenceObject

   %% -------------------------------------------------------------------------------------
-------------------



       properties

              location;

              companyName;

              licenceNumber;

       end



   %% -------------------------------------------------------------------------------------
-------------------



       methods



              %% Class constructor
```

```
              %% ------------------------------------------------------------------------
---------------------

              function obj = licenceObject(companyName,licenceNumber)

                     obj.companyName = companyName;

                     obj.licenceNumber = licenceNumber;

                     obj.location  =  [locationObject(obj.companyName,obj.licenceNumber,1)
locationObject(obj.companyName,obj.licenceNumber,2)
locationObject(obj.companyName,obj.licenceNumber,3)
locationObject(obj.companyName,obj.licenceNumber,4)];

                      %locationObject locationObject locationObject];

              end



              %% Returns aggregate data from all locations on licence

              %% ------------------------------------------------------------------------
---------------------

              function output = aggregate(obj,time,dataType)

                     output = 0;

                     for i = 1:1:length(obj.location)

                            if    strcmp(dataType,'Biomass')    output    =    output    +
obj.location(i).salmon.B(time);

                            elseif   strcmp(dataType,'Population')   output   =   output   +
obj.location(i).salmon.N(time);
```

```
                elseif strcmp(dataType,'revenue') output = output +
obj.location(i).revenue(time);
                elseif strcmp(dataType,'Supply') output = output +
obj.location(i).S(time);
                elseif strcmp(dataType,'FeedCost') output = output +
obj.location(i).feedCost(time);
                elseif strcmp(dataType,'TreatmentCost') output = output +
obj.location(i).treatmentCost(time);
                end
                %if strcmp(dataType,'TreatmentCost') aggregate = aggregate +
obj.location(i).TC(time); end
            end
        end
        %% ----------------------------------------------------------------------
---------------------

        %% Operating functions
        %% ----------------------------------------------------------------------
---------------------
        function obj = operate(obj,operation)
            global eventLog;
            global regulation;
            global t;
```

```
            if strcmp(operation,'checkForEmptyLocations')

                for i = 1:1:length(obj.location)
                    % Find locations that are currently active
                    activeSites(i) = (obj.location(i).salmon.N(t))>0;
                    % Find locations that has been passive for the required
amount of time
                    passiveSites(i) = (obj.location(i).salmon.N(t +
regulation.minResetPeriod*((t<=regulation.minResetPeriod)-1)))>0;
                end

                if sum(activeSites) <= length(obj.location) - 2
                    passiveSites = find(passiveSites==0);

                    for i = 1:1:length(passiveSites)
                        releaseDate(i) =
obj.location(passiveSites(i)).releaseDate(end);
                    end

                    for i = 1:1:length(passiveSites)
```

```
                                        if
obj.location(passiveSites(i)).releaseDate(end)==min(releaseDate)

                                                activate = passiveSites(i);

                                        end

                                end



                                obj.location(activate)                                  =
addYearclass(obj.location(activate));



                                LogIndex = length(eventLog)+1;

                                eventLog{LogIndex,1} = t;

                                eventLog{LogIndex,2} = sprintf('Comp. %s added yearclass
to location %d on licence %d',obj.companyName,activate,obj.licenceNumber);

                        end



                elseif strcmp(operation,'checkOptimalHarvestTimes')

                        global regulation;

                        global eventLog;

                        global t;



                        for i = 1:1:length(obj.location)
```

```
                    obj.location(i) = updatePV(obj.location(i));

                    if obj.location(i).PV(t) ~= 0 && obj.location(i).PV(t) == max(obj.location(i).PV(t:end))

                        harvestAmount = obj.location(i).salmon.N(t);
                        obj.location(i) = harvest(obj.location(i),harvestAmount);
                        obj.location(i) = updateB(obj.location(i));

                        LogIndex = length(eventLog)+1;
                        eventLog{LogIndex,1} = t;
                        eventLog{LogIndex,2} = sprintf('Comp. %s harvested (optimal) site %d on licence %d',obj.companyName,i,obj.licenceNumber);
                    end
                end

            elseif strcmp(operation,'checkMTB')
                global t;
                global eventLog;
                global regulation;
```

