Multi-Objective Evolutionary Algorithms for the Risk-Return Trade-off in Bank Loan Management

Amitabha Mukerjee [*] Rita Biswas [†] Kalyanmoy Deb [‡]
Amrit P Mathur [§]

February 16, 2002

Abstract

Multi-Criteria Decision Making is an increasingly accepted tool for decision making in management. In this work, we highlight the application of a novel Multi-Objective Evolutionary Algorithm, *NSGA-II* to the risk-return trade-off for a bank loan portfolio manager.

The manager of a bank operating in a competitive environment faces the standard goal of maximizing shareholder wealth. Specifically, this attempts to maximize the net worth of the bank, which in turn involves maximizing the net interest margin of the bank (among other factors, such as non-interest income). At the same time, there are significant regulatory constraints placed on the bank, such as the maintenance of adequate capital, interest-rate risk exposure, etc.

The Genetic Algorithm based technique used here obtains an approximation to the set of Pareto-optimal solutions which increases the decision flexibility available to the bank manager and provides a visualization tool for one of the tradeoffs involved. The algorithm is also computationally efficient and is contrasted with a traditional multi-objective function - the epsilon-constraint method.

Keywords: International Banking, Bank Capital Standards, Multi-Objective Optimization, Genetic Algorithms

[*] Professor, Computer Science and Engg., IIT Kanpur, INDIA, *e-mail: amit@cse.iitk.ac.in*.

[†] Visiting Professor, Finance, Hochschule fur Bankwirtschaft and Associate Professor, University at Albany SUNY, USA *e-mail: biswas@hfb.de*.

[‡] Professor, Mechanical Engineering, Indian Institute of Technology, Kanpur, India *e-mail: deb@iitk.ac.in*.

[§] Dept of Mathematics, IIT Kanpur, India

1 Multi-Objective Decision-making in Banking

1.1 Multi-Objective Optimization vs. Multi-Criteria Decision Making

Multi-Objective optimization is today a well known method for problems in areas such as product design, finance, facility planning, etc. In this work, we propose to apply a novel Multi Objective Evolutionary Algorithm *NSGA-II* (Deb, Agrawal, Pratab, and Meyarivan 2000) to the risk-return tradeoff problem for a bank loan portfolio manager.

Classic research in psychology and Multi-Criteria Decision Making has highlighted many aspects of the human decision-making process which often require the simultaneous consideration of more than one objective function (Zeleny 1982; Steuer 1986; Miettinen 1999). Quite often these objectives may be conflicting so that there is a tradeoff between the criteria, and effective solutions will not be unique, but will lie among a set of "non-dominated solutions" where any particular objective cannot be improved upon without worsening some other. Here we define the basic problem in bank credit management in terms of the multiple objectives of return maximization and risk minimization, and apply a novel evolutionary algorithm for obtaining the non-dominated solution set. We compare our results with the traditional multi-objective method of epsilon-constraints and show how NSGA-II results in a solution that is both computationally efficient and also more disbursed along the solution "front". It must be noted that in both the traditional and the evolutionary methods used, the solutions obtained are only approximations of the exact solutions, which remain unknown.

Consider the problem as one where we are attempting to minimize m objectives $f_1(X), \ldots, f_m(X)$ over an universe U. Given $p, q \in U$, p dominates q if $f_i(p) < f_i(q)$ for all $i \in [1, m]$. A solution p is efficient if for there are no other $X \in U$ such that X dominates p. The set of such efficent solutions are called the pareto-optimal set, or the non-dominated front. One of the problems in multiobjective optimization methods is that the solutions tend to be clustered along a small part of the non-dominated front (Miettinen 1999). Finding a spread of non-dominated solutions permits the decision maker greater flexibility in decision making.

Such situations involving a tradeoff arise quite frequently - for example, industrial product design may consider a tradeoff between the life of a product and its cost, or a firm may consider a tradeoff between shareholder wealth and societal benefits (see the classic (Zeleny 1982) for many well-characterized examples). In traditional optimization practice, it is customary to unify or *scalarize* these disparate objectives (for example, using lagrangian weighting) into a single objective and then use normal single-function optimization procedures (see (Ehrgott and Gandibleux 2000) for an annotated bibliography). If we wish to obtain a number of points on the non-dominated front, this procedure requires as many runs as the number of points. On the other hand, multi-point searches such as Genetic Algorithms obtain many points on the non-dominated front simultaneously. Also, even for single objective optimization, in problems like this, where the objective function may be non-linear, the solution space is often hard to characterize, and the derivative may be unknown, so that exact methods are not available or computationally intractable. A good review of modern methods in traditional optimization may be found in (Miettinen 1999).

