Journal of Applied Finance & Banking, Vol. 10, No. 4, 2020, 127-156
ISSN: 1792-6580 (print version), 1792-6599(online)
Scientific Press International Limited

Risk/Return/Retention Efficient Frontier Discovery Through Evolutionary Optimization For Non-Life Insurance Portfolio

Andrea Riva[1]

Abstract

Policyholder capability to easily and promptly change their insurance cover, in terms of contract conditions and provider, has substantially increased during last decades due to high market competency levels and favourable regulations. Consequently, policyholder behaviour modelling acquired increasing attention since being able to predict costumer reaction to future market's fluctuations and company's decision achieved a pivotal role within most mature insurance markets. Integrating existing modelling platform with policyholder behavioural predictions allows companies to create synthetic responding environments where several market projections and company's strategies can be simulated and, through sets of defined objective functions, compared. In this way, companies are able to identify optimal strategies by means of a Multi-Objective optimization problem where the ultimate goal is to approximate the entire set of optimal solutions defining the so-called Pareto Efficient Frontier. This paper aims to demonstrate how meta-heuristic search algorithms can be promptly implemented to tackle actuarial optimization problems such as the renewal of non-life policies. An evolutionary inspired search algorithm is proposed and compared to a Uniform Monte Carlo Search. Several numerical experiments show that the proposed evolutionary algorithm substantially and consistently outperforms the Monte Carlo Search providing faster convergence and higher frontier approximations.

Keywords: Policyholder behaviour, portfolio optimization, renewal price, evolutionary computation, multi-objective optimization, differential evolution, Monte Carlo optimization.

---

[1] Department of Statistics, La Sapienza University of Rome, Italy.

Article Info: *Received:* January 19, 2020. *Revised:* March 6, 2020.
Published online: May 1, 2020.

1. Introduction

During the last decades, policyholder behaviour modelling becomes one of the main areas of interest for both life and general insurance companies. Within a highly competitive market, a pricing model that do not consider the policyholder's probability to accept a given quotation could be affected by a fundamental bias preventing the company to elaborate accurate portfolio projections and profitability analysis. Web platforms that allow potential customers to easily compare different quotations as well as the introduction of Solvency II framework[2], raised the pivotal role of policyholder behaviour modelling inducing an increase of attention within the actuarial field.

Fuel by an increasing interest of actuarial practitioners in machine learning, researchers [1], [2] have mainly focused on modelling policyholder behaviour as a supervised binary classification problem in which prediction accuracy represents the ultimate objective.

Being able to predict with great accuracy policyholder behaviour is critical for an insurance company but from a practical point of view, it is also crucial to know how to optimally use these models to reach strategy goals. Solvency regulation, high market competition and shareholder requirements define an environment in which each strategy needs to balance a complex set of different objectives.

Combining several models (e.g. pricing and policyholder behaviour) in a single platform enables companies to create a synthetic responding environment allowing to simulate the effects of different strategies. This modelling platform can be represented in a three pillars architecture defined by a Company Actions Modelling which specifies what the insurer can do, an Environment Reaction Modelling that represents how the environment could react to the insurer's actions and finally, a set of Objective Functions which measure company induced changes in the environment.

Through this structure, companies can simulate different strategies and compare their results based on the selected objective functions creating a preference structure between strategies. Given two different strategies, typically one dominates the other if it is at least better in one objective function and equal in all the other. Strategies that are not dominated by any other are called efficient and define the so-called Pareto Efficient Frontier. When comparing different strategies, companies need to consider only those belonging to the Pareto Frontier. Evaluate all possible strategies is usually computational infeasible, therefore search algorithms can be deployed to approximate the Pareto Set. Several optimization techniques are available in the literature, however classical mathematical approaches may prove to be inadequate whereas the specific model complexity is high. In this paper, we will show how numerical optimization techniques can be effortlessly deployed to tackle an actuarial optimization problem without being affected by the underlying model complexity.

---

[2] Within Solvency II Framework, Lapse Risk often represents the greatest non-market risk for a life insurance company ([3]).

Specifically, the aim of this paper is to apply an evolutionary inspired multi-objective optimization algorithm to the general insurance portfolio renewal problem. Given a set of insurance contracts the insurer will need to choose to which policyholder offer an insurance cover as well as the associated renewal price. Therefore, a combined pricing and policyholder behaviour model will be used as a synthetic environment in which each policyholder decides to accept or not the proposed quotation. Finally, the objective functions will be defined as the total portfolio premium; total portfolio Tail Value at Risk and total portfolio retention. Therefore, the optimization search will need to approximate a three-dimensional Pareto Frontier in which each point represents a portfolio selection and a renewal price strategy. Algorithm's performance will be measured by the quality of the approximated Pareto Frontier and will be compared with a uniform Monte Carlo search for different portfolios and market competition levels.

To the author's knowledge, an application of Evolutionary Multi-Objective Optimization algorithm to the non-life renewal pricing problem, specifically on three-dimensional objectives functions, is still lacking in the literature and hence will be presented here.

The rest of this paper is organized as follow: Section 2 provides a literature review on policyholder behaviour modelling and portfolio renewal optimization. Methodological approach, such as problem formalization and search algorithms will be presented in Section 3. Following section reports results of extensive simulation experiments designed to fairly asses performances of the proposed algorithms whose parameterization details are showed in the appendix. Finally, Section 5 concludes the paper.

2. Related Literature

In the last decades, actuarial literature has been featured by an ever increasing interest on policyholder behavioural modelling by both academic and practitioner actuaries [1],[2],[3],[4],[5],[6]. Highly competitive markets and favourable regulation [7],[8] substantially increased policyholder capability to easily and promptly change their insurance cover both in terms of contract conditions and provider.

From its introduction in 2016, Solvency II framework highlighted how policyholder massive surrender activities has become the greatest non-financial risk to which life insurance companies are exposed [9]. From general insurance's perspective, the Casualty Actuarial Society defines pricing optimization as the "supplementation of traditional supply-side actuarial models with quantitative customer demand models. This supplementation takes place through a mathematical process used to determine the prices that best balance supply and demand in order to achieve user-defined business goals while simultaneously imposing business or regulatory limitations on how those goals are achieved. The end result is a set of proposed adjustments to the cost models by customer segment for actuarial risk classes" [10].

Therefore, to predict how customers would react to both external market fluctuations and internal company decision is a significant component of modern actuarial modelling. By this end, researchers [1],[2] studied how modern machine learning techniques are particularly suitable for these tasks when compared to more classical binomial GLM.

Although high predictive accuracy is critical, very few studies on how companies should operate on the basis of these modelling insight have been carried out. Indeed, even for a company capable to perfectly predict policyholder's reaction to any situation, further quantitative tools would be necessary to realize which set of decisions would optimally drive the insurer towards its strategy target.

Therefore, on top of prediction modelling, optimization problems that focus on defining which actions an insurer should execute to reach its strategy goal can be formalized. Several studies on the renewal optimization problem can be found in actuarial literature [11], [12], among those, [6] proposed an optimization framework, built upon a pricing and policyholder behavioural model, whose ultimate goal is to discover the optimal renewal strategies under a total retention constraint.

Rather than finding an optimal solution conditioned to some constraint, an alternative optimization approach based on multi-objective search techniques would strive to approximate the entire Efficient Frontier. Because of its built-in capability to simultaneously deal with multiple candidate solutions, which is particular suitable on a multi-objective optimization problem where there is not a unique solution, evolutionary computation [13] represents a promising toolbox to deal with these type of problems. Although rarely addressed, some application of Evolutionary Computation can be found in actuarial literature [14],[15],[16],[17],[18]. A recent survey presented by the Society of Actuaries [14] on emerging data analytics techniques explicitly references to possible applications of Genetic Algorithm [19] in actuarial science demonstrating an increasing interest on Evolutionary Computation applications to both insurance and finance sectors.

3. Methodological Approach

3.1 Problem Formalization

Consider an insurance company that holds a portfolio of m contracts at a given valuation date. Each contract is assumed to be statistically independent from the others and its own risk is fully described by frequency and severity distributions.

At the evaluation date, the company needs to select:

1. which contracts retain for the following covering period;
2. which renewal price offers to those contracts that it wants to retain.

We consider an insurance market with different competitors, therefore a policyholder could decide to change insurer by not accepting the quotation offered by the company. Furthermore, if the insurer has internally modelled the policyholder behaviour, for a specific policyholder's risk profile and the proposed quotation, there exists an expected acceptance probability available to the company. Intuitively, increasing the renewal price will lead to a greater revenue for the insurer, however this could also result in a loss of costumers that decide to terminate their contracts. At the same time, under Solvency II framework insurer needs to consider the capital requirement associated to a given portfolio, then it is critical to analyze the risk profile of each potential customer as well as the diversification achievable for a given portfolio.

To formalize this problem, we follow a classic approach in general insurance and we assume that each contract $i = 1, \dots, m$ is defined by the following distributional structure[3]:

- $\tilde{N}_i \sim Poi(\lambda_i)$ describe the claim frequency[4], where $\lambda_i > 0$ represents the distribution mean and variance;
- $\tilde{Z}_{j,i} \sim \Lambda(\mu_i, \sigma_i)$ describe the claim severity, with $\mu_i \geq 0$ and $\sigma_i > 0$ representing respectively the distribution position and diffusion parameters;
- The random variables (r.v.s) $\tilde{Z}_{1,i}, \dots, \tilde{Z}_{\tilde{N}_i,i}$ are statistically independent and identical distributed;
- The r.v.s. $\tilde{N}_i$ and $\tilde{Z}_{j,i}$ are statistically independent;
- $\tilde{L}_i = \sum_{j=1}^{\tilde{N}_i} \tilde{Z}_{j,i}$ describe the aggregate loss.

Furthermore, we assume that the fair quotation for a single contract is simply defined by the product between expected claim frequency and expected claim severity.

$$P_i = E(\tilde{N}_i)E(\tilde{Z}_{j,i})$$

The renewal price offered by the company can be represented as:

$$P_i^* = P_i \alpha_i$$

where α_i represents a renewal multiplication factor, if $\alpha_i > 1$ it means that the company is requiring a greater premium.

