Tuning Missile Guidance and Control Algorithms Using Simultaneous Perturbation Stochastic Approximation

Brian E. Reardon, Justin M. Lloyd, and Ron Y. Perel

Guidance and control (G&C) algorithms play an important role in maximizing the lethality of a guided missile system. While G&C algorithms are diverse in type and complexity, they all have adjustable parameters that affect their operation and, consequently, overall missile performance. The selection, or "tuning," process, whereby the optimum values for the adjustable parameters are obtained, is a critical challenge in G&C algorithm design. Analytical techniques often are unavailable, and manual analyze-and-iterate methods are time-consuming and suboptimal. In this article, we discuss an automated, simulation-based approach to G&C algorithm optimization that uses the simultaneous perturbation stochastic approximation (SPSA) algorithm. The practical challenges of G&C algorithm tuning, as well as effective solutions to these challenges, are presented in detail, and an example is provided.

INTRODUCTION

The overall performance of a guided missile system depends on the performance of its guidance and control (G&C) algorithms. These algorithms perform several important functions. Guidance algorithms use sensor data to guide the missile to a successful intercept. Flight-control algorithms provide stable yet agile flight within the capabilities of the airframe while managing the control energy expended to prevent hardware failure and maximize kinematic range. For a guided missile system equipped with a warhead, G&C algorithms also contribute to lethality by encouraging favorable missile/target geometries in the endgame.

In this context, the role of G&C algorithms is to orchestrate the various missile subsystems to ensure that all applicable requirements are met and to consistently maximize missile lethality. For example, a guidance law takes target sensor data and integrates them with knowledge of the missile state to generate acceleration commands to put the missile on a collision course with the target. The design of such a guidance law clearly plays a significant role in determining the final separation of the missile and target and therefore is a major contributor to the lethality of the overall system.

As threats evolve and increasingly stringent performance requirements are levied on guided missile designs, corresponding advances in missile G&C synthesis techniques and algorithms are required. Many of these advanced algorithms are multivariable techniques and employ high-order models. Furthermore, additional sensors may be required to support algorithm performance over a wide range of operating conditions. These factors increase the complexity and diversity of the algorithms and present additional design challenges. In the Integrated Guidance and Control (IGC) algorithm, for example, the guidance filter, guidance law, and autopilot are combined into a single, integrated G&C algorithm.[1] Plant order and overall IGC algorithm complexity lead to a large number of interdependent design parameters whose connections to the hardware subsystems are not intuitively obvious. This interdependency makes analytical designs difficult, if not impossible, and ad hoc manual design techniques cumbersome at best.

The Design Problem and Its Challenges

Algorithms in general, and G&C algorithms in particular, contain adjustable design parameters that control their operation and ultimate performance. The time constant of a two-state guidance Kalman filter, the navigation ratio of a proportional navigation guidance law, and the gains in a three-loop autopilot are examples of adjustable parameters that are typical first-order performance drivers. Almost every aspect of a G&C algorithm has some adjustable parameter associated with it that affects the behavior of the algorithm and, therefore, affects the guided missile system. Once the structure of a G&C algorithm is established, the design of the algorithm then consists entirely of selecting appropriate values for the adjustable parameters.

Choosing optimum values for the algorithm parameters is a challenging design problem. Most G&C algorithms contain a large number of interdependent parameters, often with nonlinear and nonintuitive influences on system performance. Analytical techniques, when available, usually apply within a restricted domain. Often, the designer has a limited understanding of the relative impact of each parameter in a multivariable setting, having a better understanding of single-parameter effects. The designer also may have a limited understanding of parameter interactions with the diversity of flight scenarios that must be considered (e.g., environmental conditions, threats, intercept ranges). In addition, there can be constraints on parameters, both real and imposed, that complicate the designer's job.

G&C algorithm parameters must be designed to operate over a wide variety of flight conditions and scenarios. Factors such as the suite of targets, flight conditions, and phase of missile flight (e.g., boost, midcourse, terminal) must be considered in the design process. There may be multiple and sometimes competing performance objectives, such as final miss distance, airframe stability, conservation of control energy, and warhead effectiveness. When these types of performance objectives are addressed directly in the design process, a missile simulation of sufficient fidelity to accurately calculate them is required. Sometimes these performance measures are difficult and costly to compute via simulations—which often are highly nonlinear with complex, noisy inputs and model characteristics—thus complicating the design task.