The human decision maker often considers these tradeoffs in an implicit manner. The process of explicitly formulating the problem as a multi-criteria problem, and the subsequent possibilities for visualization of the non-dominated solutions over a wider range of the solution space permits more lucid decision-making regarding the trade-offs, and also the queries raised by other interveners in the decision process. Also, sometimes small sacrifices in one objective may lead to tremendous improvements in the other - or constraints may be observed which were so far external to the optimization process, thus clarifying the optimization task further. An excellent discussion of the human and management aspects of the decision making process may be found in (Zeleny 1982)

Genetic algorithms (GAs) are search and optimization algorithms inspired by the principles of natural evolution. Conceived by John Holland in early Sixities (Holland 1975; Goldberg 1989), GAs have been applied to a wide range of problems since then, including, increasingly, applications in multi-objective optimization. A recent compilation of of various multi-objective genetic algorithms can be found in (Deb 2001; Ehrgott and Gandibleux 2000; Coello 1999). GAs gain their efficiency in search and optimization by searching simultaneously along a set of threads (called a *population*), and scaling up the efficiency of the search by exchanging information between these search threads (called *crossover*) (Goldberg 1989). While theoretical results on the effectiveness of such methods are limited (e.g. special functions as in (Vose 1999)), practical benefits have long endeared these methods to a wide group of optimization practitioners. Multi-objective evolutionary algorithms (MOEA's) provide the following advantages over traditional multi-objective tools:

- Since the search front in an evolutionary algorithm typically involves a population, a number of points on or near the pareto-optimal front can be obtained in a single simulation as opposed to point-by-point methods in traditional methods (Rakowska, Haftka, and Watson 1991).
- In the novel approach NSGA-II used here, specific care is taken so that the non-dominated points are not all bunched up in a small part of the front. In traditional MO optimization methods, a uniform set of weight vectors (or ϵ-vectors) does not guarantee finding a uniformly distributed set of Pareto-optimal solutions.
- No a priori knowledge is required of minimizer functions - in fact, the function need not even be analytic.
- Problems of arbitrary dimensionality (any number of objectives) can be handled.

The methodology of the NSGA-II algorithm and its advantages with respect other Multi-Objective Evolutionary Algorithms are highlighted in section 2.

The growth of multi-objective optimization methods in a broad range of disciplines has received considerable impetus in recent years from interdisciplinary work that in the areas of Multi-Criteria Decision Aid and Decision Making (MCDA / MCDM), which highlights the organizational and human difficulty of identifying a uniform set of objective criteria for any optimization task. In contrast, policy aspects (e.g. one banker has more experience with high-risk loans and has a higher "risk appetite" than another) result in differing subjective evaluations - and whenever these criteria are conflicting -

i.e. no single solution optimizes all the criteria - there must be some trade-off, and these methods highlight the decision-making benefits of presenting pareto-optimal solutions in such cases.

1.2 The Multi-Objective Bank Loan Optimization Problem

The failure of a bank imposes significant negative externalities (costs on other economic agents) and this is what makes banks unique (James 1987; Fama 1985). This also makes a case for society to regulate banks more stringently than other firms, especially with respect to safety and soundness, monetary policy, credit allocation, consumer and investor protection, and entry and chartering. On the other hand, like any other firm, the bank also maximizes shareholder wealth. This leads to some unique tradeoffs in bank management. For instance, in order to minimize interest rate risk, banks must match maturities / duration (average maturity) of assets and liabilities. However, usually with a positively sloped yield curve, lending long term maximizes interest income and borrowing short term minimizes interest expense; thus balancing average maturities is contrary to maximizing net interest margin. Considering the entire balance sheet of the bank, the situation gets more complex. Depending on the interest rate risk exposure (and other sources of risk), the bank is required to hold a certain level of capital. This again, is contrary to maximizing shareholder wealth.

In this paper however we focus on a simpler tradeoff, i.e. the risk-return tradeoff faced in loan portfolio management. We use a simpler form of the return model, and show how NSGA-II results in computationally efficient, intuitively valid non-dominated solution points.

2 Genetic Algorithms and Multi-Objective Optimization

Over the past decade, a number of multi-objective evolutionary algorithms (MOEAs) have been suggested (Fonseca and Fleming 1993; Horn, Nafploitis, and Goldberg 1994; Knowles and Corne 1999; Srinivas and Deb 1995; Zitzler and Thiele 1998; Ranjithan, Chetan, and Dakshina 2001). The primary reason for this is their ability to find multiple non-dominated solutions in a single simulation. Since the principal reason why a problem has a multi-objective formulation is because it is not possible to have a single solution which simultaneously optimizes all objectives, an algorithm that gives a large number of alternative solutions lying on or near the Pareto-optimal front is of great practical value.

In the following, we discuss one such multi-objective EA – Non-dominated sorting genetic algorithm (NSGA-II), which is used in the simulation studies of this paper. The details of three main operations – non-dominated sorting, density estimation and the crowded comparison operator – are given below. More details may be found the original study (Deb, 2001). It will suffice here to mention that the non-domination approach used here is computationally the fastest procedure reported so far. The density estimation procedure is also computationally fast, involving objective-wise comparisons of solutions, thereby making the procedure easily extendable to higher objective

problems. The crowded comparison operator requires minimal change in an existing pair-wise tournament selection opeartor yet allows NSGA-II to achieve both tasks of progressing towards the Pareto-optimal front and maintaining a diverse set of solutions.