Intuitively, a customer will be less prone to accept the insurance cover if $\alpha_i > 1$, even with $\alpha_i = 1$ the policyholder could decide to change insurer in a highly competitive market.

Let's assume that the company has modelled[5] the probability of a customer to

---

[3] Throughout this paper, we use a ~ (tilde) hat to identify random variables.

[4] As widely addressed by actuarial literature, classical Poisson distribution could provide unreliable claim frequency modelling especially on portfolios featured by empirical over-dispersion, therefore over-dispersed Poisson assumption is usually preferred. Since both optimization algorithms' dynamics are not affected by the underlying pricing model's structure, we choose the classical Poisson assumption to ease some computational burden in the simulation experiments.

[5] The proposed policyholder behaviour modelling is clearly extendable both in term of input variables, such as individual client information and market competency level, and functional form.

accept a given quotation as:

$$\hat{\rho}_i = 1 - \frac{e^{\theta_i}}{1 + e^{\theta_i}}$$
$$\theta_i = \beta_0 + \beta_1 P_i + \beta_2 \alpha_i$$

A given parameter calibration will model the sensitivity of a specific customer to the renewal price offered by the company which ultimately reflects the level of competition in the insurance market.
Considering the entire portfolio of m contracts, a selection/renewal strategy could be compactly represented by a $m \times 2$ matrix[6] $\boldsymbol{X}$ in which each row is defined by a binary selector h_i, that represents if the company wants to retain the contract for the following period, and the eventual renewal multiplication factor α_i. Hence, a selection/renewal strategy is define as $\boldsymbol{X} = (H, A)$ with $H = [h_1, \dots, h_m]$ and $A = [\alpha_1, \dots, \alpha_m]$. It is worth pointing out that the renewal factor is automatically set to zero for those contracts that the insurer does not want to retain.
Considering the probability structure defined so far, for each realization of $\boldsymbol{X}$ it is then possible to generate S stochastically independent simulations for each contract to evaluate:

1. Aggregate claims cost $\tilde{L}_i = \sum_{j=1}^{\tilde{N}_i} \tilde{Z}_{j,i}$
2. Policyholder behaviour $\tilde{B}_i \sim Bernoulli(\hat{\rho}_i)$

These simulations can be compactly stored in a $m \times S \times 2$ tensor $\boldsymbol{T}|\boldsymbol{X} = (\boldsymbol{L}, \boldsymbol{B}|\boldsymbol{X})$ where $\boldsymbol{L}$ represents a $m \times S$ matrix containing the realizations of the aggregate cost $\tilde{L}$ for each contract and simulation, while $\boldsymbol{B}|\boldsymbol{X}$ is a $m \times S$ binary matrix that represents for each policyholder and simulation if the proposed quotation has been accepted. Notice that the aggregate loss for each contract is not affected by the renewal strategy therefore is independent by $\boldsymbol{X}$ and can be simulated only once at the beginning of the optimization process.
Assuming that each contract is statistically independent from the other, it is possible to exploit simulations to evaluate the distribution of the aggregated loss at portfolio level conditioned to $\boldsymbol{X}$ as:

$$L|\boldsymbol{X} = (\boldsymbol{L} \odot \boldsymbol{B}|\boldsymbol{X})^T\, H$$

where $\odot$ represents the Hadamard product between two matrices and $H = [h_i]_{i=1,\dots,m}$ represents the first column of $\boldsymbol{X}$ (the binary selector), then $L|\boldsymbol{X}$ is a $S \times 1$ vector containing the simulated portfolio aggregated loss that can be used to

---

Likewise the pricing model, policyholder behaviour model's underlying structure does not affect the optimization algorithms' dynamics. Therefore, a simple GLM modelling has been choose to ease some computational burden in the simulation experiments.

[6] Throughout this paper, matrix are denoted in **bold**.

estimate the following risk metric[7]:

$$f_1(\boldsymbol{X}) = -TVaR_\omega(L|\boldsymbol{X}) = -\left(p_\omega(L|\boldsymbol{X}) - E(L|\boldsymbol{X})\right)$$

where $p_\omega(L|\boldsymbol{X})$ represents the ω-quantile of $L|\boldsymbol{X}$.

The retention metric will be evaluated as:

$$f_2(\boldsymbol{X}) = \frac{1}{\sum_{i=1}^{m} h_i} \sum_{i=1}^{m} \hat{\rho}_i$$

Intuitively $f_2(\boldsymbol{X}) \in [0,1]$ defines an aggregate retention score of a given renewal policy $\boldsymbol{X}$, if the acceptance probabilities are high then the sum of those probabilities will be close to the total number of contracts that the company decides to retain under $\boldsymbol{X}$.

Finally, portfolio revenue[8] will be measured as:

$$f_3(\boldsymbol{X}) = \sum_{i=1}^{m} P_i \alpha_i \hat{\rho}_i$$

where α_i and $\hat{\rho}_i$are automatically set to zero for all non selected contracts.

These three metrics will be adopted to evaluate each selection/renewal strategy $\boldsymbol{X}$ allowing to compare different strategies with the following preference structure:

$$\begin{gathered} \boldsymbol{X}_A \succ \boldsymbol{X}_B \; if \; f_1(\boldsymbol{X}_A) > f_1(\boldsymbol{X}_B) \wedge f_2(\boldsymbol{X}_A) \geq f_2(\boldsymbol{X}_B) \wedge f_3(\boldsymbol{X}_A) \geq f_3(\boldsymbol{X}_B) \; or \\ if \; f_1(\boldsymbol{X}_{\boldsymbol{A}}) \geq f_1(\boldsymbol{X}_B) \wedge f_2(\boldsymbol{X}_A) > f_2(X_B) \wedge f_3(\boldsymbol{X}_A) \geq f_3(\boldsymbol{X}_B) \; or \\ f_1(\boldsymbol{X}_A) \geq f_1(\boldsymbol{X}_B) \wedge f_2(\boldsymbol{X}_A) \geq f_2(\boldsymbol{X}_B) \wedge f_3(\boldsymbol{X}_A) > f_3(\boldsymbol{X}_B) \end{gathered}$$

Strategy $\boldsymbol{X}_A$ dominates $\boldsymbol{X}_B$ if it is at least better in one objective function and equal in all the others, strategies that are not dominated by any other are called *efficient* and define the so-called Pareto Frontier.

Finally, the multi-objective optimization problem can be formalized as follow:

$$\begin{gathered} \max_{H,A} f_j(H,A) \;\; for \; j = 1, \dots, 3 \\ sub \\ \alpha \in [\alpha_{min}, \alpha_{max}] \end{gathered}$$

$H = [h_1, \dots, h_m]$ and $A = [\alpha_1, \dots, \alpha_m]$ represent respectively the selection and renewal vectors in $\boldsymbol{X}$. Renewal boundaries are represented by $\alpha_{min}, \alpha_{max}$ and define the maximum price increase/discount allowed within a renewal strategy.

---

[7] Considering the negative of the Tail Value at Risk allows to formalize the multi-objective optimization problem as a max-search for all the objective functions.

[8] Safety loading on fair premium could be considered , nonetheless both optimization algorithms' underlying structures would not be affected by this modelling choice.

3.2 Search Algorithm

To tackle the optimization problem formulated in the previous section, two search algorithms have been applied: Uniform Monte Carlo Search (UMCS) ([20]) and Differential Evolution for Multi-Objective Optimization (DEMO) ([21]).
The UMCS approach initially generates a population of P candidate solutions $\boldsymbol{X}_1, \dots, \boldsymbol{X}_P$ where each $\boldsymbol{X}_j$ is generated as follow:

1. Sample $u_j \sim U(0,1)$
2. $H_j = \left[h_1^j, \dots, h_m^j\right]$ where $h_i^j \sim Bernoulli\left(u_j\right)$ for $i = 1, \dots, m$
3. $\mathrm{A}_j = \left[\alpha_1^j, \dots, \alpha_m^j\right]$ where $\alpha_i^j \sim U(\alpha_{min}, \alpha_{max})$ for $i = 1, \dots, m$
4. If $h_i^j = 0$ then $\alpha_i^j = 0$ else do nothing

The first sampling of u_j allows to generate portfolios with a variety number of selected contracts, otherwise the sampling procedure would concentrate on portfolio with approximately $m/2$ selected contracts, preventing a good exploration of the solution space. All candidate solutions are then evaluated and compared to all the other to identify the efficient ones. Finally, the procedure selects only those solutions flagged as *efficient* resulting in the UMCS approximation of the Pareto Frontier.
Although extremely simple, this method can be effortlessly implemented and provide a baseline performance on which compare other search strategies. Being a Monte Carlo Method, the quality of the approximation is mainly determined by the number of simulations run, therefore the dimension of the population P. It is worth notice that, in order to evaluate the Risk metric, for each candidate solution $\boldsymbol{X}_j$ additional simulations of the policyholder behaviour are run since the probability of acceptance of a potential costumer depends on the renewal strategy contained in $\boldsymbol{X}_j$.

Therefore, the total number of simulations run by the procedure is $S \times P$.

Algorithm 1: UMCS

Input

- $P\ population\ dimension$
- $Seed\ random\ number\ generator\ seed$

Monte Carlo Search

- $Set\ Seed$
- $Randomly\ create\ population\ \boldsymbol{X}_{Pop} = \{\boldsymbol{X}_1, \dots, \boldsymbol{X}_P\}$
- $\overline{\boldsymbol{A}} = FindFrontier\left(\boldsymbol{X}_{Pop}\right)$

Output

- $Return\ \overline{\boldsymbol{A}}$

where function *FindFrontier* filters the efficient subset from $\mathbf{X}_{\mathrm{Pop}}$.