Design Techniques

The techniques used to optimize G&C algorithm parameters vary widely. Analytical techniques are available for some algorithms. For example, the gains in conventional three-loop autopilots can be calculated analytically using linear techniques that satisfy time-constant, airframe stability, and robustness criteria.[2] These gains often are valid over a limited set of flight conditions, and independent gain sets must be generated over a wide range of conditions and scheduled for use in flight. Conventional guidance filter and law design techniques, on the other hand, are usually ad hoc because few analytical design methods are available. The trial-and-error design process involves manual adjustments of the design parameters based on qualitative and quantitative performance measures, sometimes obtained from guided missile simulation outputs. Occasionally, some rigor is added to the manual process by evaluating the design parameters over a range of values. For example, one algorithm gain is parametrically stepped through a range of values while holding all others constant, and the value that yields the best system performance is chosen. This process is repeated for each design parameter. Grid-search techniques such as this are tedious and time-consuming, and the resulting designs are arguably suboptimal.

These challenges demand a more rigorous and structured approach to G&C algorithm gain optimization. Problems of complexity and scale, the diversity of algorithms, and the fact that simulations are increasingly being used in the design process motivate the use of a computer to iteratively seek the optimal values. Such simulation-based optimization techniques use noisy simulation outputs to drive the relevant design parameters to their optimal values. The optimization process is automatic in that the parameters are adjusted by a numerical minimization algorithm rather than by a human (Fig. 1).

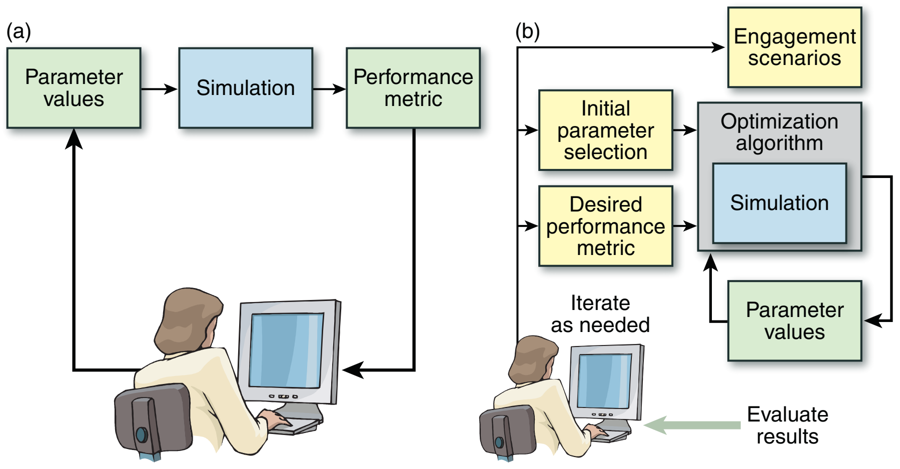


Figure 1. Manual tuning techniques (a) require a human to adjust the algorithm parameters, assess the impact, and readjust the parameters until the desired performance is achieved. Automated-tuning techniques (b), on the other hand, use a numerical optimization algorithm to adjust the parameters to minimize a user-defined performance metric.

Automated approaches have been applied to tuning G&C algorithms. Reference 2 contains a description of applying an automated simulation-based technique to the optimization of an IGC algorithm. Reference 3 discusses the results of automatically tuning an optimal guidance system and dynamic inversion autopilot, as well as an IGC algorithm. Automated tunings of Kalman filters using genetic algorithms, the downhill simplex method, and simulated annealing are discussed in Refs. 4, 5, and 6, respectively, with applications in the G&C domain.

An automated simulation-based approach offers many benefits compared to manual approaches. Automated techniques can be much more efficient than manual tunings for large-scale problems because they optimize many tuning parameters, whereas manual, human-based approaches typically can handle, at most, a few at a time. Similarly, automated techniques that adjust all parameters can exploit parameter interdependencies to produce optimal solutions. Furthermore, multiple G&C algorithms (e.g., the guidance filter, law, and autopilot gains) can be optimized concurrently using automated techniques, whereas they usually are optimized separately with conventional approaches.

