

A Parallel Approach for Computing the Expected Value of Gathering Information

Edgar Galvan, Chuck Hsiao, Sean Vermillion, and Richard Malak
Texas A&M University

ABSTRACT

It is important for engineering firms to be able to develop forecasts of recommended courses of action based on available information. In particular, engineering firms must be able to assess the benefit of performing information-gathering actions. For example, an automobile manufacturer may use a computer simulation of a hydraulic motor and pump in the design of a new vehicle. The model may contain random variables that can be more accurately determined through expensive information-gathering actions, e.g., physical experiments, surveys, etc. To decide whether to perform these information-gathering actions, the automobile manufacturer must be able to quantify the expected value to the firm of conducting them. However, the cost of computing the expected value of information (through optimization, Monte Carlo sampling, etc.) grows exponentially with the amount of information that is to be gathered and can often exceed the cost of actually gathering the information. Thus, if information decisions are to be addressed algorithmically, there exists a need for novel algorithmic approaches to reduce the computational expense associated with computing the expected value of gathering information. The contribution of this paper is a novel algorithmic approach for approximating the expected value of perfect information (EVPI) for engineering design problems. In this research, we propose to recast the EVPI as a "parametric" problem. The value of recasting the problem is an exponential reduction in the computational complexity. The proposed approach is validated against a Monte Carlo sampling based approach, the traditional approach for solving EVPIs, on an engineering problem. The engineering problem is to compute the expected value of performing physical experiments to gather information about random variables in a computational model the efficiency of an engine and transmission. The results are compared in terms of computational expense and solution accuracy. The results indicate that by recasting the EVPI as a parametric problem the computational expense is reduced drastically while maintaining solution accuracy.

CITATION: Galvan, E., Hsiao, C., Vermillion, S., and Malak, R., "A Parallel Approach for Computing the Expected Value of Gathering Information," *SAE Int. J. Mater. Manf.* 8(2):2015, doi:10.4271/2015-01-0436.

INTRODUCTION

The contribution of this paper is a novel algorithmic approach for computing the expected value of gathering information during engineering design. Often in engineering practice, a designer must choose to either design an artifact given the current information or gather information about the random variables. This is especially true during the conceptual design stages. Consider for example the design of a vehicle's multi-ratio transmission. The optimal transmission design depends on several random variables such as engine characteristics, use case, etc. The designer must choose to either design the transmission given the current state of information or perform an information-gathering action such as a physical experiment or survey, which would reduce uncertainty at a cost. The designer must determine whether the benefit of gathering information is greater than the cost of gathering that information. Standard decision-making approaches tend to neglect the information-gathering aspects of the decision making process, instead focusing exclusively on decision making with current information. Good engineering practice must take a broader view of decision making and consider the value of gathering information.

A framework for considering the resources involved in gathering information with how it improves decision making is Expected Value of Perfect Information (EVPI) [1, 2]. The EVPI is the price one is willing to pay for perfect information. One examines how much the expected reward would improve if the uncertain information were perfectly accurate. The expected increased reward is compared to the cost of gathering that information to determine whether gathering the information is worthwhile. Expected Value of Partial Perfect Information (EVPPI) is an extension that considers multiple random variables. Under EVPPI, the engineer chooses the set of parameters for which to gather perfect information that maximizes the engineer's expected reward.

However, many sources of information are not perfect, but can have uncertainty themselves. This may be due to the information-gathering method itself or with its predictive power on the parameter(s) of interest. Additionally, the cost of information-gathering typically increases dramatically with experimental accuracy, thus in practice uncertainty is typically not reduced to zero. Instead, the amount of uncertainty reduced is traded off against the cost of reducing the

uncertainty. EVPI does not account for this. Instead, as a practical matter, the value for having perfect information sets an upper bound on the value of gathering information.

In contrast to EVPI, the Expected Value of Sample Information (EVSI) framework does account for a reduction in the uncertainty based on an imperfect source of information. Each information-bearing test or sample gives information about a parameter of interest, but does not reveal the parameter directly. The decision maker takes this uncertainty into account in calculating the expected value of information. A deeply related problem is deciding not just whether or not to collect a sample, but how many samples to collect as a batch. This approach is used in applications such as medical clinical trials, whether the number of subjects needed to test an experimental drug is decided in advance of the actual trials.

For this paper, information is considered to be perfect, and thus falls under the EVPI framework rather than the EVSI framework. However, it should be noted that this is simply to simplify the exposition; the concepts described in this paper can be readily extended to account for imperfect information and EVPII.