DEMO is a multi-objective evolutionary search algorithm that has been recently introduced by Robic and Filipic [21]. Its core procedure combines single-objective Differential Evolution with *Pareto-sorting* and *Crowding Distance* mechanisms. This paper proposes a DEMO inspired search algorithm which introduces an external archive [22] that will be used to both store all efficient solutions as well as to further promote the search towards the solution space most promising area. Through the rest of this paper, the proposed approach will be referred as ADEMO. While UMCS evaluates a single population of candidate solutions, ADEMO approach starts with a smaller population that evolves through an iterative procedure for a defined number of rounds called generations. To allow fair comparability, the total number of generations multiplied by the dimension of ADEMO population is set equal to the UMCS population, therefore both algorithms' search procedures use the same amount of trials.
As in UMCS, the ADEMO procedure starts by generating an initial population of p candidate solutions $\boldsymbol{X}_1, \dots, \boldsymbol{X}_p$ with $p < P$ with the same procedure employed by UMCS. Each candidate solution is evaluated and compared to all the others to identify the initial Pareto Frontier approximation. The subset of efficient solutions is then copied in an external archive called $\overline{\boldsymbol{A}}$ that will be used to store all efficient solutions observed by the search procedure at each generation. After initializing population and archive, the search procedure employs an iterative procedure composed by the following operators (see Algorithm 2).

Algorithm 2: ADEMO - <u>Reproduce</u>
A new set of candidate solutions is generated by combining the external archive with the current population, specifically each new solution $\boldsymbol{X}_1^C, \dots, \boldsymbol{X}_p^C$ is generated as:

- $H_j^C = H_{P1}^{\overline{\boldsymbol{A}}} \odot S_j + H_{P2} \odot \overline{S_j}$
- $A_j^C = A_{P1}^{\overline{\boldsymbol{A}}} \odot S_j + A_{P2} \odot \overline{S_j}$
- $P1 = Sample\left(1, \min\left(p, length(\overline{\boldsymbol{A}})\right)\right)$
- $P2 = Sample(1, p)$
- $S_j = \{s_1^j, \dots, s_m^j\}$ with $s_i^j \sim Bernoulli(0.5)$ for $i = 1, \dots, m$
- $\overline{S_j} = \{\bar{s}_1^j, \dots, \bar{s}_m^j\}$ with $\bar{s}_i^j = 1 - s_i^j \ \forall\, i = 1, \dots, m$

where H and A represent respectively the selection and renewal vectors of a solution $\boldsymbol{X}$. $\boldsymbol{X}_{P1}^{\overline{\boldsymbol{A}}} = \left(H_{P1}^{\overline{\boldsymbol{A}}}, A_{P1}^{\overline{\boldsymbol{A}}}\right)$ represents a candidate solution randomly picked from the external archive while $\boldsymbol{X}_{P2} = (H_{P2}, A_{P2})$ has been drawn from the current population $\boldsymbol{X}_1, \dots, \boldsymbol{X}_p$. S_j is a randomly generated binary vector that allow to efficiently select features from $\boldsymbol{X}_{P1}^{A}$ while $\overline{S_j}$ will select the remaining features from $\boldsymbol{X}_{P2}$.
Notice that only the first p elements from the archive are selected for reproduction,

indeed the searching procedure will update the external archive at each iteration allowing it to grow unlimitedly. Furthermore, for each element in $\bar{\boldsymbol{A}}$ and each iteration, the so-called *crowding distance* ([23]), which represents the Euclidean distance of an element with its nearest neighbourhood in the solution space, will be evaluated. Thereafter, the archive is decreasingly sorted by the crowding distance allowing for reproduction only those solutions with the greater crowding distance. This procedure is meant to avoid excessively concentration of the search algorithm in a specific area of the solution space.
Finally, each candidate solution $\boldsymbol{X}_j^C$ will be randomly mutated by switching each element of its selection vector with a probability p_{mutate} that is an external parameter of the ADEMO algorithm. For each selection element that has been mutated, the related renewal price would be mutated as well by adding a value equal to $\varepsilon \sim U(\alpha_{min}, \alpha_{max})$. If the resulted renewal price would exceed the allowable range, it will automatically set to its nearest limit.

Algorithm 2: ADEMO - <u>Merge</u>
Each element of population $\boldsymbol{X}_1^C, \dots, \boldsymbol{X}_p^C$ is evaluated and compared with the corresponding element of the current population $\boldsymbol{X}_1, \dots, \boldsymbol{X}_p$. Following the preference structure previously defined, the merge step operates as follow:

1. If $\boldsymbol{X}_j^C \succ \boldsymbol{X}_j$ then $\boldsymbol{X}_j^C$ substitutes $\boldsymbol{X}_j$ in the current population;
2. else if $\boldsymbol{X}_j^C \prec \boldsymbol{X}_j$ then $\boldsymbol{X}_j^C$ is discard;
3. else $\boldsymbol{X}_j^C$ is added to the current population.

This procedure will lead to a new population whose dimension $p_{\cup}$ will range from p to $2p$.

Algorithm 2: ADEMO - <u>Truncate</u>
To restore the original cardinality of p elements in the population, $p_{\cup} - p$ solutions are discarded through the following procedure:

1. start with the complete population of $p_{\cup}$ elements;
2. compare each solution in the population with all the other and select the efficient ones;
3. store those solutions in an external memory and mark their *level of efficiency*;
4. remove efficient solutions from population;
5. re-execute steps 2, 3 and 4 until all candidate solutions have been marked.

The *level of efficiency* is defined by the cycle iteration in which a solution is flagged as efficient. Intuitively, solutions that are selected in the first iteration belong to the highest generation's efficient frontier, the second iteration will identify the generation's efficient frontier that do not consider those already selected and so on. Therefore, the population is *stratified* in a sequence of frontiers where the highest

one represents the actual Pareto Frontier of the generation (notice: this procedure is called *Pareto Sorting*). Following, for each frontiers the *crowding distance* of their solution is evaluated by the same procedure presented for the external archive in the reproduction step.
Finally, solutions are sorted by their ascending level of efficiency and descending crowding distance. Intuitively, solution that stays on a higher frontier are preferable being a closer approximation to the real unknown Pareto Frontier. At the same time, more spaced solutions are preferred to induce a better exploration of the solution space. Once sorted, the current population discard the last $p_{\cup} - p$ solutions restoring the original cardinality of p.

Algorithm 2: ADEMO - Update
The last step of the iterative cycle will update the external archive by adding the current population and then filtering only those solutions that are efficient. As mentioned above, the archive can grow unlimitedly but for reproduction purposes only the first p elements associated with the greatest *crowding distance* will be considered. Notice that this step does not execute the Pareto Sorting procedure because the external archive will only consider the highest frontier known by the search algorithm at any iterative step.

This iterative cycle of Reproduce, Merge, Truncate and Update will be repeated for a given number of generations defined as an external procedure parameter. When compared with UMCS, ADEMO adopts an iterative procedure that is engineered to push the random search towards the more promising area of the solution space allowing a faster convergence rate. Therefore, greater computational efficiency is expected by this former algorithm.

Algorithm 2: ADEMO
Input

- p *population dimension*
- p_{mutate} *mutation probability*
- G *number of generations*
- *Seed random number generator seed*

Initialization

- *Set Seed*
- *Randomly create population* $\boldsymbol{X}_{Pop} = \{\boldsymbol{X}_1, \dots, \boldsymbol{X}_p\}$
- $\overline{\boldsymbol{A}} = FindFrontier(\boldsymbol{X}_{Pop})$

Evolutionary Cycle

For $g = 1$ *to* G

- $\boldsymbol{X}_{Pop}^{C} = \{\boldsymbol{X}_1^C, \dots, \boldsymbol{X}_p^C\} = Reproduce(\boldsymbol{X}_{Pop}, \overline{\boldsymbol{A}};\ p_{mutate})$
- $\boldsymbol{X}_{\cup} = Merge(\boldsymbol{X}_{Pop}, \boldsymbol{X}_{Pop}^{C})$
- $\boldsymbol{X}_{Pop} = Truncate(\boldsymbol{X}_{\cup}; p)$
- $\overline{\boldsymbol{A}} = FindFrontier(\overline{\boldsymbol{A}} \cup \boldsymbol{X}_{Pop})$

next g

Output

- *Return* $\overline{\boldsymbol{A}}$

4. Empirical Evidence

ADEMO and UMCS algorithms have been compared through several simulation experiments designed to allow a fair performance comparison. A total of 3.708 runs of both algorithms have been performed to asses performance's sensitivity to changes in:

- Portfolio's Dimension: number of potential insured;
- Portfolio's Level of Homogeneity: single insured risk profile diversity;
- Market Competency Level: customer sensitivity level to change in renewal strategies;
- Algorithms' total of iteration.
-

Following sub-sections will present: IT infrastructure specification, single experiment run detailed description and adopted evaluation metrics. Numerical results will be display in final section.

4.1 Technical Specifications

Both UMCS and ADEMO algorithm have been implemented using base R code (version x64 3.5.2), libraries were used only for graphical representation, efficient data management and to evaluate algorithms' Pareto Frontier approximation.

Code and results can be found at: "*https://github.com/AndreaRiva1991/Portfolio-Optimization*".
Considering the remarkable computing effort required to run the full simulation experiment, all runs execution has been performed on a Compute Optimized Instance (c4.8xlarge – 36 CPU) provided by Amazon Web Service cloud computing EC2.
Overall, 3.708 single runs have been executed through 103 macro cycles in each of which 36 experiments where run in parallel.

4.2 Single Run Description

Single run is defined by the following five macro steps:

1. Simulate Portfolio Homogeneity Level;
2. Simulate Synthetic Portfolio;
3. UMCS Execution;
4. ADEMO Execution;
5. Algorithms performances recording.

A portfolio is featured by m statistically independent contracts fully described by their frequency and severity distributional profile. Each synthetic contract is generated as follow:

- $\widetilde{N}_i \sim Poi(\lambda_i)$ with $\lambda_i = \left|F_{mn} + N\left(0, F_{mn_{sd}}\right)\right|$;
- $\tilde{Z}_{j,i} \sim \Lambda(\mu_i, \sigma_i)$ with $\mu_i = \left|S_{mn} + N\left(0, S_{mn_{sd}}\right)\right|$ and $\sigma_i = \left|S_{sd} + N\left(0, S_{sd_{sd}}\right)\right|$.

where

- $F_{mn} = Portfolio\ Frequency\ Mean$
- $S_{mn} = Portfolio\ Severity\ Position\ Parameter$
- $S_{sd} = Portfolio\ Severity\ Diffusion\ Parameter$
- $F_{mn_{sd}} = Individual\ Frequency\ Mean\ Variability$
- $S_{mn_{sd}} = Individual\ Severity\ Position\ Parameter\ Variability$
- $S_{sd_{sd}} = Individual\ Severity\ Diffusion\ Parameter\ Variability$
-

Individual variability parameters $\left(F_{mn_{sd}}, S_{mn_{sd}}, S_{sd_{sd}}\right)$ allow to control the level of portfolio homogeneity ranging from perfectly homogeneous to highly non-homogeneous one. To assess portfolio homogeneity's impact on algorithms' performance, each run initially simulates a set of variability parameters as:

$$F_{mn_{sd}} \sim U\left(0, F^{Max}{}_{mn_{sd}}\right)$$
$$S_{mn_{sd}} \sim U\left(0, S^{Max}{}_{mn_{sd}}\right)$$
$$S_{sd_{sd}} \sim U\left(0, S^{Max}{}_{sd_{sd}}\right)$$

where $F^{Max}{}_{mn_{sd}}, S^{Max}{}_{mn_{sd}}, S^{Max}{}_{sd_{sd}}$ are external parameter defined by user.