Automated approaches to tuning G&C algorithms introduce new challenges not associated with manual techniques. Optimization algorithms include their own adjustable parameters, the values of which greatly impact the efficacy of the tuning process. The quality of the optimized gains produced by an automated simulation-based technique depends heavily on the fidelity and accuracy of the simulation used. Choosing an appropriate cost function that captures and balances all desired performance objectives also is critical to the success of an automated technique.

Selection of the optimization algorithm is a challenge as well. Some optimization algorithms are computationally expensive, while others are less suited to handling the noisy cost measurements that are the typical outputs of guided missile simulations. Others struggle with finding a global minimum to a cost function and are prone to generating only locally optimum solutions. Some have better convergence properties than others.

Well-known deterministic algorithms, such as steepest descent and Newton–Raphson, assume the performance metric to be optimized as a deterministic function of the tuning parameters; this is not the case for most guided missile systems. Stochastic algorithms such as the Robbins–Monro stochastic approximation (SA)[7] and infinitesimal perturbation analysis can process the noisy outputs produced by guided missile simulations, but they are difficult to apply to G&C algorithm tuning because they require direct measurements of the gradient of the cost function with respect to the parameters being optimized. Another class of SA algorithms does not rely on direct measurements of the cost gradient but rather estimates it using only measurements of the cost function. The cost-function gradient then is approximated in some intelligent manner and used to drive the optimization. Therefore, these SA algorithms are particularly well suited for tuning G&C algorithms. (Gradient-free algorithms that apply in a stochastic setting, such as genetic algorithms, simulated annealing, and evolutionary programming, are additional options that have not been explored by the authors and are discussed briefly later in this article.)

Finite difference stochastic approximation (FDSA) was developed by Kiefer and Wolfowitz[8] as an extension of the Robbins–Monro SA and requires one (or two) cost-function measurements per design parameter per iteration. With FDSA, systems having a large number of design parameters often require substantial computing resources. An alternative approach that avoids this drawback is the simultaneous perturbation stochastic approximation (SPSA) algorithm.[9] The SPSA algorithm requires only two cost-function measurements per iteration, regardless of the number of parameters to be tuned. When appropriately implemented, SPSA converges to the solution in roughly the same number of iterations as with finite-difference algorithms, offering overall time savings.

The SPSA algorithm has proven effective in addressing the challenges posed by the G&C problem domain as well as the challenges associated with automated approaches. SPSA is an iterative algorithm that uses simulation-generated, noisy cost-function measurements to approximate the gradient (derivative) of the cost function with respect to the design parameters. The gradient approximation then is used to drive the parameters toward their optimum values. SPSA handles noisy cost measurements well and is efficient compared to other similar algorithms in terms of the number of simulation runs required per optimization iteration, even for applications with large numbers of tuning parameters.

The purpose of this article is to describe the automated tuning of missile G&C algorithms using SPSA, focusing on the implementation challenges and solutions that have been identified. The challenges presented by the G&C problem domain pertaining to algorithm optimization are discussed, followed by a description of various performance metrics, or cost functions, that are useful for automatically tuning G&C algorithms. Then, the SPSA algorithm is presented, with a focus on implementation techniques that apply to G&C algorithm optimization. Finally, an example is provided that illustrates the challenges and efficacy of an automated approach.

G&C ALGORITHM CONSIDERATIONS

Many issues arise when applying automated-tuning algorithms, such as SPSA, to optimize G&C parameters in guided missile systems. Optimal tuning parameters may have complex dependencies on the engagement conditions. The problem of convergence to the optimal values may be sensitive to the choice of initial parameter values. In addition, bounds on the parameter search space may be required to ensure that missile stability is maintained and simulation numerical issues are avoided. The following sections discuss these issues in more depth.

