Multi-objective design based on symbolic computation and its application to hard disk slider design

Hitoshi Yanami

Received on September 16, 2009 / Revised on October 4, 2009

Abstract. We propose a new approach to tackling multi-objective optimization problems. Our method uses symbolic computation called quantifier elimination. From experimental data we first make an approximated polynomial model for each objective function and then solve a first-order formula related to those functions to find the feasible region, which contains information on the Pareto optimal front. Our approach has an advantage over classical numerical optimization methods that return only one optimal point at a time. Furthermore, by introducing into a formula an adjusting point and a new variable that restricts the parameter ranges we can visualize how the objective functions locally behave. This idea leads us to a new criterion for measuring the robustness against production tolerance. We also show how our methods are applied to the design problem of a hard disk slider.

Keywords. symbolic computation, quantifier elimination, multi-objective optimization

1. Introduction

In modern manufacturing industry multi-objective optimization is a common design method. Design problems in the engineering field are often formulated into a multi-objective optimization (MOO) problem. Although various types of research for MOO have been carried out, the most general case is still open.

One of the main difficulties in dealing with them arises from the fact that the cost values are not totally ordered as in single-objective optimization (SOO). Thus the optimal solution for an MOO problem is not a single point but forms a set of points called a Pareto optimal front. Approaches to these problems are roughly categorized into two types; one is to find an optimal point and obtain information on the Pareto set by collecting such points, and the other to try to find the Pareto front itself.

Classical attempts to MOO such as the weighted sum strategy and the ϵ-constraint method have transformed it into SOO [31, 18]. Other than those, various meta-heuristic algorithms have been proposed to tackle MOO problems. Some of them have been realizable only recently owing to the improvements in CPU performance and memory capacity. Evolutionary algorithms, a typical class of such approaches, have been attracting more attention as a practical method of tackling MOO problems and showing successful results. Particle swarm optimization (PSO), a new class of evolutionary algorithms, has been used to solve problems in control theory [20, 23].

In this paper we propose an algorithm for obtaining a Pareto optimal set using a symbolic algorithm called quantifier elimination by using polynomial models for objective functions. Symbolic algorithms can deal with constraints expressed by polynomials and return an exact formula for the feasible region, containing complete information on its Pareto front. This is a great advantage over numerical approaches that compute only an optimal point during an optimization process and require many simulation trials for obtaining Pareto data. We also propose a criterion for measuring production tolerance, which can be used to order the points that are not comparable with regard to their objective function values.

Our method has been applied to the design problem of a hard disk slider, a thin, nearly squared flat part attached to an actuator arm. On the top of a slider is a magnetic head that writes on or reads from a disk binary information. The surface of a slider, designed by nanometer-scale, is shaped to stabilize the head. The design problem of a hard disk slider is to determine the shape, or the pattern, of the slider's surface. A typical way of designing a slider is as follows. A designer first draws a basic geometrical shape of the surface and sets a set of parameters to be optimized as well as their ranges. For a set of real values for the parameters a simulator computes various physical values as to the slider's relative position to the disk. We treat the simulator as a black box.

To obtain high performance as well as durability, it is very important to control the relative position between the slider head and the disk. For example the distance between them, called the flying height, is one of the most important indicators that affect the quality of a hard disk drive because a head-disk contact might cause a crash. Relative angles between the slider and the disk such as the pitch and roll angles are also to be controlled. These performance indicators form a set of objective functions, or cost functions in the design problem of a hard disk slider.

Here is the structure of the paper: In Section 2 we set a formulation of MOO problems. We sketch several known methods of tackling MOO in Section 3. In Section 4 we overview quantifier elimination followed by our approach based on QE in Section 5. In Section 6 our approach is applied to the design problem of a hard disk slider. Section 7 concludes the paper.

2. Problem Formulation

Let us give a formal setting of a multi-objective optimization problem. We follow the notations used in [24].

Let $\mathbf{f} = (f_1, f_2, \ldots, f_n)^{\mathrm{T}}$, where $\mathbf{v}^{\mathrm{T}}$ means the transpose of $\mathbf{v}$, be a column vector of objective functions that measure some performance. These functions are to be optimized in a multi-objective optimization problem. From now on we assume that each function $f_i = f_i(\mathbf{p})$, where $\mathbf{p} = (p_1, p_2, \ldots, p_m)^{\mathrm{T}}$ is a list of parameters, takes nonnegative real values and a lower value means higher performance. The parameter vector $\mathbf{p}$ is supposed to run over a prescribed subset $\mathcal{P} \subset \mathbb{R}^m$ of the m-dimensional Euclidean space, defined as $\mathcal{P} = \{\mathbf{p} \in \mathbb{R}^m | \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\}$ for some $\mathbf{c}$. The image $\mathcal{F} = \{\mathbf{f}(\mathbf{p}) \in \mathbb{R}^n | \mathbf{p} \in \mathcal{P}\}$ is called the feasible region of $\mathbf{f}$ with respect to $\mathcal{P}$.