The simulated portfolio is then processed by both UMCS and ADEMO algorithms as described in previous section. Finally, both Efficient Frontier Approximations are assessed through several metrics that will be described in the following sub-section.

4.3 Evaluation Metrics

Multi-objective optimization algorithm's performance can be evaluated by measuring the approximated Efficient Frontier *quality* which is defined by two opposites goals:

1. find an approximated frontier as close as possible to the real Pareto Frontier;
2. find an approximated frontier as diverse as possible.

Specifically, the diversity goal is meant to balance algorithm's *Exploitation-Exploration Trade Off*, preventing an excessively concentration on a limited area of the solution space.

The following evaluation metrics have been adopted in this simulation experiment ([24]):

1. Spacing: defined as the standard deviation of the Euclidean distances between each non-dominated solutions with its closest neighbourhood. If solutions are nearly spaced, the corresponding distance will be small, indeed Pareto Frontier approximation with small spacing is preferred.
2. Spread: defined as

$$S = \frac{\sum_{h=1}^{3} d_h^e + \sum_{i \in EF}\left(d_i - \bar{d}\right)}{\sum_{h=1}^{3} d_h^e + \#EF\ \bar{d}}$$

   where
   - EF = efficient frontier set;
   - d_i = Euclidean distance of solution i to its closest neighbourhood in the solution space;
   - $\bar{d} = \frac{1}{\#EF}\sum_{i \in EF} d_i$;
   - d_h^e = difference between the minimum and maximum values obtained in the solution set for the objective function f_h.

Approximations associated with smaller S are featured by a better diversity and therefore preferred.

3. Range: defined as the sum of differences between maximum and minimum values for each objective function. It measures the largest area covered by the optimization search, hence algorithms' solutions featured by a greater Range are preferred.
4. Hypervolume: defined as the volume covered by the approximated Efficient Frontier. Intuitively, the true but unknown Pareto Frontier would overhang every other approximated frontier, therefore its underlying volume would be greater. Approximations with greater Hypervolumes are then preferable.
5. Dominance: given several Efficient Frontier approximations, a new one can be obtained by combining all candidate solutions. From this new approximation it is then possible to count how many solutions originated from each primitive Pareto Frontier. Algorithms that originate a greater amount of solutions are then preferred.
6. Cardinality: defined as the number of non-dominated solutions that feature an Efficient Frontier approximation, hence algorithms that produce approximation with a greater cardinality are preferred.

As described in ([24]) hypervolume is the most adopted evaluation metric in Multi-Objective Optimization literature being both a convergence and diversity metric. However, in some particular instances, the assessment of algorithms based solely on hypervolume could lead to biased perception of their performances.
As an example, consider a general case in which there are two Efficient Frontier approximations, one of which is consistently above the other, therefore its hypervolume metric would be greater. Now assume that the lower Efficient Frontier is featured by few exceptionally high solution, indeed these outliers could abnormally increase the underlying volume of the lower approximation up to a point in which its hypervolume metric would be greater than the *normally* higher frontier. Hence, speculate instance could occur with low outliers that could abnormally decrease the underlying volume of a higher solution set.
Without considering the highness in the solution space of a given point, Dominance metric is meant to recognize which approximation *normally* dominates the other. Resuming ADEMO's formal notation, assume two different solutions $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ featured by their Efficient Frontier approximations $\overline{\boldsymbol{A}}_j = FindFrontier(\boldsymbol{X}_j)$ for $j = 1, 2$.
Define the combined approximation as:

$$\overline{\boldsymbol{A}}_{\cup} = FindFrontier(\boldsymbol{X}_1 \cup \boldsymbol{X}_2)$$

Dominance metric is then defined as:

$$D_j = \frac{\#\{x_i \in \boldsymbol{X}_h \wedge x_i \in \overline{\boldsymbol{A}}_\cup\}}{\#\overline{\boldsymbol{A}}_\cup}$$

Intuitively, Dominance defines the frequency of solutions originated from $\boldsymbol{X}_j$ that are also found in the combined approximation. By not taking into account solutions' positions, dominance metric is not affected by possible outliers' distortions in hypervolume. To the author's knowledge, this type of evaluation metric is still lacking in multi-objective optimization literature and hence will be presented here. As a final remark, closeness metrics could not be exploit in present simulation experiment having the true Pareto Frontier unknown. However, the numerical experiment aimed to compare algorithms' performances to each other, therefore the following section will actually present *standardized performance deltas* for all evaluation metrics. Standardization compels performance metrics into $[0,1]$ range allowing to easily compare algorithms' performance on several aspects.

4.4 Numerical Results

This section presents the numerical results achieved by 3.708 runs described in previous section, full parameterization can be found in the Appendix. For each run, both UMCS and ADEMO frontier approximations have been evaluated through six quality metrics (Spacing, Spread, Range, Hypervolume, Dominance and Cardinality). Following figures will present standardized differences between the two searching algorithms for each evaluation metric.
Specifically, the simulation experiment has been organized in two main chunks:

1. Evaluate algorithms' performance sensitivity to change in external conditions such as Portfolio Homogeneity, Portfolio Dimension and Market Competency Level;
2. Evaluate algorithms' performance sensitivity to change in algorithms' internal parameters

To easily represents Portfolio Homogeneity Level with a standardize metric the following measurement has been proposed:

$$T = \frac{F_{mn_{sd}} + S_{mn_{sd}} + S_{sd_{sd}}}{F^{Max}{}_{mn_{sd}} + S^{Max}{}_{mn_{sd}} + S^{Max}{}_{sd_{sd}}}$$

where $T \in [0,1]$.
Intuitively, if $T = 0$ then all potential customers are featured by the same distributional profile, if $T = 1$ the maximum level of diversity allowed is reached. Figure 1 shows Portfolio Homogeneity's distribution achieved through all simulation experiments. As expected from the definition of T, as a sum of three uniform distributions, the empirical distribution presents a seemingly Gaussian shape.

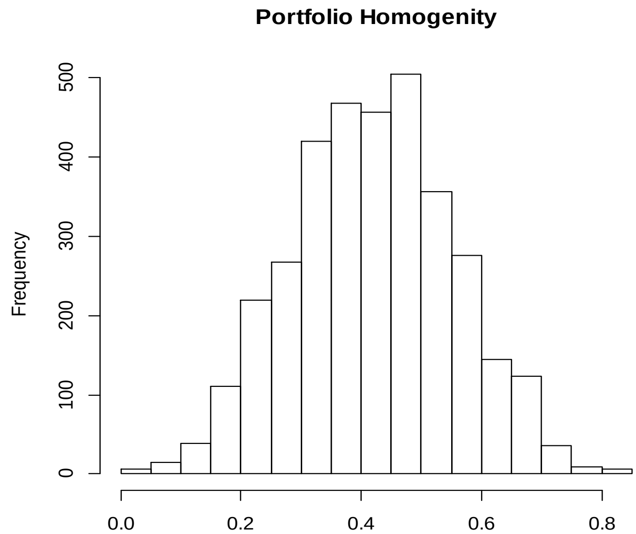

Figure 1: Numerical Distribution of Portfolio Homogeneity Level.

Concerning Figure 2, it appears that all the standardized values of delta are not affected by change in Portfolio Homogeneity Level, therefore both algorithms similarly react to in T.

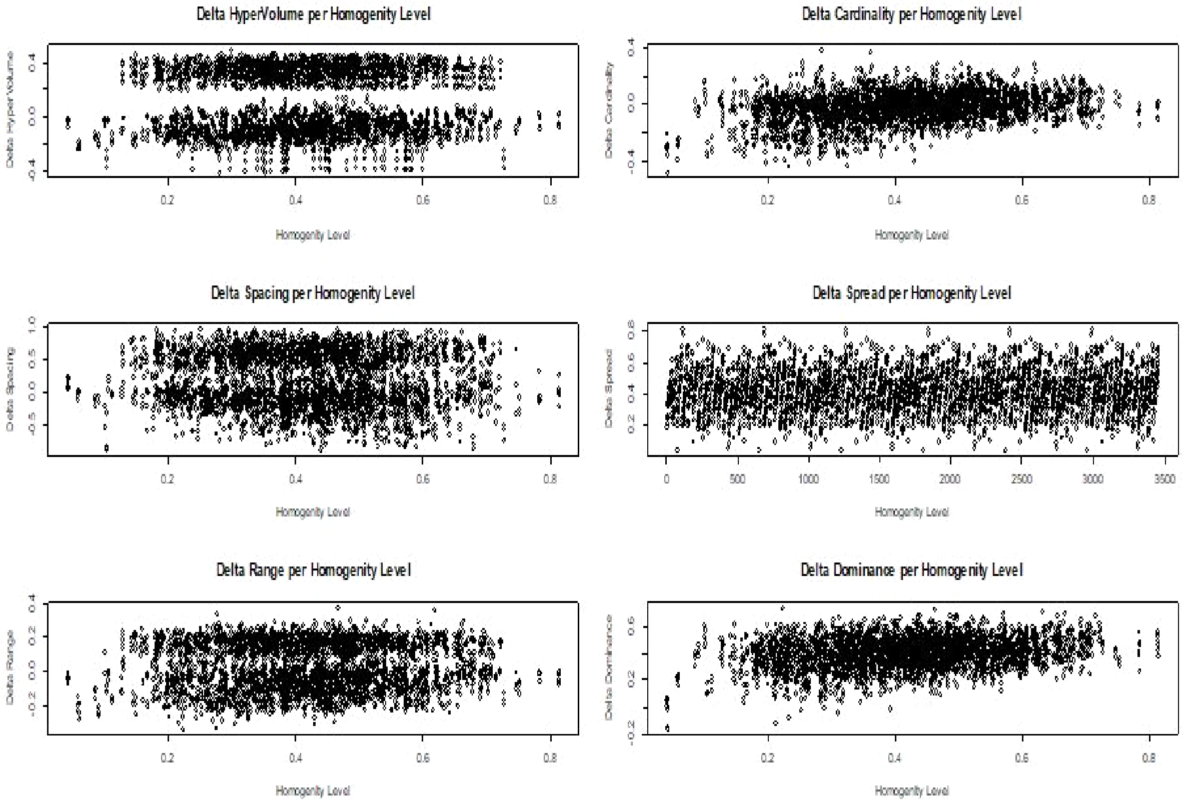

Figure 2: Standardized Evaluation Metric Deltas per Homogeneity Level.