Although the EVPI framework has been extensively studied, one drawback in its implementation is the difficulty of computing the expected reward for the possible information-gathering alternatives. Determining the expected reward for a given choice of information-gathering actions involves a nested optimization within the Monte Carlo simulations, which is computationally expensive. The contribution of this paper is a novel algorithmic approach for computing expected reward that avoids the issue of nested optimization.

EXPECTED VALUE OF PERFECT INFORMATION (EVPI) IN ENGINEERING DESIGN

The primary decision in an EVPI problem is choosing whether to gather information. For clarity, choosing what information to gather will be called the *information decision* while choosing which alternative based on that information will be called the *design decision*. In these types of problems, the information decision is made before the design decision, since the design decision depends on the outcome of the information decision.

We begin with some notation. Let x be a vector denoting the design variables (those which the designer can control), which must belong to the nonempty feasible region (set) X_f . For example, design variables might be the design details of a hydraulic pump. The feasible region X_f of the design space is defined by equality and inequality constraints, which typically correspond to geometric constraints and physical laws. Let y be a vector of parameters in the model with probability distribution $p(y)$. The random variables correspond to parameters that are outside of the control of the designer. Examples of random variables are modeling parameters or use scenarios for a hydraulic pump.

The value (or profit, benefit) model $f: X \times Y \rightarrow \mathbb{R}$ maps the Cartesian product of the *decision space* and *parameter space* to the payoff (profitability) of the design. The function relates design details and random variables to value of the design. The expected value of perfect information (EVPI) is the value of eliminating the uncertainty on y . If we do not learn the true value of y , we make the design decision (choice of x) that maximizes the expectation over the random variable; this is the *design decision*. Suppose a random variable Y can take values between y^{ub} and y^{lb} according to the probability density function $p(y)$. The expected value of solving the design decision with current information is

$$\max_{x \in X_f} E[f(x, y)] \approx \max_{x \in X_f} \left(\frac{1}{N} \sum_{i=1}^N f(x, y_i) \right), \quad (1)$$

where $y_i \sim P(Y)$ and N is the number of samples drawn to approximate the expectation. This problem can be solved using traditional methods for optimization under uncertainty, e.g., Monte Carlo Simulation, common random numbers, etc. The solution to [Eq. \(1\)](#) is the expected value of designing the artifact without gathering information. The argument that maximizes [Eq. \(1\)](#) is an instance of x , i.e., a design concept.

Alternatively, the designer can choose to gather information about random variables y through experimentation, incurring a cost, denoted C . The decision problem is now whether the designer should gather information about the random variables. Consider a situation where the designer has the option to conduct an experiment that will reveal the true value of the random variable y . If the true value of y is known with certainty, the designer can optimize the design (make the design decision) with respect to that specific value of y , i.e., $\max_{x \in X_f} f(x, y)$. The optimal solution to the design problem (choice of x) in this case is dependent on y , which is still unknown before conducting the experiment. The expected profit if we conduct a perfect experiment (i.e., EVIP) is

$$E \left[\max_{x \in X_f} f(x, y) \right] - C \approx \frac{1}{N} \sum_{i=1}^N \left(\max_{x \in X_f} f(x, y_i) \right) - C. \quad (2)$$

Unlike in the decision problem, the outcome of [Eq. \(1\)](#) is not a design concept (choice of x). Instead, the aim is to approximate the value of gathering perfect information. The optimal x will depend on the outcome of the information gathering action, which is unknown at this stage.

A numerical approximation of [Eq. \(2\)](#) requires multiple optimizations, one for each of the N instances of y_i [3]. The information decision is to choose the maximum expected value between designing with or without gathering information. [Figure 1](#) is an illustration of the design problem; the value of the alternatives is formulated on the right.

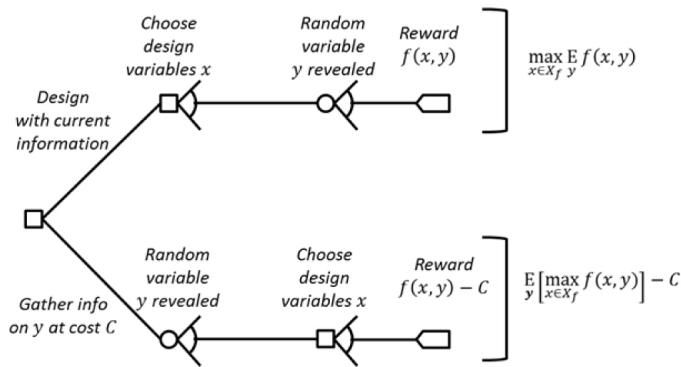


Figure 1. Decision Tree for EVPI problem.