A multi-objective optimization problem can be represented by

$$\text{minimize } \mathbf{f}(\mathbf{p}) \text{ subject to } \mathbf{p} \in \mathcal{P}.$$

It is exceptional that one can optimize all the objective functions simultaneously. In almost all cases there is a trade-off relation between the objective functions; improving one cost function should result in worsening another.

To state such a situation formally let us give one more definition. Let n be a positive integer. We define a relation $\prec$ on $\mathbb{R}^n \times \mathbb{R}^n$ as

$$\mathbf{a} \prec \mathbf{b} \overset{\text{def}}{\Longleftrightarrow} \forall_{i \in \{1,2,\ldots,n\}} a_i \leq b_i \ \wedge \ \mathbf{a} \neq \mathbf{b}$$

for $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{R}^n$ and $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{R}^n$. Thus $\mathbb{R}^n$ is a partially ordered set with respect to $\prec$. Note that it is a total ordering only when $n = 1$. We say $\mathbf{a}$ dominates $\mathbf{b}$ when $\mathbf{a} \prec \mathbf{b}$.

We call the set of points in $\mathcal{F}$ that are not dominated by any other points the Pareto optimal front. In an MOO problem we are interested in the Pareto front as well as the parameters corresponding to the front.

In the next section we sketch some of the well-known methods of obtaining information on the Pareto optimal front. Some algorithms can collect a useful data on a Pareto set while others focus on a point or two on the Pareto optimal front.

3. Multi-Objective Optimization

There have been various types of approaches to tackling multi-objective optimization problems. We show a brief sketch for some of these optimization methods. The notations in the preceding section are also used here.

3.1. Scalarization

A well-known method of dealing with multi-objective optimization is to reduce the problem to single-objective optimization with the so-called weighted sum strategy [31].

Let $\mathbf{f} = (f_1, f_2, \ldots, f_n)^{\mathrm{T}}$ be the objective functions of a given multi-objective optimization problem and $\mathbf{w} = (w_1, w_2, \ldots, w_n)^{\mathrm{T}}$ an n-dimensional vector with $w_i \geq 0$ for $i = 1, \ldots, n$. The vector $\mathbf{w}$ indicates the engineer's preference as to which functions they think of as more important. By computing the inner product of $\mathbf{w}$ and $\mathbf{f}$ we obtain a single objective function

$$f = f_{\mathbf{w}} = \mathbf{w}^{\mathrm{T}} \cdot \mathbf{f} \ .$$

By optimizing the function f one expects to find a point on the Pareto front of the original problem.

When the Pareto front is convex the optimal point is the contact point of the feasible region $\mathcal{F}$ as a hyperplane tangent to $\mathbf{w}$ approaches from the origin. Repeating this process for various weight vectors may reveal a rough shape of the Pareto optimal front.

This approach suffers from a few drawbacks. First, nonconvex Pareto sets are very difficult to treat. Second, for a fixed vector $\mathbf{w}$, optimizing $f_{\mathbf{w}}$ usually gives only one point on the Pareto front. And worse, it is difficult to predict the behavior of the resulting optimal point as $\mathbf{w}$ varies. A very slight change in $\mathbf{w}$ might cause a big move of the optimal point, which prevents us from figuring out the whole Pareto information.

Despite these disadvantages, the weighted sum strategy has long been used and is still a primary tool in the engineering field because in practical problems it is often sufficient to find only a set of parameter values that satisfies the required conditions, and experienced engineers have developed a deep instinct for which weight vector will work.

3.2. ϵ-Constraint Method

Another classical way of tackling MOO is to focus on one objective function. The ϵ-constraint method, introduced by Haimes [18], selects a primary objective function $f_l(\mathbf{p})$, the remaining objective functions being converted into constraints by setting an upper bound ϵ_i. The problem is transformed into the form

$$\text{minimize } f_l(\mathbf{p}) \text{ subject to } \mathbf{p} \in \mathcal{P} \ \wedge \bigwedge_{i \neq l} f_i(\mathbf{p}) \leq \epsilon_i \ .$$

This method can be combined with evolutionary algorithms to be described in Section 3.4.

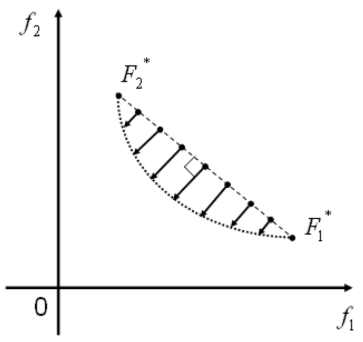

Figure 1: Pareto optimal front for a two-objective problem

3.3. Normal-Boundary Intersection

The Normal-Boundary Intersection (NBI) method has been proposed by Das and Dennis [11] for generating the Pareto optimal front for nonlinear multi-objective optimization problems. This method computes the Pareto front much more efficiently than varying a weight vector to collect Pareto points.