Table 1 presents all standardized deltas Monte Carlo statistics achieved considering all runs from experiment's first chunk.

Table 1: Monte Carlo Statistics from all runs in experiment's first chunk.

Index	Statistic	P_Hyper	P_CardinalityStd	P_SpacingStd	P_SpreadStd	P_RangeStd	P_DominanceStd
1	Min	-40.46%	-47.21%	-89.09%	-4.93%	-34.33%	-16.41%
2	q_0.05	-19.73%	-21.39%	-49.13%	-2.51%	-19.77%	20.70%
3	q_0.25	-9.90%	-7.46%	-14.67%	-1.74%	-8.94%	32.95%
4	q_0.50	0.91%	-1.02%	13.70%	-1.14%	0.55%	40.72%
5	q_0.75	33.00%	4.92%	56.10%	-0.56%	16.26%	47.98%
6	q_0.95	41.68%	13.79%	78.13%	0.29%	22.11%	57.77%
7	Max	48.56%	38.15%	96.00%	2.93%	37.75%	74.10%
8	Mean	9.63%	-1.81%	18.04%	-1.14%	2.63%	40.13%
9	Sd	22.74%	10.48%	41.67%	0.87%	14.12%	11.40%
10	Skew	8.50%	-50.03%	-12.42%	8.29%	-8.84%	-43.33%
11	Prob(ADEMO>UMCS)	52.46%	44.53%	59.32%	9.46%	51.59%	99.74%

From the diversity perspective, algorithms' approximation seems to provide comparable results in terms of Spread and Range although latter metric present a considerably high deviation which indicates possible substantial divergence from the mean. Interestingly, $Prob(ADEMO > UMCS)$ on Range, which indicates the frequency in which ADEMO solutions provide a greater range than UMCS, is almost 50% which could indicate that both algorithms provide essentially the same quality in terms of this metrics. Differently, the probability of having higher Spread metric from ADEMO algorithms is only approximately 10% which indicate a better diversity in ADEMO solution than UMCS in terms of Spread metric.

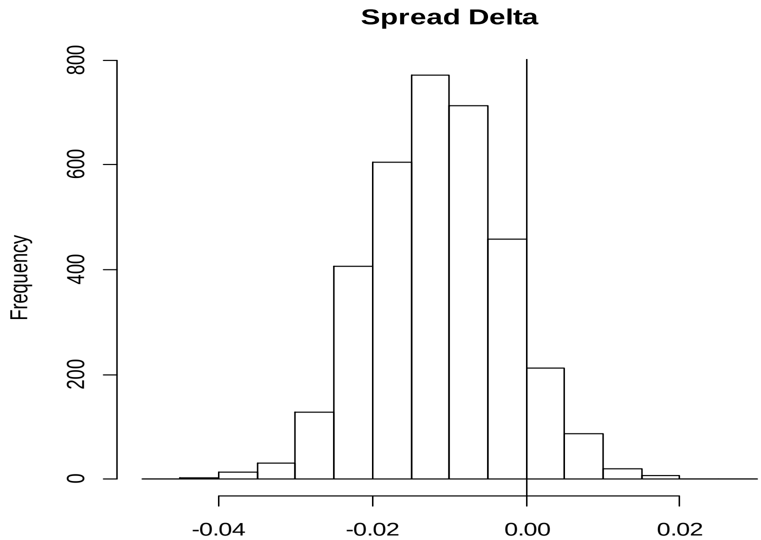

Figure 3: Standardize Spread Delta distribution from all runs in experiment's first chunk.

In terms of Spacing, UMCS seems to bring better spaced solutions although the skewness of the distribution shows a considerably high value which could be affected by abnormal realizations. Nonetheless, UMCS algorithm probability to provide better spaced solutions is almost 60%.

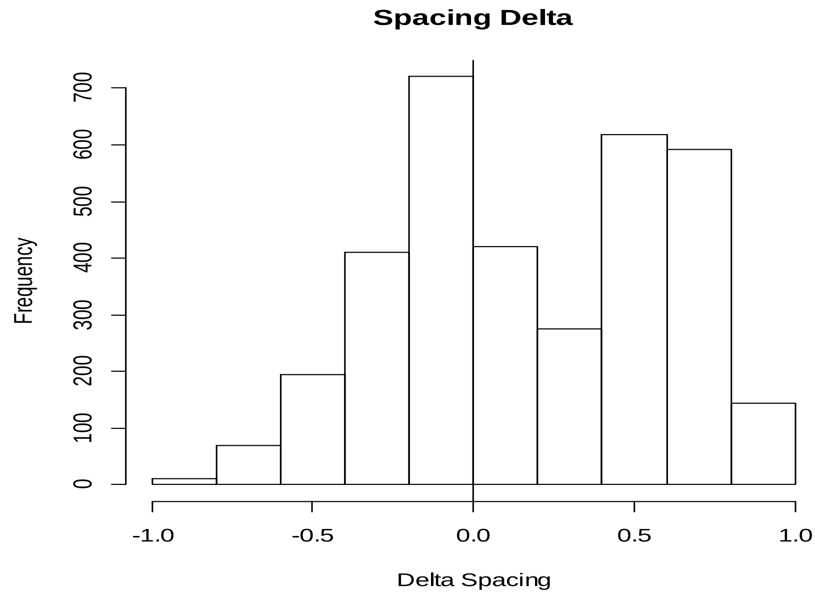

Figure 4: Spacing Delta distribution from all runs in experiment's first chunk.

Cardinality metric distribution seems to presents Gaussian's attributes as shown in Figure 5. From numerical distribution it seems that no algorithm is able to provide consistently more granular solutions as indicated by $Prob(ADEMO > UMCS)$ which indicates the frequency in which ADEMO solutions present a greater cardinality than UMCS.

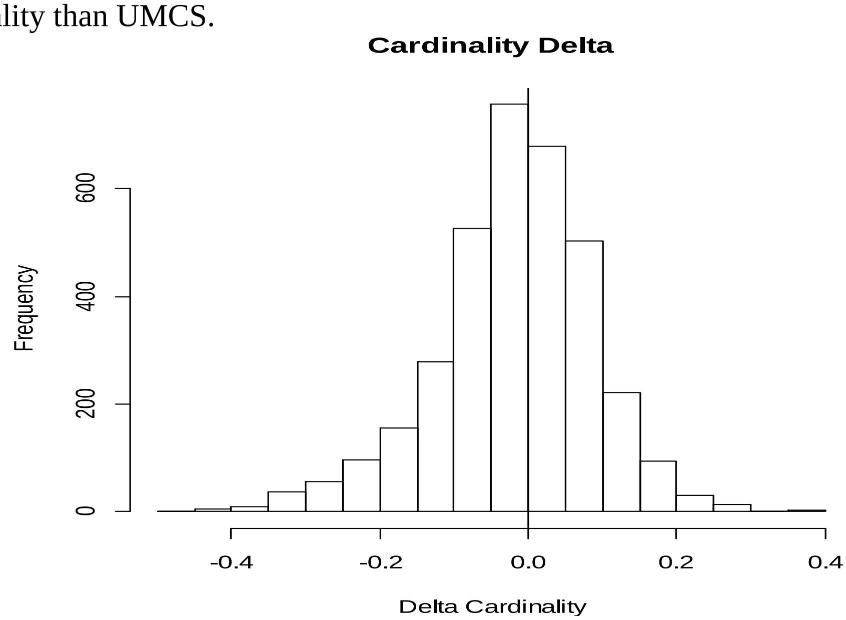

Figure 5: Cardinality Delta distribution from all runs in experiment's first chunk.

Regarding hypervolume metric, numerical results show that on average ADEMO algorithm is capable to find an approximation featured by a 10% greater underlying volume. However, other statistics highlighted how this better performance occurs with a 50% frequency which suggest that there could be no substantial difference between ADEMO and UMCS algorithms. Indeed, the positive average result could be caused by few abnormally positive runs in which ADEMO performed substantially better than UMCS.

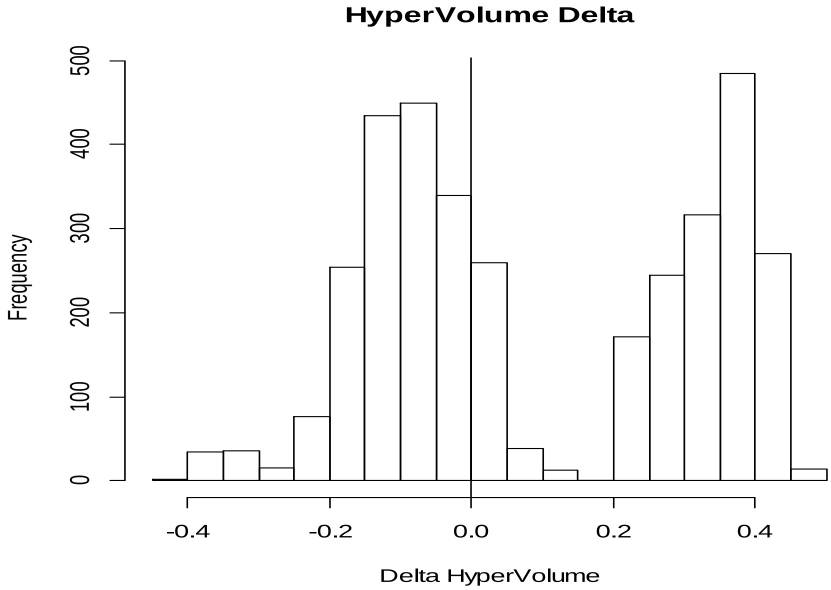

Figure 6: Hypervolume Delta distribution from all runs in experiment's first chunk.