```
                             for i = 1:1:length(obj.location)
                                    % Find locations that are currently active
                                    activeSites(i) = (obj.location(i).salmon.N(t))>0;
                                    % Find locations that has been passive for the required
amount of time
                                    passiveSites(i)    =    (obj.location(i).salmon.N(t    +
regulation.minResetPeriod*((t<=regulation.minResetPeriod)-1)))>0;
                             end

                             if aggregate(obj,t,'Biomass') >= regulation.MTB

                                    activeSites = find(activeSites);

                                    for i = 1:1:length(activeSites)
                                           releaseDate(i)                                  =
obj.location(activeSites(i)).releaseDate(end);
                                    end

                                    for i = 1:1:length(activeSites)
                                           if
obj.location(activeSites(i)).releaseDate(end)==min(releaseDate)
```

```
                            harvestTarget = activeSites(i);
                        end
                    end

                    harvestB = aggregate(obj,t,'Biomass') - regulation.MTB;;
                    harvestN = round(harvestB /
obj.location(harvestTarget).salmon.w(t));

                    obj.location(harvestTarget) =
harvest(obj.location(harvestTarget),harvestN);
                    obj.location(harvestTarget) =
updateB(obj.location(harvestTarget));

                    LogIndex = length(eventLog)+1;
                    eventLog{LogIndex,1} = t;
                    eventLog{LogIndex,2} = sprintf('Comp. %s harvested (MTB)
site %d on licence %d',obj.companyName,harvestTarget,obj.licenceNumber);

                end

            elseif strcmp(operation,'updateData')
                for i = 1:1:length(obj.location)
```

```
                    obj.location(i) = updatew(obj.location(i));
                    obj.location(i) = updateB(obj.location(i));
                    obj.location(i) = updatePV(obj.location(i));
                    obj.location(i) = getFeedCost(obj.location(i));
                end

            elseif strcmp(operation,'checkLiceLevels')
                for i = 1:1:length(obj.location)
                    obj.location(i) = liceExterminator(obj.location(i));
                end

            end


        end
        %% --------------------------------------------------------------------------
----------------------

    end
```

```
   %% -------------------------------------------------------------------------------------
-------------------

   end % End classdef



   %% -------------------------------------------------------------------------------------
-------------------

   classdef locationObject

   %% -------------------------------------------------------------------------------------
-------------------



       %% Production properties

       properties

              companyName;

              licenceNumber;

              locationNumber;

       end



       %% Production properties

       properties

              releaseDate;                  % Date of releases

              salmon;                               % Locations salmon population
```

```
    end



    %% Decision properties

    properties

        FV;                                 % Forward value of fish in the pen

        PV;                                 % Present value of fish in the pen

    end



    %% Economic properties

    properties

        revenue;                            % Discounted revenues

        treatmentCost;              % Cost of lice treatment

        feedCost;                           % Cost of salmon feed

        S;                                          % Salmon supply [kg]

    end



  %%  -----------------------------------------------------------------------------------
-------------------



    methods
```

```
        %% Class constructor

        %% -----------------------------------------------------------------------
---------------------

        function obj = locationObject(companyName,licenceNumber,locationNumber)

              global tickRate;

              global simulationLength;



              obj.salmon     = populationObject('salmonFarmed');

              obj.releaseDate = 0;



              obj.companyName = companyName;

              obj.licenceNumber = licenceNumber;

              obj.locationNumber = locationNumber;



              %% Initialize array memory for all major data carrying properties to
reduce compute time

              obj.FV                      ([1:1:tickRate*simulationLength]) = 0;

              obj.PV                      ([1:1:tickRate*simulationLength]) = 0;

              obj.S                       ([1:1:tickRate*simulationLength]) = 0;

              obj.revenue           ([1:1:tickRate*simulationLength]) = 0;
```

```
            obj.feedCost          ([1:1:tickRate*simulationLength]) = 0;

            obj.treatmentCost     ([1:1:tickRate*simulationLength]) = 0;

        end

        %% -------------------------------------------------------------------------
---------------------