The NBI method first finds the so-called anchor points F_i^*, $i = 1, \ldots, n$, obtained by optimizing only the ith objective function, and then tries to improve a point from the convex hull of F_i^* by searching for a better point along a line perpendicular to the convex hull towards the origin. Figure 1 illustrates how NBI works. This approach can efficiently find a suitable set of optimal points that are roughly equally spaced on the Pareto front.

The NBI method supposes that the Pareto optimal front lies outside the convex hull. When the Pareto front is nonconvex, some of the optimal points might not be obtained.

3.4. Evolutionary Algorithms

Evolutionary algorithms (EAs) are meta-heuristic search methods that have been inspired by natural selection and survival-of-the-fittest criteria in the biological field. When used as an optimization algorithm it starts from a group of individuals, called a population, placed in the parameter space. After all the points are evaluated, a competitive selection process is emulated in which better points survive with a higher probability, followed by a reproduction process to recover the population. Mutations are also taken into consideration in this process, which helps prevent the population falling into a local optimum. The population gradually converges to an optimal point according to the generation. Refer to [15] for a comprehensive guide to the area.

This approach is considered flexible as to objective functions and there have been many successful results in various areas of optimization problems. Recording the nondominated points throughout the generations gives information on the Pareto optimal front.

Ideas behind EAs are natural and larger-scale experiments are efficiently realizable as high-performance computers have been ubiquitous.

3.4.1. Genetic Algorithms

Genetic Algorithms, first proposed by Holland in 1975, are the most common type of EAs simulating genetic reproduction, crossover (recombination), or mutation [16]. A problem is encoded in a series of bit strings that are manipulated by the algorithm; a binary string of data behaves like a gene.

By exploiting the information throughout the generations they search the area with better performance within the parameter space. Adopting a different way of encoding can change the rules of evolution. But it is sometimes difficult to find out the actual, theoretical meaning of the operations on binary data such as reproduction and crossover.

3.4.2. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is another type of EAs, first introduced by Eberhart and Kennedy [14]. PSO is based on the observation of how a swarm of insects or a flock of birds behaves. First a swarm is randomly placed in the parameter space. An algorithm keeps the data of the best position ever obtained by the swarm and the personal best position for each individual.

Each individual simultaneously moves to the next position that are the sum of relative vectors from itself to (1) the group's best position so far, (2) its best position, and (3) a random vector, each multiplied by respective constants. Random vectors help prevent a swarm from falling into a local optimum. As time passes, they are likely to gather at an optimal point. They are sometimes divided into a smaller groups and each gathers different local optimal points. There are lots of variants for PSO; even in the basic algorithm described above it is left to the user to determine the constant values by which three vectors are multiplied.

PSO has been attracting more and more attention these days. There are a lot of empirical evidence to support this approach in the engineering field considering it is easily implemented. Kim, Maruta, and Sugie [20, 23] have applied a variant of PSO to robust PID controller design and to fixed-structure H_∞ controller synthesis.

4. Quantifier Elimination

We change the subject in this section. It would be helpful to overview quantifier elimination (QE) before presenting our symbolic approach to MOO.

Let φ be a first-order formula over a real closed field of the form

$$\varphi = \mathsf{Q}_1 x_1 \cdots \mathsf{Q}_n x_n \; F(x_1, \ldots, x_n, p_1, \ldots, p_k) \ ,$$

where Q_i is either the existential quantifier $\exists$ or the universal quantifier $\forall$ and F is composed of integral polynomial equations, inequations, or inequalities that are appropriately combined by boolean operators $\vee$, $\wedge$, or $\neg$. Variables

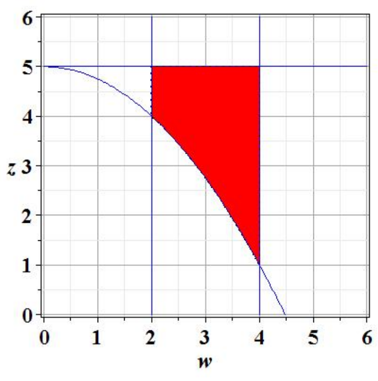

Figure 2: The feasible region of of φ_0

x_i in φ are called quantified variables and p_i free variables. Quantifier elimination is a procedure that takes φ as input and returns an quantifier-free equivalent of φ.