This interpretation seemed to be confirmed by the bi-modals numerical distribution's shape which could suggest that solutions from the two algorithms are not substantially different in terms of hypervolume metric.

As previously suggested, hypervolume metric could be biased by both high and low outlier in the Efficient Frontier approximations, to avoid this shortcoming the Dominance metric has been proposed. Interestingly, Dominance numerical distribution shows that, on average, ADEMO provides 40% more solution to the aggregate approximation than UMCS. From the $Prob(ADEMO > UMCS)$ statistics it seems that ADEMO approximation *normally* dominate UMCS solution in almost all runs.

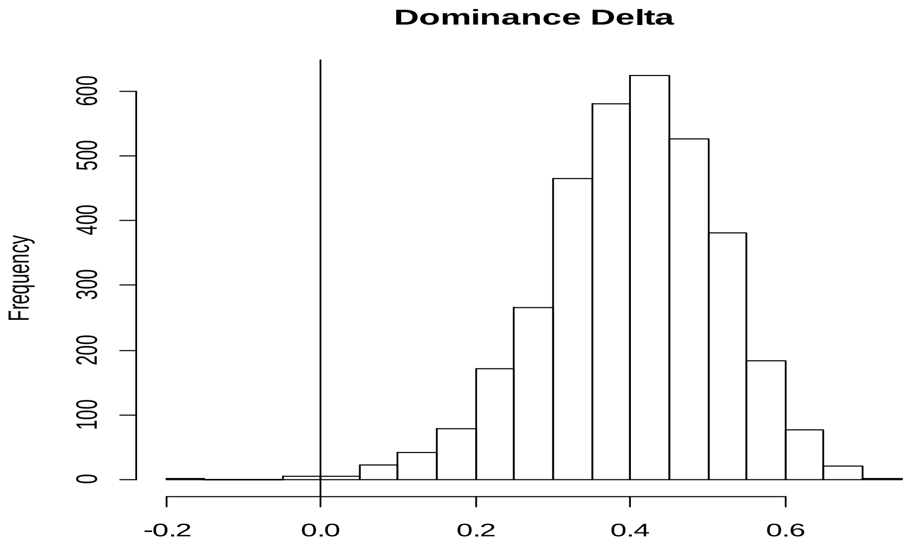

Figure 7: Dominance Delta distribution from all runs in experiment's first chunk.

Dominance and Hypervolume results could collectively suggest that first chunk's experimental runs are potentially still affected by a non-trivial amount of uncertainty which could indicate that both algorithm haven't converge yet to stable solutions. Specifically, both algorithms could need more iterations to achieve stable approximations, therefore experiment's second chunk of run will be featured by a greater amount of iterations for both ADEMO and UMCS. The following two tables report Monte Carlo mean and $P(ADEMO > UMCS)$ sensitivity to change in portfolio and market condition, Specifically, Portfolio High sensitivity assumes a greater number of contracts selectable by the insurer while Market Low, Medium and High define three different policy holder behaviour modelling settings featured by an increasing level of competition in the market. For further details on the assumed parameterization please go to appendix.

Table 2: Monte Carlo Average sensitivity to change in Portfolio and Market conditions.

Index	Trials	Type	P_Hyper	P_CardinalityStd	P_SpacingStd	P_SpreadStd	P_RangeStd	P_DominanceStd
1	1000	Total	9.63%	-1.81%	18.04%	-1.14%	2.63%	40.13%
2	1000	Portfolio Low	10.86%	-0.49%	17.62%	-1.20%	4.08%	40.34%
3	1000	Portfolio High	8.41%	-3.13%	18.46%	-1.08%	1.19%	39.93%
4	1000	Market Low	12.64%	-3.85%	20.84%	-1.24%	5.49%	43.63%
5	1000	Market Medium	9.43%	-1.57%	18.07%	-1.15%	2.46%	38.66%
6	1000	Market High	6.83%	-0.01%	15.21%	-1.03%	-0.04%	38.11%

Table 3: Monte Carlo P(ADEMO>UMCS) sensitivity to change in Portfolio and Market conditions.

Index	Trials	Type	P_Hyper	P_CardinalityStd	P_SpacingStd	P_SpreadStd	P_RangeStd	P_DominanceStd
1	1000	Total	52.46%	44.53%	59.32%	9.46%	51.59%	99.74%
2	1000	Portfolio Low	54.92%	48.38%	58.62%	9.03%	53.59%	99.65%
3	1000	Portfolio High	50.00%	40.68%	60.01%	9.90%	49.59%	99.83%
4	1000	Market Low	51.13%	36.72%	60.76%	5.99%	54.08%	99.91%
5	1000	Market Medium	51.91%	46.53%	59.29%	10.24%	49.83%	99.83%
6	1000	Market High	54.34%	50.35%	57.90%	12.15%	50.87%	99.48%

While Dominance metric seems to be fairly resilient, other metrics such as Hypervolume and Range show greater sensitivity. As expected, statistic $P(ADEMO > UMCS)$ is apparently not impacted by change in external and portfolio condition. By definition, the latter statistic only considers frequency on which ADEMO solutions are better than UMCS but it does not take into account by *how much* ADEMO solutions are better, therefore, $P(ADEMO > UMCS)$ is less affected by potential outliers in algorithms performance. Finally, table 4 presents Monte Carlo statistics achieved by considering all runs from second experiment's chunk. Specifically, this second chunk of simulations allows both algorithms to execute a higher amount of trials, precisely from 1000 to 2000.

Table 4: Monte Carlo Statistics from all runs in experiment's second chunk.

Index	Statistic	P_Hyper	P_CardinalityStd	P_SpacingStd	P_SpreadStd	P_RangeStd	P_DominanceStd
1	Min	-13.56%	-35.78%	-33.95%	-2.98%	-8.78%	22.42%
2	q_0.05	-10.37%	-16.95%	-7.47%	-2.21%	-4.11%	34.77%
3	q_0.25	35.58%	-2.68%	30.31%	-1.71%	16.99%	45.75%
4	q_0.50	36.15%	5.01%	48.45%	-1.18%	20.18%	52.56%
5	q_0.75	38.32%	10.36%	62.43%	-0.73%	22.68%	56.77%
6	q_0.95	43.52%	17.18%	82.13%	-0.11%	28.84%	64.48%
7	Max	45.01%	21.24%	95.72%	0.59%	46.05%	69.70%
8	Mean	28.69%	2.91%	43.53%	-1.19%	17.33%	51.35%
9	Sd	18.76%	10.28%	26.69%	0.67%	10.37%	8.51%
10	Skew	-136.17%	-83.91%	-62.62%	12.67%	-75.25%	-58.32%
11	Prob(ADEMO>UMCS)	78.89%	63.89%	91.67%	3.89%	87.78%	100.00%

In terms of Hypervolume, ADEMO seems to experience a considerable increase in performance moving from an average of 9.63% up to 28.69% while standard deviation decrease of about 4%. Furthermore, $P(ADEMO > UMCS)$ statistic moved from 52.46% to 78.89% suggesting that ADEMO better performance is not purely incidental. Concurrently, Dominance metric raises from 40.13% to 51.35% while Range gains 15%.

Comparing with Table 1, Cardinality average increases from -1.81% to 2.91% with an almost invariant standard deviation and a $P(ADEMO > UMCS)$ statistic indicating that this better performance, although slight, happens with a 60% frequency.

Finally, comments from experiment first chunk about Spacing metric are confirmed with an average of 43.53%, starting from 18% in first chunk, and a $P(ADEMO > UMCS)$ statistic of almost 92% which indicates that ADEMO consistently provide poor performance in terms of Spacing when compared with UMCS.

5. Conclusion, Limitations and Future Work

This paper presents an application of Evolutionary Multi-Objective Optimization to the portfolio renewal problem for a non-life insurance company. Assuming a competitive market, an existing insurance contract portfolio and a pricing/policyholder behavioural model, the insurer has to decide which contracts retain as well as the renewal quotation to offer.

As described by the policyholder behavioural model, potential customers accept a proposed quotation with probability dependent on their risk profiles, the renewal proposition and the market's competency level. Therefore, companies need to carefully design a selection/optimization strategy that allows to reach the profitability/solvency targets defined by the management committee as well as to maximally retain desirable customers. The renewal problem is then naturally formalize as a three objective optimization problem whose ultimate goal is to approximate the Pareto Frontier of all possible selection/renewal strategies.

Several search algorithms are available in multi-objective optimization literature, nonetheless this paper focused on the evolutionary family for its built-in capability to simultaneously handle several candidate solutions which is particularly suitable in a multi-objective problem where there is no single optimal solution but a set of non-dominated one instead.

Introducing an external archive mechanism for both elitism preservation and faster convergence, a DEMO inspire algorithm has been compared with a simple Uniform Monte Carlo Search strategy. Several numerical experiments showed that, as the number of iteration of both algorithms increase, performance achieved by the propose evolutionary approach substantially and consistently outperform the pure random search for almost all the evaluation metrics adopted. While UMCS simply evaluates several independently random generated selection/renewal strategies, ADEMO exploits knowledge acquired through generations, driving the random search towards more promising areas of the solution space, indeed achieving better performance.

Algorithms' performance comparison on not entirely stabilized run induced the design of the Dominance evaluation metric which, by assessing the frequency of solutions originated by a search strategy on a combined Pareto Frontier approximation without considering their actual search space position, is not affected by abnormally high or low realization that could anomaly increase/decrease the hypervolume metric.