2.1 NSGA-II Algorithm: The main loop

Initially, a random parent population P_0 is created. The population is sorted based on the non-domination. Each solution is assigned a fitness equal to its non-domination level (1 is the best level). Thus, minimization of fitness is assumed. Binary tournament selection, recombination, and mutation operators are used to create a child population Q_0 of size N. From the first generation onward, the procedure is different. The elitism procedure for $t \geq 1$ and for a particular generation is shown in the following:

$R_t = P_t \cup Q_t$	combine parent and children population
$\mathcal{F} =$ `fast-nondominated-sort(`R_t`)`	$\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \ldots)$, all non-dominated fronts of R_t
$P_{t+1} = \emptyset$	
until $\lvert P_{t+1} \rvert < N$	till the parent population is filled
`crowding-distance-assignment(`$\mathcal{F}_i$`)`	calculate crowding distance in $\mathcal{F}_i$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i$	include i-th non-dominated front in the parent pop
Sort($P_{t+1}, \prec_n$)	sort in descending order using $\prec_n$
$P_{t+1} = P_{t+1}[0 : N]$	choose the first N elements of P_{t+1}
Q_{t+1} = `make-new-pop(`P_{t+1}`)`	use selection, crossover and mutation to create a new population Q_{t+1}
$t = t + 1$	

First, a combined population $R_t = P_t \cup Q_t$ is formed. The population R_t will be of size $2N$. Then, the population R_t is sorted according to non-domination. The new parent population P_{t+1} is formed by adding solutions from the first front till the size exceeds N. Thereafter, the solutions of the last accepted front are sorted according to $\prec_n$ and a total of N solutions are picked. This is how we construct the population P_{t+1}. This population of size N is now used for selection, crossover and mutation to create a new population Q_{t+1} of size N. It is important to note that we use a binary tournament selection operator but the selection criterion is now based on the niched comparison operator $\prec_n$.

2.2 A Fast Non-dominated Sorting Procedure

In this approach, every solution in a GA population is checked with a partially filled population for domination. To start with, the first solution from the population is kept in a set P'. Thereafter, each solution p (the second solution onwards) is compared with all members of the set P' one by one. If the solution p dominates any member q of P', then solution q is removed from P'. This way non-members of the non-dominated from get deleted from P'. Otherwise, if solution p is dominated by any member of P', the solution p is ignored. If solution p is not dominated by any member of P', it is entered in P'. This is how the set P' grows with non-dominated solutions. When

all solutions of the population is checked, the remaining members of P' constitute the non-dominated set.

```
fast-nondominated-sort(P)
```

$P' = \{1\}$	include first member in P'
for each $p \in P \wedge p \notin P'$	take one solution at a time
$P' = P' \cup \{p\}$	include p in P' temporarily
for each $q \in P' \wedge q \neq p$	compare p with other members of P'
if $p \prec q$, then $P' = P' \backslash \{q\}$	if p dominates a member of P', delete it
else if $q \prec p$, then $P' = P' \backslash \{p\}$	if p is dominated by other members of P', do not include p in P'

To find other fronts, the members of P' will be discounted and the above procedure is repeated.

2.3 Density estimation

To get an estimate of the density of solutions surrounding a particular point in the population we take the average distance of the two points on either side of this point along each of the objectives. This quantity $i_{distance}$ serves as an estimate of the size of the largest cuboid enclosing the point i without including any other point in the population (we call this the *crowding distance*). In Figure 1, the crowding distance of the i-th solution in its front (marked with solid circles) is the average side-length of the cuboid (shown with a dashed box). The following algorithm is used to calculate the

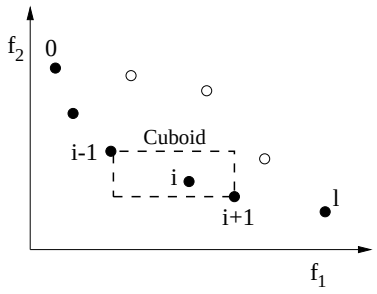


Figure 1: The crowding distance calculation is shown

crowding distance of each point in the set $\mathcal{I}$:

```
crowding-distance-assignment(I)
```

$l = \|\mathcal{I}\|$	number of solutions in $\mathcal{I}$
for each i, set $\mathcal{I}[i]_{distance} = 0$	initialize distance
for each objective $j \in [1, m]$	
$\mathcal{I}$ = sort($\mathcal{I}, j$)	sort using each objective value
$\mathcal{I}[1]_{distance}$ = $\mathcal{I}[l]_{distance}$ = ∞	so that boundary points are always selected

for $i = 2$ to $(l-1)$ for all other points
$\mathcal{I}[i]_{distance} = \mathcal{I}[i]_{distance} + (\mathcal{I}[i+1].j - \mathcal{I}[i-1].j)$

Here $\mathcal{I}[i].j$ refers to the j-th objective function value of the i-th individual in the set $\mathcal{I}$. The complexity of this procedure is governed by the sorting algorithm. In the worst case (when all solutions are in one front), the sorting requires $O(mN \log N)$ computations.

2.4 Crowded comparison operator

The crowded comparison operator ($\prec_n$) guides the selection process at the various stages of the algorithm towards a uniformly spread-out Pareto-optimal front. Let us assume that every individual i in the population has two attributes.

1. Non-domination rank (i_{rank})
2. Local crowding distance ($i_{distance}$)

We now define a partial order $\prec_n$ as :
$i \prec_n j$ if ($i_{rank} < j_{rank}$) or ((i_{rank} = j_{rank}) and ($i_{distance} > j_{distance}$))
That is, between two solutions with differing non-domination ranks we prefer the point with the lower rank. Otherwise, if both the points belong to the same front then we prefer the point which is located in a region with lesser number of points (the size of the cuboid inclosing it is larger).

Let us now look at the complexity of one iteration of the entire algorithm. The basic operations being performed and the worst case complexities associated with are as follows:

1. Non-dominated sort is $O(mN^2)$,
2. Crowding distance assignment is $O(mN \log N)$, and
3. Sort on $\prec_n$ is $O(2N \log(2N))$.