        %% Function applies impact of lice to weight curve

        %% -------------------------------------------------------------------------
---------------------

        function obj = updatew(obj)

            global environment;

            global t;

            global tickRate;


            currentLiceLevel                                                      =
environment.lice.N(t)/(environment.salmonFarmed.N(t)+environment.salmonWild.N(t));

            startLiceLevel                                                        =
environment.lice.N(1)/(environment.salmonFarmed.N(1)+environment.salmonWild.N(1));


            if currentLiceLevel > startLiceLevel

                obj.salmon.w(t:end)   =   obj.salmon.w(t:end)   *   (1-(0.1   *
currentLiceLevel)/tickRate);
```

```
                       end

               end

               %% --------------------------------------------------------------------------
----------------------



               %% Delousing function

               %% --------------------------------------------------------------------------
----------------------

               function obj = liceExterminator(obj)

                       global environment;

                       global regulation;

                       global market;

                       global t;

                       global eventLog;

                       global tickRate;



                       global freerider;

                       %freerider = 'A';



                       currentLiceLevel                                                     =
environment.lice.N(t)/(environment.salmonFarmed.N(t)+environment.salmonWild.N(t));
```

```
            %if  strcmp(freerider,obj.companyName)  % Do nothing if 'Free rider'

            if environment.salmonWild.speciesData.X(t) >=
regulation.liceLevelYellow

                liceToRemove = (regulation.liceLevelYellow -
regulation.liceLevelGreen)*obj.salmon.N(t);
                environment.lice.N(t+1) = environment.lice.N(t+1) -
liceToRemove;

                obj.treatmentCost(t) = market.treatmentPrice * liceToRemove;
                obj.treatmentCost = obj.treatmentCost ./ ( (1 + market.rate)
.^[1/tickRate:1/tickRate:length(obj.treatmentCost)/tickRate] );

                LogIndex = length(eventLog)+1;
                eventLog{LogIndex,1} = t;
                eventLog{LogIndex,2} = sprintf('Comp. %s removed lice on site
%d, licence %d',obj.companyName,obj.locationNumber,obj.licenceNumber);
```

```
                      elseif isempty(freerider) %Current regime

                             currentLiceLevel > regulation.maxLiceLevel

                             liceOnLocation = obj.salmon.N(t) * currentLiceLevel;


                             maxAllowedLice = obj.salmon.N(t) * regulation.maxLiceLevel;


                             liceToRemove = liceOnLocation - maxAllowedLice;


                             environment.lice.N(t+1) = environment.lice.N(t+1) - liceToRemove;


                             obj.treatmentCost(t) = market.treatmentPrice * liceToRemove;

                             obj.treatmentCost = obj.treatmentCost ./ ( (1 + market.rate) .^[1/tickRate:1/tickRate:length(obj.treatmentCost)/tickRate] );


                             LogIndex = length(eventLog)+1;

                             eventLog{LogIndex,1} = t;

                             eventLog{LogIndex,2} = sprintf('Comp. %s removed lice on site %d, licence %d',obj.companyName,obj.locationNumber,obj.licenceNumber);

                      end
```

```
        end

        %% ------------------------------------------------------------------------------------------------


        %% Populates the site with a yearclass at a specified time of release

        %% ------------------------------------------------------------------------------------------------

        function obj = addYearclass(obj)

            global t;

            global tickRate;

            fishReleased = 70000;



            for i = 0:1:obj.salmon.speciesData.maxLifeTime

                obj.salmon.w(t + i) = polyval(obj.salmon.speciesData.weightCurve,i/tickRate);

                obj.salmon.N(t + i) = round(fishReleased * exp(-obj.salmon.speciesData.M * (i/tickRate)));

            end

            %% Marks the date of release in the releaseDate property

            obj.releaseDate(length(obj.releaseDate)+1) = t;

        end
```

```
              %% --------------------------------------------------------------------------
----------------------


              %% Harvest a specified amount of fish at a specified time
              %% --------------------------------------------------------------------------
----------------------
              function obj = harvest(obj,harvestAmount)
                     global t;
                     global tickRate;
                     global market;

                     for i = t:1:length(obj.salmon.N);
                            if obj.salmon.N(i) - harvestAmount < 0
                                   obj.salmon.N(i:end) = 0;
                                   obj.salmon.w(i:end) = 0;
                                   break;
                            else
                                   obj.salmon.N(i) = obj.salmon.N(i) - harvestAmount;
                            end
                     end
```