We show an example problem here. Consider the following first-order formula φ_0 with quantified variables x and y and free variables w and z:

$$\begin{aligned}\varphi_0 \;=\; & \exists x \exists y\ (4x - w^2 = 0 \ \wedge\ x - xy - z + 5 = 0 \\ & \wedge\ 1 \leq x \leq 4 \ \wedge\ 1 \leq y \leq 2)\ .\end{aligned}$$

A QE procedure solves φ_0 to return a quantifier-free formula with respect to w and z that is equivalent to φ_0. A possible output formula would be

$$\begin{aligned}& (w - 2 \geq 0 \ \vee\ w + 2 \leq 0)\ \wedge\ w + 4 \geq 0 \ \wedge\ w - 4 \leq 0 \\ & \wedge\ z - 5 \leq 0 \ \wedge\ 4z + w^2 - 20 \geq 0\ .\end{aligned}$$

We can visualize the feasible region in the w-z plane in Figure 2. The red region in Figure 2 is a necessary and sufficient condition for (w, z) to make φ_0 true.

The history of QE dates back to 1930 when Tarski first proved the existence of a decision procedure for sentences in a real closed field. Tarski later published a QE procedure [25], but that is not practical.

In 1975 Collins [8, 10] made a breakthrough in this field. His method is based on cylindrical algebraic decomposition (CAD), partitioning the variable space into invariant subspaces for the polynomials involved in the input formula. The formula is evaluated by the representative points from those invariant subspaces. Hong [9] has implemented a CAD-based QE algorithm on a computer. Several improvements have been proposed mainly in the projection phase of CAD-based QE [19, 22, 7].

But from the computational point of view QE is indeed very hard. Davenport and Heinz [12] have proved that real quantifier elimination is doubly exponential in the worse case with respect to the number of quantified blocks.

Other approaches to tackling QE began to rise around the middle of the 1980s. Weispfenning [26] has proposed a QE algorithm by virtual term substitution, followed by some improvements [21, 27, 28]. This method efficiently works for the formulas with a low degree. González-Vega [17] has focused on a univariate polynomial and examined the condition for the polynomial to have a constant sign by using the Sturm-Habicht sequence as well as combinatorial arguments. These algorithms are efficiently applicable to a restricted class of input formulas and are called special QE. In contrast to special QE, a CAD-based QE algorithm is called general QE because it accepts any first-order formulas.

Improvements in algorithms coupled with enhanced hardware performance enable engineers to apply QE approached to their practical problems. Anai and Hara [2, 3, 4] have investigated that constraints arising in robust control problems can be recast as a special type of formulas that they call the sign definite condition (SDC) and succeeded in solving their problems very efficiently by applying a special QE algorithm for SDC. Dorato et al. [13, 1] have also proposed a robust control design method by QE.

We have been developing on the Maple software a QE toolbox named SyNRAC that solves first-order formulas over the reals [5, 29, 30]. SyNRAC is an abbreviation for a Symbolic-Numeric toolbox for Real Algebraic Constraints. Using SyNRAC as a core engine we have developed a MATLAB toolbox for fixed-structure robust control synthesis [6] and a Maple package for MOO that makes a polynomial model from the input-output data of a simulator, seeks a model reduction, and computes a feasible region by QE. Our MOO package has been applied to the design problem of a hard disk slider.

5. Our Approach

Our approach to multi-objective optimization problems begins with making a polynomial model for each of the objective functions. In most applications an optimization process is realized via a simulator that takes a list of real values for parameters as input and returns a collection of real values representing various physical properties of the product, and the objective functions are defined from these output values. The simulator is often treated as a black box.

We need not only to make a model that neatly fits input-output data but to express it simply so as to make symbolic computation work. A low-degree model would be desirable for QE in a later phase.

5.1. Drawing Feasible Regions

Once the objective functions are expressed as approximated polynomial models in parameters, we formulate a constraint as a first-order formula and compute the feasible region by QE.

To show how an MOO problem is interpreted as a first-order formula, we recall that the parameter space $\mathcal{P}$ and the feasible region $\mathcal{F}$ of a mult-objective optimization problem can be expressed as

$$\begin{aligned}\mathcal{P} &= \{\mathbf{p} \in \mathbb{R}^m | \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\}\ , \\ \mathcal{F} &= \{\mathbf{f}(\mathbf{p}) \in \mathbb{R}^n | \mathbf{p} \in \mathcal{P}\}\ .\end{aligned}$$

From these notations we can naturally reform them into a first-order formula

$$\psi = \exists \mathbf{p}\ \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\ \wedge\ \mathbf{y} = \mathbf{f}(\mathbf{p})\ ,$$

where $\mathbf{y} = (y_1, \ldots, y_n)$ is introduced to represent a point in the feasible region. By eliminating $\mathbf{p}$ from ψ we obtain a quantifier-free formula with respect to $\mathbf{y}$.

A quantifier elimination algorithm symbolically solves ψ to return a quantifier-free equivalent of the input, i.e., a formula only on $y_1, \ldots, y_n$. This means that QE can compute not only its Pareto optimal front but the exact feasible region(s) of ψ, bringing an enormous advantage compared to numerical optimization methods that usually find only one optimal point—or at most a finite set of them—at a time.