Let K be the number of steps required to solve each optimization problem. The number of evaluations of $f(\cdot)$ required to solve the decision tree in Figure 1 would be $2KN$. Consider a scenario where the optimization algorithm requires $K = 500$ steps, and $N = 50$ samples are used to approximate the expected value. If it takes only 10 minutes to evaluate the function $f(\cdot)$, the total computation time required to evaluate expected value of all alternatives would be **nearly a year** of computation time. Even with inexpensive engineering models and parallel computing, the cost of analyzing the decision problem may even exceed the cost of actually gathering the information (physical experiments, surveys, etc.). Thus, if information decisions are to be addressed algorithmically, there exists a need for a novel algorithmic approach to reduce the computational expense.

COMPUTING EVPI USING PARAMETRIC OPTIMIZATION

The focus of this paper is on reducing the computational expense of the EVPI problem described in Eq. (2). If the random variable y is a continuous variable with probability density function $p(y)$, the EVPI problem is defined by the integral

$$E \left[\max_{x \in X_f} f(x, y) \right] = \int_{y^{lb}}^{y^{ub}} \left(\max_{x \in X_f} f(x, y) \right) p(y) dy \quad (3)$$

where y^{ub} and y^{lb} are the upper and lower bounds on y , respectively. Note that we have ignored cost of gathering information in Eq. (3). The probability density function $p(y)$ is derived from the beliefs of the decision maker. Thus, the difficulty in solving the EVPI problem in Eq. (3) is the computational expense in evaluating $\max_{x \in X_f} f(x, y)$ at several values of y . This notion is the basis for using parametric optimization to solve EVPI problems. Under the concept of parametric optimization, we can solve several related optimization problems simultaneously. We will decompose Eq. (3) into two problems. First, we will approximate the solution of $\max_{x \in X_f} f(x, y)$ at several values of y . Secondly, we will sample those solutions according to the distribution $P(Y)$ to recover the solution to Eq. (3).

Parametric Optimization for the Computing Expected Value

In this research, we propose to recast Eq. (3) as a “parametric” search problem. In order to formulate the parametric search problem mathematically, it is helpful to consider first the non-parametric case, i.e., traditional single objective optimization. We will use maximization throughout this article without loss of generality since minimizing f is equivalent to maximizing $-f$. Using the notation from the previous subsection, we formulate the traditional optimization problem as

$$\max_{x \in X_f} f(x) \quad (4)$$

The set X_f is typically defined by geometric constraints, physical laws, and practical considerations. The aim is to find a solution that maximizes the objective function $f(x)$, and $x \in X_f$. Intuitively, the parametric search problem can be thought of as an application of traditional optimization at every combination of some parameter value(s). We extend Eq. (4) to the general parametric case as

$$\begin{aligned} &\max_{x \in X_f} f(x), \\ &\text{subject to } h(x) = y, \\ &\text{where } y \in Y_f \end{aligned} \quad (5)$$

The principal difference between a traditional optimization problem and Eq. (5) are the additional constraints $h(x) = y$ for **any** $y \in Y_f$ where Y_f is the space of feasible parameter values defined as $Y_f = \{y | y = h(x), x \in X_f\}$. In a traditional optimization, the objective must be improved subject to $x \in X_f$. In Eq. (5), there is the additional condition that the constraint in **any one element** of the set $\{h(x) = y | y \in Y_f\}$ must be satisfied. In other words, the solutions to Eq. (5) are the minimum of $f(x)$ for every possible value of y . Thus, the solution rather than being a single point is a (potentially infinite) set. A parallel can be drawn with Eq. (3) where, to determine the expectation, we need to know the solution to

$$\max_{x \in X_f} f(x, y) \quad (6)$$

for every foreseeable value of y . Thus, the general EVPI problem in Eq. (3) can be recast as parametric optimization problem as follows

$$\begin{aligned} &\max_{x \in X_f} f(x, a), \\ &\text{subject to } a = y, \\ &\text{where } y \in Y_f \end{aligned} \quad (7)$$

The solution to Eq. (7) is a surface rather than a single optimum. Figure 2 is an illustration of a solution in the 2-dimensional case. The feasible region, X_f , is the shaded region in the graph. The bold line is the maximum of the objective function for a given y . The surface can then be inexpensively sampled according to $P(Y)$ to approximate the solution to Eq. (3).

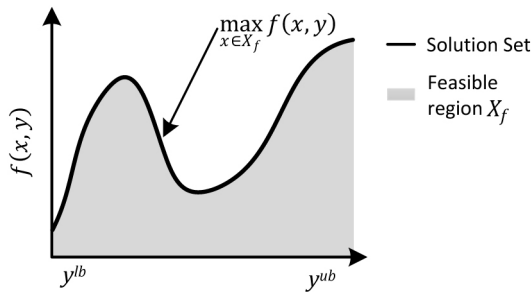


Figure 2. Notional illustration of a solution set to a parametric optimization problem. The shaded region indicates the feasible region X_f . The solid line indicates the maximum value of $f(x, y)$ as a function of y .