```
                    %% Add harvest to revenue

                    obj.revenue(t)   =   (harvestAmount   *   polyval(market.salmonPrice,
obj.salmon.w(t))) / (1+market.rate)^(t/tickRate);

                    obj.revenue    =    obj.revenue    ./    (    (1    +    market.rate)
.^[1/tickRate:1/tickRate:length(obj.revenue)/tickRate] );



                    %% Add harvest to supply

                    obj.S(t) = harvestAmount * obj.salmon.w(t);

                    market.supply(t) = market.supply(t) + harvestAmount;

             end

             %% -------------------------------------------------------------------------
---------------------



             %% Updates the feed costs based on previous period growth

             %% -------------------------------------------------------------------------
---------------------

             function obj = getFeedCost(obj)

                    global t;

                    global tickRate;

                    global market;



                    if t > 1
```

```
                    deltaGrowth = obj.salmon.w(t) - obj.salmon.w(t-1);

                    obj.feedCost(t) = deltaGrowth*obj.salmon.N(t)*market.feedPrice;

                    obj.feedCost   =   obj.feedCost   ./   (   (1   +   market.rate)
.^[1/tickRate:1/tickRate:length(obj.feedCost)/tickRate] );

               end

               %obj.salmon.B = obj.salmon.N .* obj.salmon.w;

          end

          %% ------------------------------------------------------------------------
---------------------


          %% Updates the biomass records for the entire site

          %% ------------------------------------------------------------------------
---------------------

          function obj = updateB(obj)

               obj.salmon.B = obj.salmon.N .* obj.salmon.w;

          end

          %% ------------------------------------------------------------------------
---------------------


          %% Updates the PV and FV records for the entire site

          %% ------------------------------------------------------------------------
---------------------
```

```
            function obj = updatePV(obj)

                   global tickRate;

                   global market;




                   %% Calculate PV based on price

                   price = [1 0];

                   obj.FV = obj.salmon.N .* polyval(price,obj.salmon.w);



                   % Discount FV by annual compound rate, extrapolate to all data points

                   obj.PV      =      obj.FV      ./      (      (1      +      market.rate)
.^[1/tickRate:1/tickRate:length(obj.FV)/tickRate] );

            end

            %% -------------------------------------------------------------------------------------
---------------------



      end

   %%  -------------------------------------------------------------------------------------------
-------------------

   end % End classdef
```

```
   %% -------------------------------------------------------------------------------------
-------------------

   classdef regulationObject

   %% -------------------------------------------------------------------------------------
-------------------


       properties

              changeRate;                  % Increase/Decrease rate from decision rule

              minResetPeriod;              % Required minimum period between production runs
on locations

              MTB;                         % Max allowed biomass

              maxLiceLevel;         % Max allowed lice level

              liceLevelGreen;

              liceLevelYellow;

              liceLevelRed;

       end


   %% -------------------------------------------------------------------------------------
-------------------


       methods
```

```
              %% Class constructor
              %% ---------------------------------------------------------------------------
----------------------
              function obj = regulationObject
                     global tickRate;
                     obj.changeRate = 0.05;
                     obj.minResetPeriod = ceil(2/12*tickRate);
                     obj.MTB = 680*10^3;
                     obj.maxLiceLevel = 0.5;
                     obj.liceLevelGreen = 0.1; %Below
                     obj.liceLevelYellow = 0.3; %Below
                     obj.liceLevelRed = 0.3; %Above
              end

       end

   %%  -------------------------------------------------------------------------------------
-------------------
   end % End classdef
```

```
   %% -------------------------------------------------------------------------------------
-------------------

   classdef salmonWildObject

   %% -------------------------------------------------------------------------------------
-------------------



       %% Species related properties

       properties

              name;

              I;                                  % Number of natural occuring individuals
in environment