But computing feasible regions is one thing; visualizing them is another. When the number of objective functions exceeds three, it is difficult to show the feasible region on screen and to make engineers understand what is going on. Even in the three-objective case, supplementary figures sometimes need to help the user's understanding.

From the practical point of view there are two concerns in our QE approach. One is, as is often the case with symbolic computation, QE has a bad computational complexity in nature. We will make our model as simple as possible to avoid heavy computation. The other is how our result reflects the reality; we cannot say how much difference is caused by an approximated model. We discuss these in the next section.

5.2. Robustness of Production tolerance

We propose another application of our symbolic method to measure the robustness of production tolerance. In MOO problems, two points on the Pareto front are not comparable, i.e., neither is better with regard to the objective functions. We introduce another criterion for judging which point would be better.

Suppose there are two candidate parameter vectors $\mathbf{p}_1$ and $\mathbf{p}_2$ and that neither of their respective lists of objective function values $\mathbf{y}_1 = \mathbf{f}(\mathbf{p}_1)$ and $\mathbf{y}_2 = \mathbf{f}(\mathbf{p}_2)$ dominates the other. A simple way of selecting such a pair of vectors is just choose two different points $\mathbf{y}_1$ and $\mathbf{y}_2$ on or near the Pareto optimal front and let $\mathbf{p}_1$ and $\mathbf{p}_2$ be the respective parameters for them.

Next we make first-order formulas

$$\psi_i = \exists \mathbf{p}_i\ \mathbf{p}_i \in B_\epsilon(\mathbf{p}_i)\ \wedge\ \mathbf{y} = \mathbf{f}(\mathbf{p}_i)$$

for $i = 1, 2$, similar to one in the previous section but $\mathbf{p}_i$ can move only in $B_\epsilon(\mathbf{p}_i)$, the neighborhood of $\mathbf{p}_i$ whose boundary is a hypercube with its center being $\mathbf{p}_i$ and its side a length of 2ϵ.

By solving ψ_i with a QE command we obtain a quantifier-free formula with respect to $\mathbf{y}$ and ϵ whose feasible region of course contains $\mathbf{y}_i$. For a fixed ϵ draw the two feasible regions to compare their shapes. We can estimate that the parameter set with a smaller feasible region is more robust because it means that manufacturing errors, arising when, for example, cutting materials, less affect the characteristics of the product. Similarly it is preferable for $\mathbf{y}_i$ to be positioned near the center of the possible region.

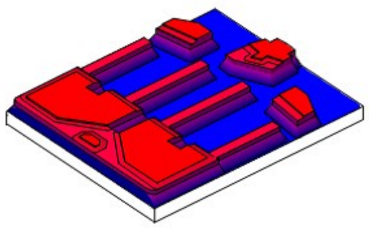
Figure 3: A hard disk slider

6. Application to ABS Design

We show an application of our method to the design problem of air bearing surface (ABS). ABS is part of a hard disk slider. As was briefly explained in Section 1, binary data is written on or read from a disk with an actuator arm. At the end of the arm is attached ABS, on the tip of which a magnetic head is fixed.

As the disk of a HDD rotates the header flies over its surface. ABS is not perfectly flat but is shaped to stabilize aerodynamically the relative position between the disk and the read-write head, its distance being the order of ten nanometers. Figure 3 shows an example of ABS. Notice that it is attached upside-down to an actuator arm. A connected component of the same height, or thickness is sometimes called an island.

We explain a more concrete setting for a hard disk slider design problem. First the designer draws a base shape of ABS. Next the designer introduces a list of parameters $\mathbf{p} = (p_1, \ldots, p_m)$. Each p_i corresponds to the x- or y-coordinate of a vertex chosen from the boundary of an island. According to these parameter values the related islands change in shape.

Simply speaking one searches an optimal ABS shape by iterating simulation for various combinations of parameters and checking its performance. It is the number of parameters that greatly influences the time needed for an optimization. In most cases the designer sets several dozen parameters. To cover a variety of shapes and to have optimization done in one attempt, they tend to set many parameters. Sometimes they pick nearly 50 of them. It is often the case that some of the parameters affect very little the characteristics of the slider. That is why it is very important to remove irrelevant parameters when we make a polynomial model.

The ABS design problem we take here has 23 parameters $\mathbf{p} = (p_1, p_2, \ldots, p_{23})$. Each of them relates to a geometrical shape of the surface though we do not explain their roles in detail. It is possible that only several parameters dominate the objective functions.