Presently, actuarial literature's discussion on non-life portfolio optimization problem has mainly focused on the design of accurate policyholder behaviour model and Efficient Frontier approximation on Risk and Retention metrics. Present paper's purpose is to highlight meta-heuristic optimization algorithm's capability to easily handle more general problems by introducing a third optimization objective. Indeed, on a pure actuarial perspective, the underlying model structure presents several improvement opportunities such as:

1. dependencies through potential customers may be introduce;
2. new customers, that do not belong to the starting portfolio, could be modelled;
3. multiple portfolios, possibly dependent, could be simultaneously modelled;
4. renewal quotations could be define on a discrete grid.

Although all these extension potentially present non-trivial implementation issues, remarkably both optimization procedures would not be affected by these improvements. By their very nature, meta-heuristic algorithms are not concerned by the underlying structure of the objective functions which are dealt as black boxes. Therefore, all actuarial concept and sophistications will only affect the underlying behaviour of the *black box* without affecting the searching strategy.

Meta-heuristic optimization could then be exploit in several actuarial contests featured by complexity levels such as classical mathematical optimization is infeasible. Simultaneously, meta-search strategies may also allow actuaries to enrich classical optimization problems with realistic constraints that may excessively burden their mathematical formalization .

Regarding the optimization strategies, practical implementations showed how objective function evaluation appears to be the most time-consuming task therefore, a hybrid approach could presents an initial *warm up* UMCS phase that is employed to train an objective function approximation $\hat{f}$ whose computation time shall be considerable lower than f. Reproducing a *Least Square Monte Carlo* approach, ADEMO procedure should then rely on $\hat{f}$ instead of actually recalculate the objective functions at each generation for all candidate solutions. Another promising research area may consists in designing a parallelized version of both search algorithms allowing a potentially massive computation time reductions by exploiting modern computation accelerator such as Graphical Processing Unit.

ACKNOWLEDGEMENTS. The author wishes to thank Prof. Gian Paolo Clemente for his continuous support and precious feedbacks.

References

[1] M. Aleandri, Modeling Dynamic Policyholder Behaviour through Machine Learning Techniques, Università La Sapienza. Rapporto Tecnico, available at: https://www.dss.uniroma1.it/it/system/files/pubblicazioni/CopiaPubblicazione.pdf, (2017).

[2] C. Dutang, L. Petrini. and G.A. Spedicato, Machine Learning Methods to Perform Pricing Optimization. A Comparison with Standard GLMs, Variance, 12(1), (2018), 69-89.

[3] J. Campbell, Modeling of Policyholder Behaviour for Life Insurance and Annuity Products: A survey and literature review, Society of Actuaries, available at: https://www.soa.org/resources/research-reports/2014/research-2014-modeling-policy, (2014).

[4] F. Gomez-Alvado and M. Looft, Using Analytics beyond Price Optimization, The Actuarial Profession, available at: https://www.actuaries.org.uk/system/files/documents/pdf/b02gomez-alvado.pdf, (2007).

[5] K. Kaufhold and W. Lennartz, Optimizing Risk Retention: Quantitative Retention Management for Life Insurance Companies, Society of Actuaries, available at: https://www.soa.org/resources/research-reports/2016/research-quantitative-retention, (2016).

[6] E. Krasheninnikova & alt., Reinforcement learning for pricing strategy optimization in the insurance industry, Engineering Applications of Artificial Intelligence, 80(1), (2019), 8-19.
[7] Gazzetta Ufficiale, Legge annuale per il mercato e la concorrenza, (2017).
[8] Istituto per la Vigilanza sulle Assicurazioni, Codice delle Assicurazioni Private, Decreto Legislativo n. 209, (2005).
[9] European Insurance And Occupational Pension Authority, 2018 EIOPA Insurance Stress Test Report, ISSN 2599-8854, available at: https://www.eiopa.europa.eu/Publications/Surveys/EIOPA%202018%20Insurance%20Stress%20Test%20Report.pdf, (2018).
[10] M. Bugdee & alt., Price Optimization Overview, Casualty Actuarial Society Price Optimization Working Party, available at: https://www.casact.org/cms/files/Price_Optimization_Overview_FINAL.pdf, (2014).
[11] W. Mayorga and D. Torres, A practical model for pricing optimization in car insurance, 2017 ASTIN and AFIR/ERM Colloquia Panama, available at: https://www.semanticscholar.org/paper/A-practical-model-for-pricing-optimization-in-car-Mayorga-Torres/bddd416f8967adb1e4d3839a98c1fe0e1a8c3df2, (2017).
[12] R. Shnieper, Portfolio Optimization, ASTIN Bulletin, 30(1), (2000),195-248.
[13] A. Zhou & alt., Multi-objective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation, 1(1), (2011), 32-49.
[14] H. Day, M.C. Koissi and V. Whitledge, Emerging Data Analytics Techniques with Actuarial Applications, Society of Actuaries, available at: https://www.soa.org/resources/research-reports/2019/emerging-analytics-techniques-applications, (2019).
[15] M. Echchelh, Optimisation of asset allocation and reinsurance strategy by the mean of a genetic algorithm: NSGA-II, ENSAE Paristech, available at: https://eudl.eu/doi/10.4108/eai.24-4-2019.2284089, (2019).
[16] A.F. Shapiro, Soft Computing Applications in Actuarial Science, Smeal College of Business, Penn State University, University Park, available at: https://www.semanticscholar.org/paper/Soft-Computing-Applications-in-Actuarial-Science-Shapiro/46d7bbea1f1c5bb60128418b22d08da5cd07d08f, (2001).
[17] D. Snell, Genetic Algorithms: Genetic Algorithm Applications to Actuarial Problems, Society of Actuaries, available at: http://www.umanitoba.ca/faculties/management/academic_depts_centres/centres_institutes/warren/media/P6A_Snell_2_GA_workshop_-_ARC_2012-08-04.pdf, (2012).
[18] B. Wadsley, Are Genetic Algorithms Even Applicable to Actuaries?, Society of Actuaries Investment Section, Risk & Rewards, 57(1), (2011), 4-7, available at: "https://www.soa.org/globalassets/assets/library/newsletters/risks-and-rewards/2011/february/rar-2011-iss57-wadsley.pdf".

[19] J.H. Holland, Adaptation in Natural and Artificial Systems, Cambridge, MA: MIT Press, available at: https://mitpress.mit.edu/books/adaptation-natural-and-artificial-systems, (1992).
[20] D. Frenkel, Introduction to Monte Carlo methods, Physica A: Statistical Mechanics and its Applications. 418(1), (2004),78-87.
[21] B. Filipi and T. Robic, DEMO: Differential Evolution for Multiobjective Optimization, EMO 2005. Lecture Notes in Computer Science, 3410(1), (2005), 520-533.
[22] V.L. Huang & alt., Multiobjective Differential Evolution with External Archive and Harmonic Distance-Based Diversity Measure, available at: https://www.semanticscholar.org/paper/Multiobjective-Differential-Evolution-with-External-Huang-Suganthan/17609ee1ad1d5d4c842cc5a28c5bd04e7d0013c2, (2007).
[23] C.A. Coello-Coello and M.R. Sierra, Improving PSO-based Multi-Objective Optimization using Crowding, Mutation and e-Dominance, EMO 2005. Lecture Notes in Computer Science, 3410(1), (2005), 505-519.
[24] N.R. Granada & alt., Performance metrics in multi-objective optimization, 2015 Latin American Computing Conference (CLEI), available at: https://ieeexplore.ieee.org/document/7360024, (2015).

Appendix

Following tables display the parameterization setting adopted by the simulation experiment as well as all macro cycles specification. Within each macro cycle, 36 portfolios are generated in parallel and then evaluated by both UMCS and ADEMO leading on a total of 3.708 single runs. As a final remark, each portfolio simulation adopted a Poisson/Lognormal distributional assumption for single policyholder frequency/severity modelling.

Table 5: Simulation Experiment Parameterization.

Index	Sector	Parameter	Value	Description
1	Macro Setting	alfa	0.99	Tail Value at Risk confidence level
2	Macro Setting	C_Min	-0.50	Renewal Quotation lower limit
3	Macro Setting	C_Max	0.50	Renewal Quotation upper limit
4	Macro Setting	m_low	100	Number of policy holder in Low Portfolio Sensitivity
5	Macro Setting	m_high	200	Number of policy holder in High Portfolio Sensitivity
6	Policyholder Behaviour	Beta_0_Low	0.10	Beta 0 for Low Market Sensitivity
7	Policyholder Behaviour	Beta_0_Medium	0.20	Beta 0 for Medium Market Sensitivity
8	Policyholder Behaviour	Beta_0_High	0.30	Beta 0 for High Market Sensitivity
9	Policyholder Behaviour	Beta_1_Low	0.30	Beta 1 for Low Market Sensitivity
10	Policyholder Behaviour	Beta_1_Medium	0.50	Beta 1 for Medium Market Sensitivity
11	Policyholder Behaviour	Beta_1_High	0.80	Beta 1 for High Market Sensitivity
12	Policyholder Behaviour	Beta_2_Low	1.00	Beta 2 for Low Market Sensitivity
13	Policyholder Behaviour	Beta_2_Medium	1.50	Beta 2 for Medium Market Sensitivity
14	Policyholder Behaviour	Beta_2_High	2.00	Beta 2 for High Market Sensitivity
15	Synthetic Portafolio Simulation	s	10 000	Collective loss number of simulation for each policyholder
16	Synthetic Portafolio Simulation	F_mn	1.00	Portfolio Frequency Mean
17	Synthetic Portafolio Simulation	S_mn	5.00	Portfolio Severity Position Parameter
18	Synthetic Portafolio Simulation	S_sd	1.00	Portfolio Severity Diffusion Parameter
19	Synthetic Portafolio Simulation	F_mn_sd_Max	2.50	Frequency Mean maximum deviation for a single policyholder
20	Synthetic Portafolio Simulation	S_mn_sd_Max	1.50	Severity Position maximum deviation for a single policyholder
21	Synthetic Portafolio Simulation	S_sd_sd_Max	0.50	Severity Diffusion maximum deviation for a single policyholde
22	UMCS	Trials_low	1 000	Number of Monte Carlo trials for experiment chunk 1
23	UMCS	Trials_high	2 000	Number of Monte Carlo trials for experiment chunk 2
24	ADEMO	Generation	10	Number of evolutionary cycles in ADEMO algorithm
25	ADEMO	Pop_N_low	100	Population cardinality for experiment chunk 1
26	ADEMO	Pop_N_high	200	Population cardinality for experiment chunk 2
27	ADEMO	prob_m	0.01	Mutation probability in ADEMO algorithm

Table 6: Macro Cycles Control Table.