              M;                                  % Natural mortality rate

              R;                                  % Natural reproduction rate

              X;                                  % Increase in mortality rate

       end



   %% -------------------------------------------------------------------------------------
-------------------



       methods



              %% Class constructor
```

```
              %% --------------------------------------------------------------------------
----------------------

              function obj = salmonWildObject

                      global tickRate;

                      global simulationLength;



                      obj.name = 'Wild Salmon';

                      obj.I = 1000000; % Must be 10 times greater than I for lice in the
model for equlibirum

                      obj.M = 0.1;

                      obj.R = 0.1 + obj.M;

                      obj.X  ([1:1:tickRate*simulationLength]) = 0;

              end



       end



   %%  -------------------------------------------------------------------------------------
-------------------

   end % End classdef



   %%  -------------------------------------------------------------------------------------
-------------------
```

```
   %%  This object represents an environment, and holds all species populations in that
environment
   %%  In addition it can have factors such as
   %% -------------------------------------------------------------------------------------
-------------------
   classdef environmentObject
   %% -------------------------------------------------------------------------------------
-------------------


       %% Populations
       properties
              lice;
              salmonFarmed;
              salmonWild;
       end


   %% -------------------------------------------------------------------------------------
-------------------


       methods


              %% Class constructor
```

```
        %% --------------------------------------------------------------------------
----------------------

        function obj = environmentObject

            %% Initialize populations

            obj.lice                            = populationObject('lice');

            obj.salmonFarmed             = populationObject('salmonFarmed');

            obj.salmonWild               = populationObject('salmonWild');

        end

        %% --------------------------------------------------------------------------
----------------------



        %% Function that controls the populations biomass at each time

        %% --------------------------------------------------------------------------
----------------------

        function obj = updatePopulationB(obj,time,biomass,species)



            if strcmp(species,'lice')

            elseif strcmp(species,'salmonFarmed')

                obj.salmonFarmed.B(time) = obj.salmonFarmed.B(time) + biomass;

            elseif strcmp(species,'salmonWild')

            end
```

```
        end

        %% --------------------------------------------------------------------------
---------------------



        %% Function sums up farmed biomass from all locations

        %% --------------------------------------------------------------------------
---------------------

        function obj = updateSalmonFarmed(obj,biomass,population)

            global t;

            obj.salmonFarmed.B(t) = obj.salmonFarmed.B(t) + biomass;

            obj.salmonFarmed.N(t) = obj.salmonFarmed.N(t) + population;

        end

        %% --------------------------------------------------------------------------
---------------------



        %% Function that accounts for impact of lice on wild salmon

        %% --------------------------------------------------------------------------
---------------------

        function obj = updateSalmonWild(obj)

            global t;

            global tickRate;
```

```
                    if t > 1

                            obj.salmonWild.speciesData.X(t)        =        (obj.lice.N(t-
1)/(obj.salmonWild.N(t-1)+obj.salmonFarmed.N(t-1)));

                            salmonWildGrowthRate = -(obj.lice.N(t-1)/(obj.salmonWild.N(t-1)
+ obj.salmonFarmed.N(t-1))) - obj.salmonWild.speciesData.M + obj.salmonWild.speciesData.R;

                            obj.salmonWild.N(t)            =            obj.salmonWild.N(t-
1)*exp(salmonWildGrowthRate/tickRate);

                    else

                            obj.salmonWild.N(t) = obj.salmonWild.speciesData.I;

                    end

            end

            %% --------------------------------------------------------------------------
----------------------



            %% Update lice population for the next tick

            %% --------------------------------------------------------------------------
----------------------

            function obj = updateLice(obj)

                    global t;

                    global tickRate;

                    modifier = 20; % Relative impact modifier for farmed vs wild salmon on
lice growth
```

```
                if t > 1
                    liceGrowthFactor = -obj.lice.speciesData.M + (obj.salmonWild.N(t-1)+modifier*obj.salmonFarmed.N(t-1))/(100*obj.lice.N(t-1));
                    if obj.lice.N(t) + obj.lice.N(t-1)*exp(liceGrowthFactor/tickRate) < obj.lice.speciesData.I;
                        obj.lice.N(t) = obj.lice.speciesData.I;
                    else
                        obj.lice.N(t) = obj.lice.N(t) + obj.lice.N(t-1)*exp(liceGrowthFactor/tickRate);
                    end
                else
                    obj.lice.N(t) = obj.lice.speciesData.I;
                end
            end
            %% ------------------------------------------------------------------------------------------------


        end


    %% --------------------------------------------------------------------------------------------------------

    end % End classdef
```

```
   %%  -------------------------------------------------------------------------------------
-------------------

   classdef liceObject

   %%  -------------------------------------------------------------------------------------
-------------------