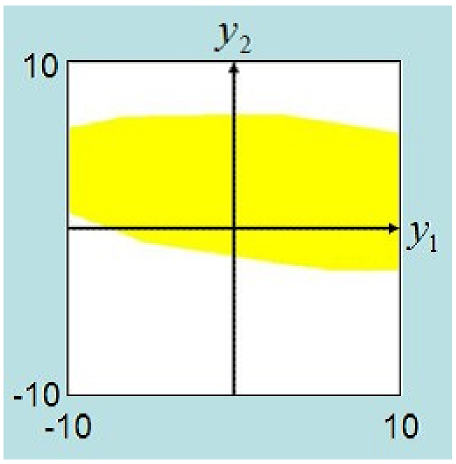

Figure 4: A feasible region of ψ

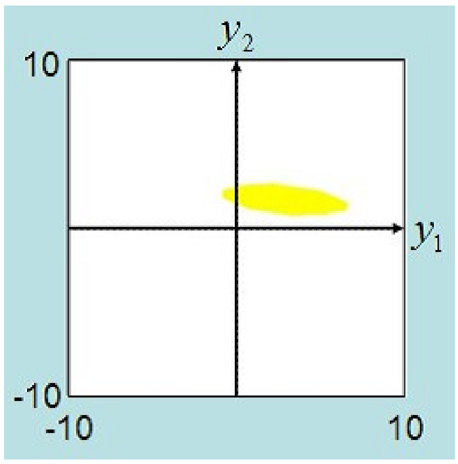

Figure 5: A feasible region of ψ'

As for the objective functions, the performance of a slider is evaluated by nine functions $f_1, f_2, \ldots, f_9$ computed from the output of an ABS simulator. They evaluate some performance related to the flying height, the roll, or the pitch of the slider. Note that each parameter is normalized, running between zero and one inclusive. From now on we focus on a pair of functions f_1 and f_2, which measure the performance of the flying height at a low and a high altitude, respectively, and consider the Pareto front with respect to these two objective functions.

From the samples collected we make an approximated model for f_1 and f_2. We have shed the parameters down to seven in the following manner. First we made a linear regression using every five-parameter set to compute a determination coefficient and found that the parameter set $P_1 = \{p_2, p_4, p_{13}, p_{16}, p_{17}\}$ had a highest determination coefficient of 0.549 for f_1 and $P_2 = \{p_2, p_{14}, p_{15}, p_{16}, p_{17}\}$ for f_2 with that of 0.859. And we made a linear regression for f_1 and f_2 using $P_1 \cup P_2 = \{p_2, p_4, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}\}$. The resulting approximated models were

$$\begin{aligned} f_1 = &-3.92343588542323074 + 5.63538665709504727p_2 \\ &+2.43177957302682746p_4 + 7.39011285328667266p_{13} \\ &-3.33577994026196478p_{14} + 0.440737805380383762p_{15} \\ &-7.01941019322546733p_{16} + 10.9647299755436568p_{17} \end{aligned}$$

and

$$\begin{aligned} f_2 = &\ 2.89852466590775526 - 2.16942175484827126p_2 \\ &-0.0252395610401109291p_4 + 0.159055858255768234p_{13} \\ &+1.73525308573789183p_{14} + 2.02926186610533454p_{15} \\ &-1.52264764786695528p_{16} - 1.69478663255212502p_{17} \ . \end{aligned}$$

Note that coefficients in the above formulas are all interpreted as rational numbers when we construct a formula.

Next we make a first-order formula for QE. For economy of space let $I = \{2, 4, 13, 14, 15, 16, 17\}$ denote the set of indices for the selected parameters. The formula we need to solve can be expressed as

$$\psi(y_1, y_2) = \exists \mathbf{p}\ \mathbf{p} \in \mathcal{P}\ \wedge\ y_1 = f_1(\mathbf{p})\ \wedge\ y_2 = f_2(\mathbf{p}),$$

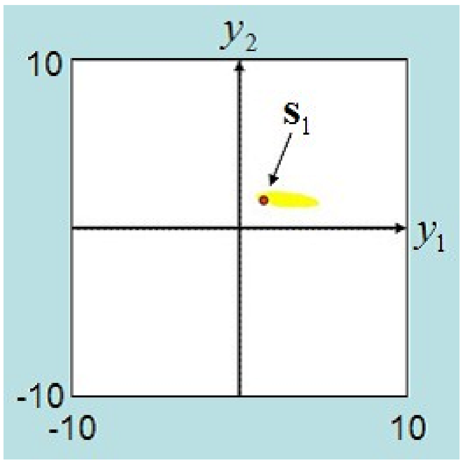

Figure 6: A feasible region near $\mathbf{p}_1$

where $\mathbf{p} = (p_i)_{i \in I}$, and $\mathcal{P} = \{(p_i)_{i \in I} |\ 0 \le p_i \le 1 \text{ for } \forall i \in I\}$. By solving this formula we obtain the feasible region in Figure 4, which shows there is a trade-off between the two cost functions f_1 and f_2. But the feasible region runs over the first quadrant, larger than expected. This is probably due to a simplified model and a large parameter space.