Recasting the problem in such a way eliminates the need to perform optimization within a Monte Carlo simulation. Instead of solving N optimization problems (where N is the number of Monte Carlo samples), we need only solve a single parametric optimization problem. In prior research, the authors have developed an efficient algorithm, named Parameterized Predictive Pareto Genetic Algorithm (P3GA) to solve parametric search problems efficiently.

P3GA: AN ALGORITHM FOR PARAMETRIC OPTIMIZATION

Predictive Parameterized Pareto Genetic Algorithm (P3GA)

As mentioned in the previous section, a parametric optimization problem (POP) can be thought of as an application of a traditional single optimization at every feasible combination of parameter values. Therefore, *in theory*, it is possible to solve a POP by discretizing the parameter space then iteratively solving the single objective problems (SOPs) that correspond to every combination of parameter values. The resulting SOPs can be solved using traditional optimization techniques. Unfortunately, this approach can be prohibitively expensive, even for relatively simple problems. Suppose a POP has $p = 3$ parameter variables. We can discretize the parameter space at $u = 10$ different values along each parameter dimension to create several SOPs. If each SOP requires $k = 1,000$ function evaluations to converge, the total number of required function evaluations to solve a PMOP is $ku^p = 10^6$. If the function takes just one minute to evaluate, the total computational runtime would be nearly 2 years of computation time. Sampling the parameter space (i.e., the space of uncertain variables) according to the corresponding distribution would obviously not result in any computational savings compared to traditional methods. A more efficient approach is needed.

Since genetic algorithms work with a population rather than a single point, a number of solutions to a POP can be found in a single run, allowing us to avoid the multiple optimization runs described above and reduce the overall computational expense. Furthermore, genetic algorithms leverage the diversity in the entire population to guide each individual population member. This makes genetic algorithms well suited for the problem at hand. In the proposed algorithm, we adapt the general multiobjective genetic algorithm (MOGA) strategy

to be compatible with POPs. Without loss of generality, suppose designer preferences are monotonically decreasing in each objective. The multiobjective search problem is formulated as

$$\max_{x \in X_f} f_i(x), \quad i = 1, 2, \dots, m \quad (8)$$

The set of solutions to Eq. (8) is termed the Pareto frontier in the literature. Several researchers have successfully approximated the Pareto frontier using MOGAs [4, 5, 6, 7, 8, 9, 10, 11, 12]. A MOGA typically begins its search with a random set of solutions, called the population. Each member of the population is evaluated and assigned a fitness value relative to two meta-objectives:

1. Find a set of solutions close to the Pareto frontier.
2. Maintain diversity in the solution set.

Importantly, most MOGAs rely on the concept of *dominance* to assign higher fitness values to members closer to the Pareto frontier. A Pareto dominated alternative is one for which there exists another alternative that is at least as preferred in all objectives and strictly more preferred in at least one objective. Figure 3a is an illustration of Pareto dominance for three alternatives, denoted a , b and c . The space dominated by each member is shaded and outlined by the dashed lines. In Figure 3a, member b is dominated by a while members c and a are mutually nondominating.

In MOGAs, two members are compared on the basis of whether one dominates the other or not. The nondominated member is closer to the Pareto frontier and is assigned a higher fitness value. The second goal is typically achieved through the use of diversity preservation operator. Then a termination condition is checked. If the termination criterion is not satisfied three main operators modify the population: selection, crossover, and mutation. Each new population is termed a generation.

A genetic algorithm intended for POPs should have similar goals. The principal difference being that rather than finding solutions close to the **Pareto frontier**, the algorithm must find solutions close to the **parametric optimal surface**, illustrated in Figure 2. We adapt the MOGA scheme to suit our needs by replacing the concept of Pareto dominance with *parameterized Pareto dominance* (PPD) in the general MOGA strategy. Under the PPD rule, a dominated alternative is one for which there exists an alternative with a more preferred objective value and equal parameter value. Thus, under the PPD rule a non-dominated alternative will be necessarily closer to the solution of a POP.

The complicating factor in the practical application of the PPD is that **alternatives must have equal parameter values for dominance to occur**. Because genetic algorithms are variants of randomized search algorithms, this is unlikely to occur. In other words, a straightforward application of PPD to the population members is unlikely to dominate *any* members. Figure 3b is an illustration of this difficulty. Figure 3b depicts PPD in the case where the designer has monotonically increasing preference for the objective, denoted f_1 , and y is a