As can be seen, the worst-case complexity of the above algorithm is $O(mN^2)$. where N is the population size and the number of objectives is m. At most N points along the pareto-optimal front are identified by the algorithm so the complexity of the algorithm is directly a function of the accuracy to which the pareto-optimal front is to be identified (N).

3 Modeling the Risk-return tradeoff

What are the decision variables for the loan portfolio manager? The primary decision made by the portfolio manager is regarding the allocation of loans in the different categories, or *loan allocation*. In addition, he may also make decisions regarding the interest rates to be charged for loans in different categories. In particular, the quantity of loans demanded in a particular category may be influenced by the interest rate charged. This latter consideration is often not very important in traditional banking practice

where in the short run, this demand curve for loans is considered to be inelastic - i.e. the volume of loan applications remains unchanged irrespective of the interest charged.

In the first model below, the decision variables for the portfolio manager is the loan allocation. Given N loan categories or tranches (e.g. "AAA", "BBB", etc.), with interest rates R_i what fraction of loans X_i should be allocated in each category?

In the second model, the demand for loans is not inelastic, and the total loan applications received in category i is assumed to have an upper bound, which is a negatively sloped function of the interest rate R_i. Thus the loan manager faces a negatively sloped demand curve for loans, and he needs to allocate X_i subject to this additional constraint. However, it is clear that if X_i is far below this bound, then it is more efficient for him to charge a higher interest R_i for the category i, such that his return is higher for the same allocation X_i. This implies that in an equilibrium model the X_i's will settle to a value close to the bound imposed by the loan demand function. This in turn implies that the loan manager's primary decision variable is the interest rate, which then, given a loan demand function, determines the loan allocation X_i.

3.1 Traditional models - allocation under inelastic demand

In traditional banking each loan was considered separately, based on their credit-risk worthiness, evaluated on their credit history, default risk, quality of return and other traditional measures. Based on Portfolio Theory, one assumes that the loan portfolio manager applies the idea of diversification to the Loan portfolio as well.

The Modern Portfolio Theory (MPT) (Elton and Gruber 1998), model of Credit Risk (Saunders 1999) assumes that individual asset returns are normally distributed. An alternate assumption may be that loan portfolio managers exhibit quadratic utility preferences. Both these assumptions lead to models for the return and risk on a given loan portfolio. One model of risk, based on the Black-Scholes-Merton model, considers the borrower's incentive to pay back as positive so long as the value of his assets beyond a certain date exceeds the loan amount. This leads to a "distance from default" model of Credit Risk based on the difference between total assets and loan amount as a function of the annual volatility in value or the standard deviation in the firms' equity. Applying such metrics to a portfolio leads to very complex models requiring individual borrower information. In this work, we adopt a simpler model of portfolio credit risk based on the standard deviation of the return over the entire portfolio.

Return

Mean return on portfolio R_p = weighted average of the return R_i on the individual loan categories = $\Sigma X_i R_i$, where X_i is the proportion of total loans allocated to the i-th category. The return is determined by other factors such as the prime rate, processing costs, expected default rate, probability of unexpected losses, etc. In our partial equilibrium analysis, we assume these factors to be external to the optimization task, and therefore the return R_i is the interest rate charged on the loan category i less some constant, and thus maximizing the return is equivalent to maximizing the interest income.

Table 1: Interest rates charged for loan risk categories

Loan Cat		Benchmark R_i	More Risk-Averse	Less Risk-Averse
1	(AAA)	4.0	3.75	4.25
2	(AA)	4.5	4.25	4.5
3	(A)	5.5	5.5	5.25
4	(BBB)	7.0	7.25	6.75
5	(BB)	8.0	8.25	7.5
6	(B)	8.5	9.0	8.0
7	(CCC)	10.0	11.0	9.25

Table 2: Standard Deviation σ_i on returns of loans in category i

	1 (AAA)	2 (AA)	3 (A)	4 (BBB)	5 (BB)	6 (B)	7 (CCC)
σ_i	0.06	0.17	1.56	3.37	8.63	12.84	22.67

Risk

A number of measures are today used for measuring risk in a particular credit portfolio (Altman and Saunders 1997; Leland 1998; Shearer 1997) Risk is assumed to be correlated to the standard deviation of returns on the portfolio:

$$\sigma_p^2 = \Sigma({X_i}^2{\sigma_i}^2) + \Sigma_i, \Sigma_j(X_i X_j \sigma_{ij})$$

where σ_{ij} is usually approximated as correlation $\rho_{ij}\sigma_i\sigma_j$. If the correlation is negative, say, then the portfolio manager can increase the corresponding allocation to reduce overall risk exposure.

Note the additional constraint

$$\Sigma_i X_i = 1$$

arising from the fact that the X_i are fractional allocations of the total sum.

Multi-Objective Optimization Model

The optimization model for the allocation under inelastic demand case is as follows:

- $f_1()$: Maximize mean return on portfolio $R + p$
- $f_2()$: Minimize variance on return σ_p
- Variables: assets X_i in each set of credit risk classes with given interest rates R_i (return).
- $R_p = \Sigma_i X_i R_i$
- ${\sigma_p}^2 = \Sigma_i {X_i}^2 {\sigma_p}^2 + \Sigma_i \Sigma_j X_i X_j \rho_{ij} \sigma_i \sigma_j$,
  where ρ_{ij} = correlation between returns on i-th and j-th assets.