       %% Inherrited model properties

       properties

               name;

               I;                                  % Number of natural occuring individuals
in environment

               M;                                  % Natural mortality rate

               R;                                  % Natural reproduction rate

       end


   %%  -------------------------------------------------------------------------------------
-------------------


       methods


               %% Class constructor
```

```
               %% ----------------------------------------------------------------------------
----------------------

               function obj = liceObject

                      obj.name = 'Sea Lice';

                      obj.I = 100000; % Must be 10 times less than I for lice in the model
for equlibirum

                      obj.M = 0.1;

                      obj.R = 0.1;

               end


       end


   %%  -------------------------------------------------------------------------------------
-------------------

   end % End classdef


   function intitalize


       clear *

       clear global

       clc
```

```
        %%  Model options
        %% --------------------------------------------------------------------------------------------------------
        global modelDebug;
        global plotting;
        global eventLog;
        global freerider;
        modelDebug = 'normal';
        plotting = 0;


        %%  Splash screen info
        %% --------------------------------------------------------------------------------------------------------

        fprintf('------------------------------------------------------------------------------\n');
        fprintf(' Version 1.2.0\n');
        fprintf('------------------------------------------------------------------------------\n\n');
```

```
       fprintf('  This model simulates the impact of sea lice on wild and farmed\n  salmon
in an isolated region.\n\n');

       fprintf('  Two company objects represents the fish farmers in the region.\n\n');

       fprintf('  "Company A" controls X/10 of total regional production volume.\n  "Company
B" represents all firms in the region by controlling\n  the rest of production volume\n\n');

       fprintf('---------------------------------------------------------------------------
---\n\n');

       input('Press enter to run model ');

       clc



       fprintf('---------------------------------------------------------------------------
---\n');

       fprintf('\tInitializing..\n');

       fprintf('---------------------------------------------------------------------------
---\n\n');



       %%  Setting model input parameters

       %% ---------------------------------------------------------------------------------
-----------------------

       global tickRate;
```

```
    global simulationLength;

    while 1
        userInput = input('Input simulation length in years [default = 30] (press enter for default)\n >');
        if isempty(userInput)
            simulationLength = 30;
            clear userInput;
            break;
        elseif isnumeric(userInput)
            simulationLength = userInput;
            clear userInput;
            break;
        else
            fprintf('ERROR: Invald input\n');
            clear userInput
        end
    end

    while 1
```

```
        userInput = input('\nInput tick rate ''day'',''week'',''month'' [default = ''month'']\nRemember the '''' marks (or press enter for default)\n >');

        if isempty(userInput)

            tickRate = 12;

            clear userInput;

            break;

        elseif strcmp(userInput,'day')

            tickRate = 365;

            clear userInput;

            break;

        elseif strcmp(userInput,'week')

            tickRate = 52;

            clear userInput;

            break;

        elseif strcmp(userInput,'month')

            tickRate = 12;

            clear userInput;

            break;

        else

            fprintf('ERROR: Invalid input\n');
```

```
                        clear userInput
                end
        end


        while 1
                userInput = input('\nInput marketshare X for "Company A" [default = 1] (press
enter for default)\n >');
                if isempty(userInput)
                        marketShare = 1;
                        clear userInput;
                        break;
                elseif isnumeric(userInput)
                        marketShare = userInput;
                        clear userInput;
                        break;
                else
                        fprintf('ERROR: Invald input\n');
                        clear userInput
                end
        end
```

```
       while 1

              userInput = input('\nEnter ''n'' to turn off ''free rider mode'' for company
A \n(or press enter to continue)\n >');

              if isempty(userInput)

                     freerider = 'A';

                     clear userInput;

                     break;

              else

                     freerider = [];

                     clear userInput;

                     break;

              end

       end



       %%  Creating model objects

       %% ---------------------------------------------------------------------------------
-----------------------



       %%     Create environment

       global environment;
```

```
environment = environmentObject;