Now we try to impose some restriction on parameters; instead of letting each parameter p_i run freely in $[0, 1]$, make p_i move only around a_i, where $a_i - 1/10 \le p_i \le a_i + 1/10$ for some $A = (a_i)_{i \in I}$ near the Pareto front, i.e., consider the following:

$$\psi'(y_1, y_2) = \exists \mathbf{p}\ \mathbf{p} \in B_{1/10}(A)\ \wedge\ y_1 = f_1(\mathbf{p})\ \wedge\ y_2 = f_2(\mathbf{p})\ .$$

We call A an adjusting point. Figure 5 shows the resulting possible region, which almost lies on the first quadrant.

Lastly, to use the criterion for comparing the robustness of production tolerance take two points near the Pareto front. We chose two sample points S_1 and S_2 shown in Figures 6 and 7. Let A_i be the corresponding parameters for S_i and consider the following formulas

$$\varphi(y_1, y_2, \epsilon) = \exists \mathbf{p}\ \mathbf{p} \in B_\epsilon(A_i)\ \wedge\ y_1 = f_1(\mathbf{p})\ \wedge\ y_2 = f_2(\mathbf{p}).$$

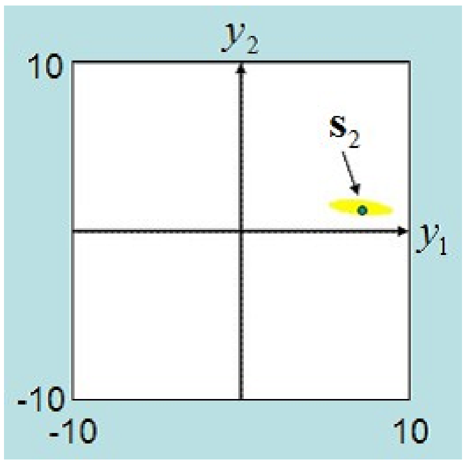

Figure 7: A feasible region near $\mathbf{p}_2$

After solving the formula we substitute 1/100 for ϵ to obtain the respective shaped regions in Figures 6 and 7.

By comparing these two regions we can estimate that the parameter set A_2 corresponding to S_2 would be more robust because S_2 is placed at the center of the region while S_1 is placed on the extreme left of the region.

Note that all the computations were done on a 3.80 GHz Pentium 4 desktop with 2 GB of RAM. It takes a few seconds to solve the QE problems in this section, partly due to the model reduction. Roughly speaking, it takes a whole night to find an optimal point by the weighted sum strategy with a dozen CPUs or so. This means that trying a few set of parameters would take a day. Compared to that time, the solving time of our QE computation is negligible. Our approach can reuse the data taken an optimization and provide information on Pareto optimal front. This feedback can also be used to decide a weight vector in the weighted sum strategy.

7. Conclusions

We have proposed a new approach to tackling multi-objective optimization problems with symbolic computation and applied the method to the design problem of a hard disk slider. Starting from the input-output data of a simulator, we have made approximated polynomial models and constructed a first-order formula. By solving the formula we have been able to visualize the possible region. Our QE approach can compute the feasible region, which has an advantage over classical numerical optimization methods that return only one optimal point per optimization.

Experiments have indicated that our polynomial model was not good enough to grab the global relation between cost functions. But by introducing an adjusting point and by restricting the parameter ranges it is possible to obtain some information on the feasible region locally from which we deduce some properties of the Pareto front. This approach can be used to compare the robustness against production tolerance for two points that are not comparable with regard to their objective function values.

Acknowledgments

This work has been partially supported by CREST, Japan Science and Technology Agency.

References

[1] Abdallah, C., Dorato, P., Yang, W., Liska, R., and Steinberg, S.: Application of quantifier elimination theory to control system design, In: Proceedings of 4th IEEE Mediterranean Symposium on Control and Automation. Maleme, Crete. (1996) 340–345.

[2] Anai, H. and Hara, S.: Fixed-structure robust controller synthesis based on sign definite condition by a special quantifier elimination, In: Proceedings of American Control Conference 2000 (2000) 1312–1316.

[3] Anai, H. and Hara, S.: Linear programming approach to robust controller design by a quantifier elimination, In: Proceedings of SICE Annual Conference 2002 (Osaka, Japan) (2002) 863–869.

[4] Anai, H. and Hara, S.: A parameter space approach for fixed-order robust controller synthesis by symbolic computation, In: Proceedings of IFAC World Congress on Automatic Control b'02 (2002).

[5] Anai, H. and Yanami, H.: SyNRAC: A maple-package for solving real algebraic constraints, in: Proceedings of International Workshop on Computer Algebra Systems and their Applications (CASA) 2003 (Saint Petersburg, Russian Federation), P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2657, Springer (2003) 828–837.