Table 3: Coefficient of correlation ρ_{ij} matrix

Loan Cat	1 (AAA)	2 (AA)	3 (A)	4 (BBB)	5 (BB)	6 (B)	7 (CCC)
1 (AAA)	1	0.45	0.45	0.45	0.15	0.15	0.15
2 (AA)	0.45	1	0.45	0.45	0.15	0.15	0.15
3 (A)	0.45	0.45	1	0.45	0.15	0.15	0.15
4 (BBB)	0.45	0.45	0.45	1	0.15	0.15	0.15
5 (BB)	0.15	0.15	0.15	0.15	1	0.35	0.35
6 (B)	0.15	0.15	0.15	0.15	0.35	1	0.35
7 (CCC)	0.15	0.15	0.15	0.15	0.35	0.35	1

For developing the test case we use data from the CreditMetrics Technical Document [1]. Seven loan categories (AAA, AA, A, BBB, BB, B, CCC) were used, with interest rates, standard deviation of returns, and cross-correlations as given in tables 1, 2, and 3.

GA Run parameters

All the runs used binary version of NSGA-II with 7 bits alloted to each variable making each solution a 49 bit string. The mutation rate was fixed at 0.02 (1/49) and the crossover rate was 0.9. All the simulations were run for 100 generations.

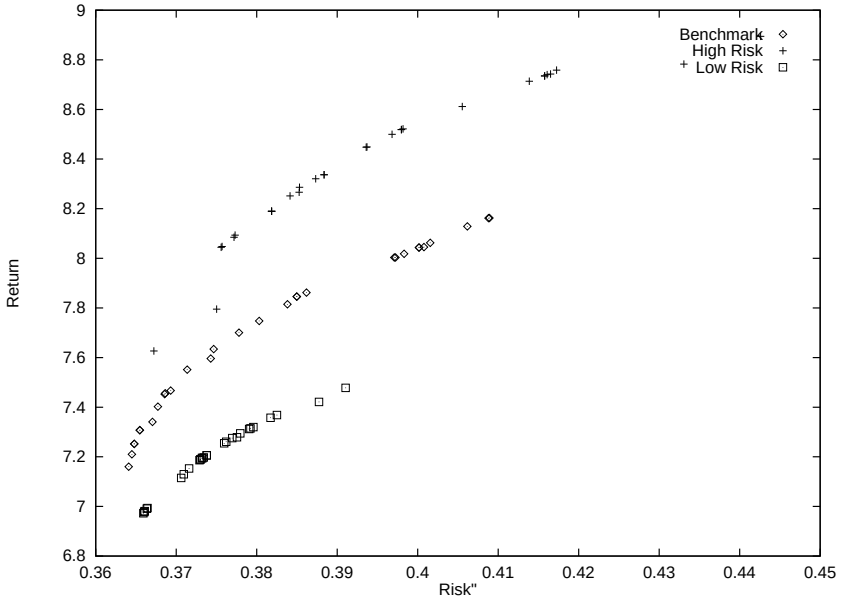
Before evaluating the fitness, the x-values of each soluion was normalized to take care of the $\Sigma_i X_i = 1$ constraint.

3.2 Elastic Loan Demand Model

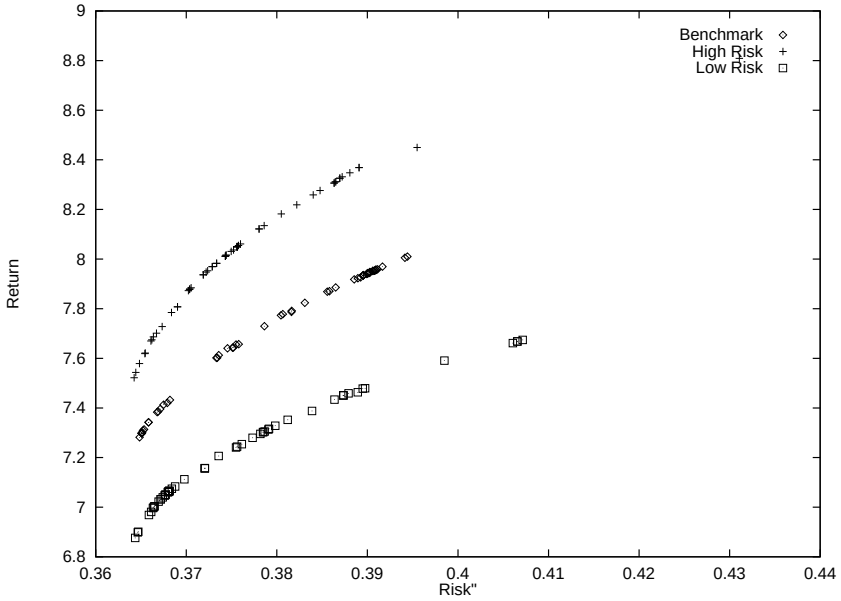
In this model, we assume that the amount of loan applications received in a given loan category is a function of the interest rate charged. Typically this loan demand function is negatively-sloped - i.e. the demand falls as higher interests are charged. Traditionally loan managers have not been concerned with the loan elasticity since demand for loans typically outstripped supply and applicants were available at most realistic interest rates. However, this results in a skew in performance - loan officers charging higher interest rates are modeled with better returns since this does not reflect on volumes. Furthermore, with increased access to information and mobile assets, and also with risk-based loan pricing at banks, the customer may increasingly prefer to go to cheaper sources of loans.