%%      Create regulations

global regulation;

regulation = regulationObject;



%% Create market

global market;

market = marketObject;



%%      Create companies

global fishery;

fishery = companyObject('A');

fishery(2) = companyObject('B');



%%      Distribute market share between companies

for i = 1:10

        if i <= marketShare fishery(1) = addLicence(fishery(1)); end

        if i > marketShare fishery(2) = addLicence(fishery(2)); end

end
```

```
    eventLog{1,1} = 1;

    eventLog{1,2} = '...';

    eventLog{2,1} = 1;

    eventLog{2,2} = '...';



end



function logger

    global modelDebug;

    global tickRate;

    global simulationLength;

    global t;

    global fishery;

    global environment;

    global eventLog;





    if strcmp(modelDebug,'verbose');

    elseif strcmp(modelDebug,'normal');
```

```
              clc

              if tickRate == 12

                     timeunit = 'month';

              elseif tickRate == 52

                     timeunit = 'week';

              elseif tickRate == 365

                     timeunit = 'day';

              end



              fprintf('--------------------------------------------------------------------
----------\n');

              fprintf(' Simulating..\n');

              fprintf('--------------------------------------------------------------------
----------\n');

              fprintf('> Year %.0f, %s %.0f\n',t/tickRate,timeunit,mod(t-1,tickRate)+1);

              fprintf('--------------------------------------------------------------------
----------\n\n');



              MarketSize = [length(fishery(1).licence);length(fishery(2).licence)];
```

```
            Revenue                                                          =
[sum(fishery(1).aggregate([1:t],'revenue'))/10^6;sum(fishery(2).aggregate([1:t],'revenue'))
/10^6];

            TreatmentCost                                                    =
[sum(fishery(1).aggregate([1:t],'TreatmentCost'))/10^6;sum(fishery(2).aggregate([1:t],'Trea
tmentCost'))/10^6];

            FeedCost                                                         =
[sum(fishery(1).aggregate([1:t],'FeedCost'))/10^6;sum(fishery(2).aggregate([1:t],'FeedCost'
))/10^6];

            Profit = Revenue - TreatmentCost - FeedCost;



            fprintf('Company\t\t\t\tA\t\t\tB\t\n');

            fprintf('---------------------------------------------------------------------
----------\n');



            fprintf(' Production licences \t%d\t\t\t%d\n\n',MarketSize(1),MarketSize(2));

            fprintf('                         +                          Revenue
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n',Revenue(1),Revenue(1)/sum(Revenue)*100,Revenue(2),Reven
ue(2)/sum(Revenue)*100);

            fprintf('                                                    Salmon
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n',Revenue(1),Revenue(1)/sum(Revenue)*100,Revenue(2),Reven
ue(2)/sum(Revenue)*100);

            fprintf('                         -                          Costs
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n',FeedCost(1)+TreatmentCost(1),(FeedCost(1)+TreatmentCost
```

```
(1))/(sum(FeedCost)+sum(TreatmentCost))*100,FeedCost(2)+TreatmentCost(2),(FeedCost(2)+Treat
mentCost(2))/(sum(FeedCost)+sum(TreatmentCost))*100);

              fprintf(' Feed
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n',FeedCost(1),FeedCost(1)/sum(FeedCost)*100,FeedCost(2),F
eedCost(2)/sum(FeedCost)*100);

              fprintf(' Delousing
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n',TreatmentCost(1),TreatmentCost(1)/sum(TreatmentCost)*10
0,TreatmentCost(2),TreatmentCost(2)/sum(TreatmentCost)*100);

              fprintf(' = Profit
\t\t%.2f\t%.0f%%\t\t%.2f\t%.0f%%\n\n\n',Profit(1),Profit(1)/sum(Profit)*100,Profit(2),Profi
t(2)/sum(Profit)*100);

              %fprintf('Lice per
fish\t\t\t%.2f\n',environment.lice.N(t)/(environment.salmonFarmed.N(t)+environment.salmonWi
ld.N(t)));

              %fprintf('Increase wild fish mortality
\t%.2f\n',environment.salmonWild.speciesData.X(t));



              fprintf('--------------------------------------------------------------------
----------\n\n');