Index	Trials	Name	Seed	m	Beta0	Beta1	Beta2
1	1000	Run_PrtLow_MktLow_Number_1	11	100	0.1	0.3	1
2	1000	Run_PrtLow_MktLow_Number_2	12	100	0.1	0.3	1
3	1000	Run_PrtLow_MktLow_Number_3	13	100	0.1	0.3	1
4	1000	Run_PrtLow_MktLow_Number_4	14	100	0.1	0.3	1
5	1000	Run_PrtLow_MktLow_Number_5	21	100	0.1	0.3	1
6	1000	Run_PrtLow_MktLow_Number_6	22	100	0.1	0.3	1
7	1000	Run_PrtLow_MktLow_Number_7	23	100	0.1	0.3	1
8	1000	Run_PrtLow_MktLow_Number_8	24	100	0.1	0.3	1
9	1000	Run_PrtLow_MktLow_Number_9	31	100	0.1	0.3	1
10	1000	Run_PrtLow_MktLow_Number_10	32	100	0.1	0.3	1
11	1000	Run_PrtLow_MktLow_Number_11	33	100	0.1	0.3	1
12	1000	Run_PrtLow_MktLow_Number_12	34	100	0.1	0.3	1
13	1000	Run_PrtLow_MktLow_Number_13	41	100	0.1	0.3	1
14	1000	Run_PrtLow_MktLow_Number_14	42	100	0.1	0.3	1
15	1000	Run_PrtLow_MktLow_Number_15	43	100	0.1	0.3	1
16	1000	Run_PrtLow_MktLow_Number_16	44	100	0.1	0.3	1
17	1000	Run_PrtLow_MktMed_Number_1	11	100	0.2	0.5	1.5
18	1000	Run_PrtLow_MktMed_Number_2	12	100	0.2	0.5	1.5
19	1000	Run_PrtLow_MktMed_Number_3	13	100	0.2	0.5	1.5
20	1000	Run_PrtLow_MktMed_Number_4	14	100	0.2	0.5	1.5
21	1000	Run_PrtLow_MktMed_Number_5	21	100	0.2	0.5	1.5
22	1000	Run_PrtLow_MktMed_Number_6	22	100	0.2	0.5	1.5
23	1000	Run_PrtLow_MktMed_Number_7	23	100	0.2	0.5	1.5
24	1000	Run_PrtLow_MktMed_Number_8	24	100	0.2	0.5	1.5
25	1000	Run_PrtLow_MktMed_Number_9	31	100	0.2	0.5	1.5
26	1000	Run_PrtLow_MktMed_Number_10	32	100	0.2	0.5	1.5
27	1000	Run_PrtLow_MktMed_Number_11	33	100	0.2	0.5	1.5
28	1000	Run_PrtLow_MktMed_Number_12	34	100	0.2	0.5	1.5
29	1000	Run_PrtLow_MktMed_Number_13	41	100	0.2	0.5	1.5
30	1000	Run_PrtLow_MktMed_Number_14	42	100	0.2	0.5	1.5
31	1000	Run_PrtLow_MktMed_Number_15	43	100	0.2	0.5	1.5
32	1000	Run_PrtLow_MktMed_Number_16	44	100	0.2	0.5	1.5
33	1000	Run_PrtLow_MktHig_Number_1	11	100	0.3	0.8	2
34	1000	Run_PrtLow_MktHig_Number_2	12	100	0.3	0.8	2
35	1000	Run_PrtLow_MktHig_Number_3	13	100	0.3	0.8	2
36	1000	Run_PrtLow_MktHig_Number_4	14	100	0.3	0.8	2
37	1000	Run_PrtLow_MktHig_Number_5	21	100	0.3	0.8	2
38	1000	Run_PrtLow_MktHig_Number_6	22	100	0.3	0.8	2
39	1000	Run_PrtLow_MktHig_Number_7	23	100	0.3	0.8	2
40	1000	Run_PrtLow_MktHig_Number_8	24	100	0.3	0.8	2
41	1000	Run_PrtLow_MktHig_Number_9	31	100	0.3	0.8	2
42	1000	Run_PrtLow_MktHig_Number_10	32	100	0.3	0.8	2
43	1000	Run_PrtLow_MktHig_Number_11	33	100	0.3	0.8	2
44	1000	Run_PrtLow_MktHig_Number_12	34	100	0.3	0.8	2
45	1000	Run_PrtLow_MktHig_Number_13	41	100	0.3	0.8	2
46	1000	Run_PrtLow_MktHig_Number_14	42	100	0.3	0.8	2
47	1000	Run_PrtLow_MktHig_Number_15	43	100	0.3	0.8	2
48	1000	Run_PrtLow_MktHig_Number_16	44	100	0.3	0.8	2
49	1000	Run_PrtMed_MktLow_Number_1	11	200	0.1	0.3	1
50	1000	Run_PrtMed_MktLow_Number_2	12	200	0.1	0.3	1
51	1000	Run_PrtMed_MktLow_Number_3	13	200	0.1	0.3	1
52	1000	Run_PrtMed_MktLow_Number_4	14	200	0.1	0.3	1
53	1000	Run_PrtMed_MktLow_Number_5	21	200	0.1	0.3	1
54	1000	Run_PrtMed_MktLow_Number_6	22	200	0.1	0.3	1
55	1000	Run_PrtMed_MktLow_Number_7	23	200	0.1	0.3	1
56	1000	Run_PrtMed_MktLow_Number_8	24	200	0.1	0.3	1
57	1000	Run_PrtMed_MktLow_Number_9	31	200	0.1	0.3	1
58	1000	Run_PrtMed_MktLow_Number_10	32	200	0.1	0.3	1
59	1000	Run_PrtMed_MktLow_Number_11	33	200	0.1	0.3	1
60	1000	Run_PrtMed_MktLow_Number_12	34	200	0.1	0.3	1
61	1000	Run_PrtMed_MktLow_Number_13	41	200	0.1	0.3	1
62	1000	Run_PrtMed_MktLow_Number_14	42	200	0.1	0.3	1
63	1000	Run_PrtMed_MktLow_Number_15	43	200	0.1	0.3	1
64	1000	Run_PrtMed_MktLow_Number_16	44	200	0.1	0.3	1
65	1000	Run_PrtMed_MktMed_Number_1	11	200	0.2	0.5	1.5
66	1000	Run_PrtMed_MktMed_Number_2	12	200	0.2	0.5	1.5
67	1000	Run_PrtMed_MktMed_Number_3	13	200	0.2	0.5	1.5
68	1000	Run_PrtMed_MktMed_Number_4	14	200	0.2	0.5	1.5
69	1000	Run_PrtMed_MktMed_Number_5	21	200	0.2	0.5	1.5
70	1000	Run_PrtMed_MktMed_Number_6	22	200	0.2	0.5	1.5
71	1000	Run_PrtMed_MktMed_Number_7	23	200	0.2	0.5	1.5
72	1000	Run_PrtMed_MktMed_Number_8	24	200	0.2	0.5	1.5
73	1000	Run_PrtMed_MktMed_Number_9	31	200	0.2	0.5	1.5
74	1000	Run_PrtMed_MktMed_Number_10	32	200	0.2	0.5	1.5
75	1000	Run_PrtMed_MktMed_Number_11	33	200	0.2	0.5	1.5
76	1000	Run_PrtMed_MktMed_Number_12	34	200	0.2	0.5	1.5
77	1000	Run_PrtMed_MktMed_Number_13	41	200	0.2	0.5	1.5
78	1000	Run_PrtMed_MktMed_Number_14	42	200	0.2	0.5	1.5
79	1000	Run_PrtMed_MktMed_Number_15	43	200	0.2	0.5	1.5
80	1000	Run_PrtMed_MktMed_Number_16	44	200	0.2	0.5	1.5
81	1000	Run_PrtMed_MktHig_Number_1	11	200	0.3	0.8	2
82	1000	Run_PrtMed_MktHig_Number_2	12	200	0.3	0.8	2
83	1000	Run_PrtMed_MktHig_Number_3	13	200	0.3	0.8	2
84	1000	Run_PrtMed_MktHig_Number_4	14	200	0.3	0.8	2
85	1000	Run_PrtMed_MktHig_Number_5	21	200	0.3	0.8	2
86	1000	Run_PrtMed_MktHig_Number_6	22	200	0.3	0.8	2
87	1000	Run_PrtMed_MktHig_Number_7	23	200	0.3	0.8	2
88	1000	Run_PrtMed_MktHig_Number_8	24	200	0.3	0.8	2
89	1000	Run_PrtMed_MktHig_Number_9	31	200	0.3	0.8	2
90	1000	Run_PrtMed_MktHig_Number_10	32	200	0.3	0.8	2
91	1000	Run_PrtMed_MktHig_Number_11	33	200	0.3	0.8	2
92	1000	Run_PrtMed_MktHig_Number_12	34	200	0.3	0.8	2
93	1000	Run_PrtMed_MktHig_Number_13	41	200	0.3	0.8	2
94	1000	Run_PrtMed_MktHig_Number_14	42	200	0.3	0.8	2
95	1000	Run_PrtMed_MktHig_Number_15	43	200	0.3	0.8	2
96	1000	Run_PrtMed_MktHig_Number_16	44	200	0.3	0.8	2
97	1500	Run_PrtLow_MktLow_Number_1	11	100	0.1	0.3	1
98	1500	Run_PrtLow_MktLow_Number_2	12	100	0.1	0.3	1
99	2000	Run_PrtLow_MktLow_Number_1	11	100	0.1	0.3	1
100	2000	Run_PrtLow_MktLow_Number_2	12	100	0.1	0.3	1
101	2000	Run_PrtLow_MktLow_Number_3	13	100	0.1	0.3	1
102	2000	Run_PrtLow_MktLow_Number_4	14	100	0.1	0.3	1
103	2000	Run_PrtLow_MktLow_Number_5	21	100	0.1	0.3	1