However, the precise structure of this function is a matter of some uncertainty. This is particularly so because empirical identification of loan demand or supply functions is made difficult since only the actual intersection of these curves is observable, and also because of the large number of extraneous factors (such as inflation rate, time, GNP, etc.) which affect this function. However, there is some empirical evidence for a linear model of the loan demand as a function of the interest rate and the GNP (Hülsewig, Winkler, and Worms 2001), based on which we have also assumed a linear function for the loan interest rate.

[1] CreditMetrics Technical Document, JP Morgan, April 1997



(a) Model 1 with population size 30 (30 points)



(b) Model 1 with population size 60 (60 points)

Figure 2: Results for model 1 (Inelastic Demand for Loans). N = 30, 60

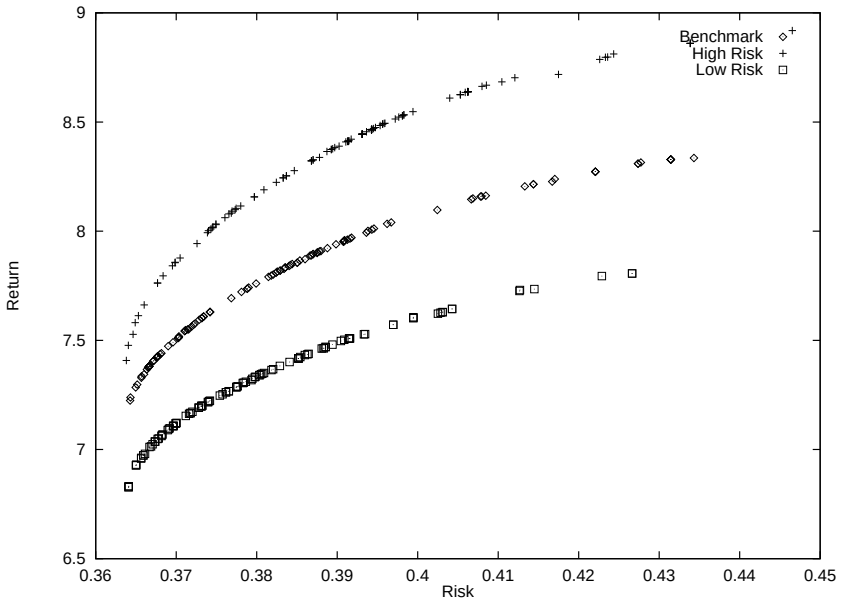


Figure 3: Results for Model 1 (Inelastic Demand for Loans) N=120 (120 points)

As observed earlier, the demand function provides the bank with the maximum applications possible in a loan category i charging interest rate R_i. The loan portfolio manager can still optimize his allocations in each category so as to optimize his risk-return tradeoff. However it is clear that a better return can be obtained in those categories where the actual allocation is less than the demand - here the interest rate can be raised a bit until the demand drops to precisely that level, thus increasing return for the same allocation. In an equilibrium situation the loan manager would have adjusted his interest rates to match this, and hence we simplify the model by keeping only the interest rates as the decision variables for the loan manager - given these interest rates and a loan demand function $X_i = f(R_i)$, the actual allocations X_i are directly identified. The results for this model are presented for population sizes of 30 and 60 in figures 4 and 5 respectively.

For the NSGA-II run, the independent variables were the interest rates R_i and from these the allocation X_i is calculated based on a demand function $Q_i = \frac{50}{0.25}R_i + 200$ where Q_i is the quantity allocated in loan category i and the corresponding allocation is obtained by normalizing this: $X_i = Q_i/\Sigma Q_i$. The initial values for the interest rates R_i are randomly seeded from the range [4,12] to obtain the results presented.

3.3 Comparison with traditional MO approaches

Traditional Multi-Objective optimization techniques essentially reduce the multi-objective problem to a single objective problem in such a manner that each optimum point is

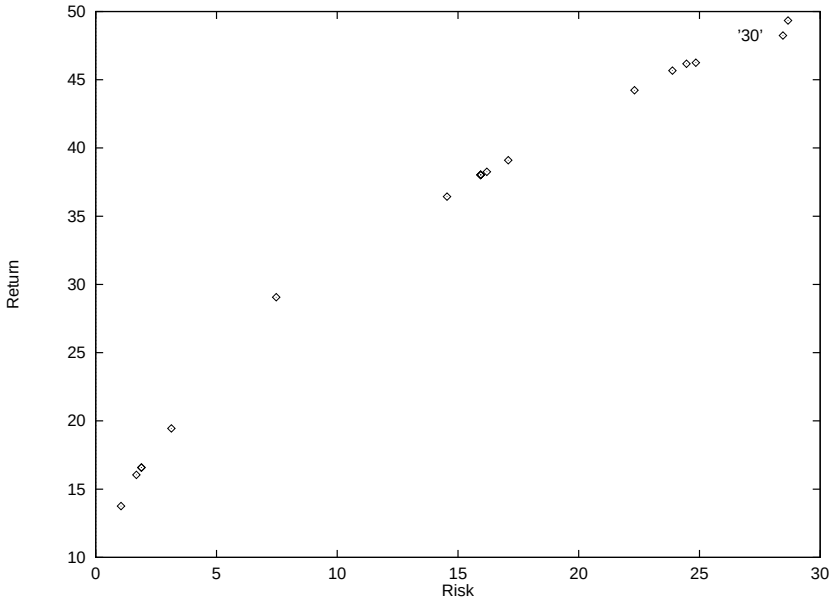


Figure 4: Results for model 2 (Elastic Loan Demand), N=30 (30 points)