              %fprintf('\n-----------------------------------------------------------------
-------------\n');

              %fprintf(' Event log:\n');

              %fprintf('-------------------------------------------------------------------
-----------\n');
```

```
              showMaxEvents = 15;

              a  =  3  +  (length(eventLog(:,1))>=showMaxEvents+3)*(length(eventLog(:,1))-
showMaxEvents-2);

              for i = a:1:length(eventLog(:,1))

                     %fprintf('Year          %.0f,          %s          %.0f          |
%s\n',eventLog{i,1}/tickRate,timeunit,mod(eventLog{i,1}-1,tickRate)+1,eventLog{i,2});

              end


       else


       end


       fprintf('\n\n');


   end


   %% ----------------------------------------------------------------------------------
------------------

   classdef marketObject

   %% ----------------------------------------------------------------------------------
------------------
```

```
       properties

              rate;                         % Market interest rate

              supply;                       % Market supply

              MTB;                          % Max allowed biomass

              salmonPrice;           % Market salmon price

              treatmentPrice;               % Market unit price for lice treatment

              feedPrice;                    % Market unit price for feed

       end


   %% -------------------------------------------------------------------------------------
-------------------


       methods


              %% Class constructor

              %% --------------------------------------------------------------------------
----------------------

              function obj = marketObject

                     global tickRate;

                     global simulationLength;
```

```
                      obj.rate = 0.05;

                      obj.salmonPrice = [1 0];

                      obj.feedPrice = 0.8;

                      obj.treatmentPrice = 10;

                      obj.supply([1:1:tickRate*simulationLength]) = 0;

              end



       end



   %% -------------------------------------------------------------------------------------
-------------------

   end % End classdef




   %% -------------------------------------------------------------------------------------
-------------------

   classdef salmonFarmedObject

   %% -------------------------------------------------------------------------------------
-------------------



       %% Inherrited model properties
```

```
       properties

               name;

               I;                                 % Number of natural occuring individuals
in environment

               M;                                 % Natural mortality rate

               R;                                 % Natural reproduction rate

               weightCurve;          % Weight curve

               maxLifeTime;          % Lifespan of species

       end


   %%  -------------------------------------------------------------------------------------
-------------------


       methods


               %% Class constructor

               %% --------------------------------------------------------------------------
----------------------

               function obj = salmonFarmedObject

                       global tickRate;
```

```
                      obj.name = 'Farmed salmon';

                      obj.I = 0;

                      obj.M = 0.1;

                      obj.R = 0;

                      obj.weightCurve = [-2.08 5.72 0 0];

                      obj.maxLifeTime = 2.5*tickRate;

              end


       end


   %% -------------------------------------------------------------------------------------
-------------------

   end % End classdef



   %% -------------------------------------------------------------------------------------
-------------------

   %%  This object hold population parameters such as number of individuals, weight etc.

   %%  The species specific parameters are sourced from species related objects during the
construction

   %% -------------------------------------------------------------------------------------
-------------------
```

```
   classdef populationObject

   %% -------------------------------------------------------------------------------------
-------------------

       %% Population properties

       properties

              speciesData;          % Species related parameters

              w;                                  % Weight

              N;                                  % Number of individuals

              B;                                  % Biomass

       end



   %% -------------------------------------------------------------------------------------
-------------------



       methods



              %% Class constructor

              %% --------------------------------------------------------------------------
----------------------

              function obj = populationObject(species);
```

```
        global tickRate;
        global simulationLength;

        %% Set population species
        if strcmp(species,'lice')
            obj.speciesData = liceObject;
        elseif strcmp(species,'salmonFarmed')
            obj.speciesData = salmonFarmedObject;
        elseif strcmp(species,'salmonWild')
            obj.speciesData = salmonWildObject;
        end

        %% Initialize array memory for all major data carrying properties to reduce compute time
        obj.N  ([1:1:tickRate*simulationLength]) = 0;
        obj.w  ([1:1:tickRate*simulationLength]) = 0;
        obj.B  ([1:1:tickRate*simulationLength]) = 0;

    end
```

```
        end

    %% -------------------------------------------------------------------------------------
-------------------

    end % End classdef
```