[6] Anai, H., Yanami, H., Sakabe, K., and Hara, S.: Fixed-structure robust controller synthesis based on symbolic-numeric computation: design algorithms with a CACSD toolbox (invited paper), In: Proceedings of CCA/ISIC/CACSD 2004, Taipei, Taiwan (2004) 1540–1545.

[7] Brown, C.W.: Improved projection for cylindrical algebraic decomposition, Journal of Symbolic Computation **32** (2001) 447–465.

[8] Collins, G.E.: Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition, In Brakhage, H., ed.: Automata Theory and Formal Languages. 2nd GI Conference, Volume 33 of Lecture Notes in Computer Science., Gesellschaft für Informatik, Springer-Verlag, Berlin, Heidelberg, New York (1975) 134–183.

[9] Collins, G.E. and Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination, Journal of Symbolic Computation **12** (1991) 299–328.

[10] Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition, In Caviness, B., Johnson, J., eds.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and Monographs in Symbolic Computation, Springer, Wien, New York (1998) 85–121.

[11] Das, I. and Dennis, J. E.: Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems, SIAM Journal of Optimization **8** (1998) 631–657.

[12] Davenport, J. H. and Heinz, J.: Real Quantifier Elimination Is Doubly Exponential, *J. Symbolic Computation* **5** (1-2) (1988) 29–35.

[13] Dorato, P., Yang, W., and Abdallah, C.: Robust multi-objective feedback design by quantifier elimination, J. Symb. Comp. **24** (1997) 153–159.

[14] Eberhart, R. C. and Kennedy, J.: A new optimizer using particle swarm theory, in: *Proc. 6th Int. Symp. Micromachine Human Sci.* **1** (1995) 39–43.

[15] Eiben, A. E. and Smith, J. E.: Introduction to Evolutionary Computing, Springer, 2nd printing, 2008.

[16] Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Professional, 1989.

[17] González-Vega, L.: A combinatorial algorithm solving some quantifier elimination problems, In Caviness, B., Johnson, J., eds.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and monographs in symbolic computation, Springer-Verlag (1998) 365–375.

[18] Haimes, Y. Y.: Integrated System Identification and Optimization, Control and Dynamic Systems: Advances in Theory and Applications, **10** (1973) 435–518.

[19] Hong, H.: An improvement of the projection operator in cylindrical algebraic decomposition, In Caviness, B., Johnson, J., eds.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and Monographs in Symbolic Computation, Springer, Wien, New York (1998) 166–173.

[20] Kim, T.-H., Maruta, I., and Sugie, T.: Robust PID controller tuning based on the constrained particle swarm optimization, Automatica **44** (2008) 1104–1110.

[21] Loos, R. and Weispfenning, V.: Applying linear quantifier elimination, The Computer Journal **36** (1993) 450–462, Special issue on computational quantifier elimination.

[22] McCallum, S.: An improved projection operation for cylindrical algebraic decomposition, In Caviness, B., Johnson, J., eds.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and Monographs in Symbolic Computation, Springer, Wien, New York (1998) 242–268.

[23] Maruta, I., Kim, T.-H., and Sugie, T.: Fixed-structure H_∞ controller synthesis: A meta-heuristic approach using simple constrained particle swarm optimization, Automatica **45** (2009) 553–559.

[24] Stehr, G., Graeb, H., and Antreich, K. Performance trade-off analysis of analog circuits by normal-boundary intersection, in: *DAC 2003, June 2-6, 2003, Anaheim, California, USA*, (2003) 958–963.

[25] Tarski, A.: A decision method for elementary algebra and geometry, in Quantifier Elimination and Cylindrical Algebraic Decomposition, In Caviness, B., Johnson, J., eds.: Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and Monographs in Symbolic Computation. Springer, Wien, New York (1998) 24–84.

[26] Weispfenning, V.: The complexity of linear problems in fields, Journal of Symbolic Computation **5** (1988) 3–27.

[27] Weispfenning, V.: Applying quantifier elimination to problems in simulation and optimization, Technical Report MIP-9607, FMI, Universität Passau, D-94030 Passau, Germany, 1996.

[28] Weispfenning, V.: Quantifier elimination for real algebra—the quadratic case and beyond, Applicable Algebra in Engineering Communication and Computing **8** (1997) 85–101.

[29] Yanami, H. and Anai, H.: Development of SyNRAC—formula description and new functions, in: Proceedings of International Workshop on Computer Algebra Systems and their Applications (CASA) 2004 : ICCS 2004, LNCS 3039, Springer (2004) 286–294.

[30] Yanami, H. and Anai, H.: The Maple package SyNRAC and its application to robust control design, Future Generation Computer Systems **23** (2007) 721–726.

[31] Zadeh, L.: Optimality and non-scalar-valued performance criteria, Automatic Control, IEEE Transactions **8** (1963) 59–60.

Hitoshi Yanami
FUJITSU Laboratories Ltd., Kawasaki 211-8588, Japan
E-mail: yanami(at)labs.fujitsu.com