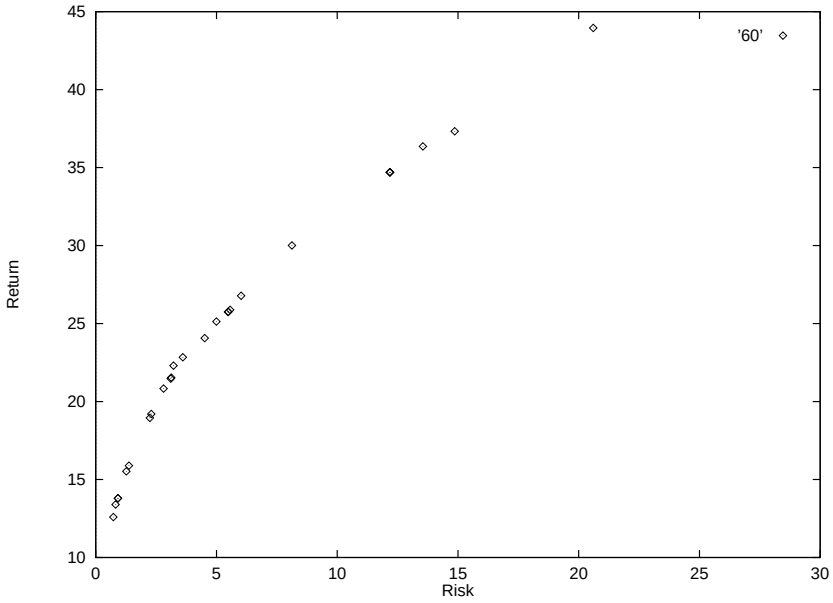


Figure 5: Results for model 2 (Elastic Loan Demand), N=60

guaranteed to lie on the pareto-optimal front. A separate optimization run is needed for each desired point on the front. The algorithm is even slower if there are more than two objectives, Furthermore, it is difficult to pre-determine how these points will be spread out along the front, and the main problem in most of these approaches (and in many MOEA's) is that the points are bunched up over a small part of the pareto-frontier.

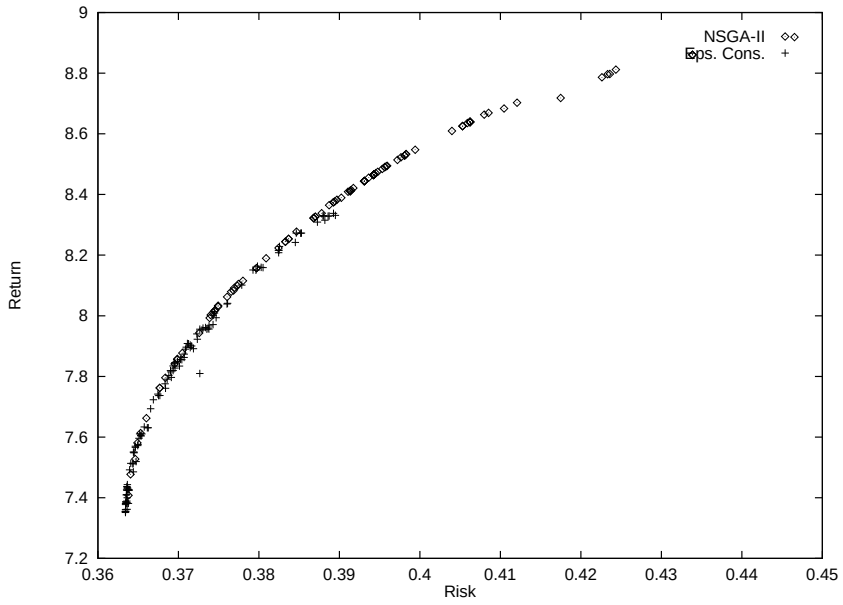


Figure 6: Comparative pareto-optimal fronts obtained by GA and by epsilon method .

One of the popular non-evolutionary methods is the epsilon-constraint approach, which converts the problem into a one-dimensional one by re-formulating some of the objectives as constraints. For example, for a D-dimensional problem, $D-1$ objectives are reduced to constraints. Thus in our case, we can minimize $f_1()$ s.t. $f_2() < \epsilon_k$ where a different ϵ_k is chosen for each pareto- point. Unlike some other traditional methods, this approach does not require the normalization of the objectives, but the epsilon vectors need to be chosen carefully, often with partial hindsight. In our case, we have chosen the a set of 10 epsilon vectors from the range of Pareto points obtained using NSGA-II which results in a far more uniform distribution than would have happened in ab initio search. Even then, one can see that most of the solutions are bunched towards the lower-risk end of the spectrum. We have used a simple single-objective GA (and not NSGA-II) as our optimization method.

Note that the pareto-frontier itself is quite strongly validated by the high degree of overlap between these two approaches.

4 Conclusion

In this work, we have presented the application of a novel tool for multi-objective optimization problems in finance. The method permits one to have many criteria, does not require any prior knowledge of the optimizer functions, and can generate a complete pareto-optimal front in a single simulation run.

The problem addressed is that of the risk-return tradeoff in bank loan portfolio management. This problem is well known in the literature as one requiring a possible multi-objective formulation. One of the models for converting it to a single objective is to hypothesize a tsingle combination - e.g. return per unit risk - which is to be maximized. One such measure is the Sharpe Ratio (Saunders 1999) - maximize $\frac{(R_p - R_f)}{\sigma_p}$, where R_f is the risk-free interest, and σ_p is the standard deviation on return for the portfolio. Any such combination however, is in the final analysis subject to discussion and criticism by other decision makers and it is always an useful alternative to have a number of solutions, and to be able to visualize and discuss these (at least in the two-objective scenario) which is what is provided by this tool.

In the coming years, as banks enter an era of even tighter regulatory constraints, and also face greater competitive pressures from wider dissemination of information, new modalities like internet banking, and greater fluidity of assets worldwide, it will be necessary to deploy more and more decision making tools and to suit them to match the needs of a wider group of people. At the same time, it is clear that the tool developed here, which is standalone, will need to be used in more interactive ways by the Decision Makers so that situations involving more than two variables can be sliced along appropriate sections for better visualization, thus empowering the DM with new tools which were hitherto not within reach.

One of the problems that we hope to focus on next is one of immediate concern to the banking community. One of the modern requirements in banking practice is to minimize the interest-rate risk exposure of the bank. Ideally, in order to maximize their immediate net worth, banks want to minimize the amount of capital they hold and would like to have no constraints on the amounts of interest-rate sensitive assets (IRAs) and interest-rate sensitive liabilities (ISLs) they have on their balance sheets. In practice, since interest rates rise with duration, it is most desirable to hold long-term loans (IRAs) and short term deposits (ISLs) - but this is exactly the kind of mismatched portfolio that would be subject to severe risk under interest rate fluctuations. We believe the multi-objective MOET approach may be one of the best tools to address this type of problem in practice.

Reference

Altman, E. and Saunders, A. (1997). Credit risk measurement: Developments over the last twenty years. *J. of Banking and Finance*, 1721–1742.

Coello, C. A. C. (1999). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems. An International Journal*, *1*(3), 269–308.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Wiley, Chichester, UK.

Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. Technical report KanGAL report 200001, Indian Institute of Technology, Kanpur, India.

Ehrgott, M. and Gandibleux, X. (2000). An Annotated Bibliography of Multi-objective Combinatorial Optimization. Technical report Nr. 62/2000, Dept. Mathematics, Univ. Kaiserslautern, Germany.

Elton, E. and Gruber, M. J. (1998). *Modern Portfolio Theory and Investment Analysis* (6 edition). John Wiley and Sons.

Fama, E. (1985). What's different about banks? *J. of Monetary Economics*, *15*, 29–39.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In *Forrest, S., editor, Proceedings of the Fifth International Conference on Genetic Algorithms*, pages Morgan Kauffman, San Mateo, California. 416–423.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, Mass. : Addison-Wesley Pub. Co.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: MIT Press.

Horn, J., Nafploitis, N., and Goldberg, D. E. . (1994). A niched pareto genetic algorithm for multi-objective optimization. In *Michalewicz, Z., editor, Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 82–87. IEEE Service Center, Piscataway, New Jersey.

Hülsewig, O., Winkler, P., and Worms, A. (2001). Monetary policy and long-run bank loan supply and demand in germany. Unpublished Manuscript.

James, C. (1987). Some evidence on the uniqueness of bank loans. *J. of Financial Economics*, *19*, 217–235.

Knowles, J. and Corne, D. (1999). The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimisation. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 98–105. Piscataway: New Jersey: IEEE Service Center.

Leland, H. (1998). Agency costs, risk management and capital structures. *J. of Finance*, 1213–1242.

Miettinen, K. (1999). *Nonlinear multiobjective optimization*. Boston: Kluwer.

Rakowska, J., Haftka, R. T., and Watson, L. T. (1991). Tracing the efficient curve for multi-objective control-structure optimization. *Computing Systems in Engineering, 2*, 461–471.

Ranjithan, S. R., Chetan, S. K., and Dakshina, H. K. (2001). Constraint method-based evolutionary algorithm (CMEA) for multiobjective optimization. In *Proceedings Evolutionary Methods in Optimization*, pages 299–313.

Saunders, A. (1999). *Credit Risk Management*. John Wiley and Sons.

Shearer, A. (1997). Pricing for risk is the key in commercial lending. *American Banker*.

Srinivas, N. and Deb, K. (1995). Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation, 2*(3), 221–248.

Steuer, R. (1986). *Multiple Criterion Optimization: Theory, Computation, and Application*. New York: Wiley.

Vose, M. D. (1999). *Simple Genetic Algorithm: Foundation and Theory*. Ann Arbor, MI: MIT Press.

Zeleny, M. (1982). *Multiple Criteria Decision Making*. McGraw-Hill Book Company: New York.

Zitzler, E. and Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms—a comparative case study. In *In Eiben, A. E., Bäck, T., Schoenauer, and M. and Schwefel, H.-P., editors, Parallel Problem Solving from Nature, V*, pages 292–301. Springer, Berlin, Germany.