Parameter Scheduling

As noted above, well-designed missile G&C algorithms should operate over a wide range of engagement scenarios. This requirement complicates the design task because the optimal algorithm parameter values rarely remain constant over the entirety of the scenario and flight-condition space. For instance, the engagements may occur at different speeds and altitudes, variables that affect the maneuverability of the missile by constraining the aerodynamic forces and moments the missile can generate. The inertial properties of the missile also vary throughout flight. As the motor burns—altering the mass, rotational inertia, and center of gravity—the lateral and angular acceleration capabilities of the missile change. These changing aerodynamic and inertial properties strongly affect optimal control parameter values. Thus, the output of an automated tuning may not be a single value for a given parameter but rather a number of values, each of which is optimal for a given scenario or flight condition and must be interpolated for use in real time during flight. This process of interpolating a set of parameter values is known as parameter scheduling and often is motivated by the use of linear design techniques.

A good example of the need for parameter scheduling to accommodate parameter dependence on missile and engagement properties is the autopilot time constant. Ideally, the controller should have the smallest time constant possible so that it can generate commands at the highest frequency and amplitude to which the missile system can respond. Because the missile system response time depends strongly on the dynamic pressure experienced through flight, it is customary to vary controller parameters as functions of dynamic pressure, q. Since SPSA and other automated optimization algorithms only tune a finite number of constant-valued parameters simultaneously, each parameter p_n ($n = 1, \ldots, N$) is expressed as a function f_n of q and a fixed number of constants $c_{n,m}$ ($m = 1, \ldots, M$):

$$p_n = f_n(q, c_{n,1}, \ldots, c_{n,M}) \,. \tag{1}$$

There are many potential forms for f_n, and analytical results often suggest a particular form. For example, we may know that p_n varies proportionally to q, and thus f_n can be written as

$$f_n(q, c_{n,1}) = c_{n,1}\, q, \tag{2}$$

and SPSA can tune directly for the single constant $c_{n,1}$. Often, the form of the function f_n may be unknown, and a more general approximation, $\hat{f}_n$, must be used. One candidate form is

$$f_n(q, c_{n,1}, \ldots, c_{n,M}) \approx \hat{f}_n(q, c_{n,1}, \ldots, c_{n,M}) = [c_{n,1} \cdots c_{n,M}] \begin{bmatrix} g_{n,1}(q) \\ \vdots \\ g_{n,M}(q) \end{bmatrix}, \tag{3}$$

where the $g_{n,m}$ are a set of basis functions of the dynamic pressure. Another candidate form for $\hat{f}_n$ is a linear spline of a set of M breakpoints $(c_{n,m}, q_m)$,

$$\hat{f}_n(q, c_{n,1}, \ldots, c_{n,M}) = q_{n,m-1} + \left(\frac{q_{n,m} - q_{n,m-1}}{c_{n,m} - c_{n,m-1}} \right) (q - q_{n,m-1}) \,, \tag{4}$$

when $q_{n,m-1} \leq q \leq q_{n,m}$ and assuming $q_{n,1} \leq q \leq q_{n,M}$. In the case of Eqs. 3 and 4, the problem of tuning p_n is cast in terms of tuning the M constant-valued parameters $c_{n,m}$. Figure 2 illustrates how the autopilot time constant may be scheduled with dynamic pressure.

Using an approximate form for f_n poses some additional difficulties within the tuning process. First, the form of $\hat{f}_n$ must be chosen to approximate f_n with sufficient accuracy. Specifically, there must be an optimal set of $c_{n,m}$ such that the performance degradation of using $\hat{f}_n$ versus f_n, in terms of changes in the performance measure, is acceptable. It is not known *a priori* whether the choice of $\hat{f}_n$ is sufficient. Therefore, the number of coefficients, M, used to parameterize the approximation $\hat{f}_n$ may be large.

In addition, the increase in the number of parameters to be identified is a problem that must be addressed. Each additional parameter creates an additional dimension in the search space and may require more optimization iterations to converge to the optimal solution, resulting in increased computation time. The computational requirements of the search algorithm can, however, be reduced if tuning parameters are known to be or can reasonably be assumed to be decoupled. Often, the designer can structure the optimization problem in a manner that intentionally decouples various tuning parameters. For example, in the case of tuning terminal homing missile engagements, the designer has the freedom to design $\hat{f}_n$ such that each $c_{n,m}$ only affects the value of $\hat{f}_n$ for a limited range of dynamic pressures. Thus, each $c_{n,m}$ can be tuned independently through the simulation of engagements, which exercise only the targeted dynamic pressure ranges.

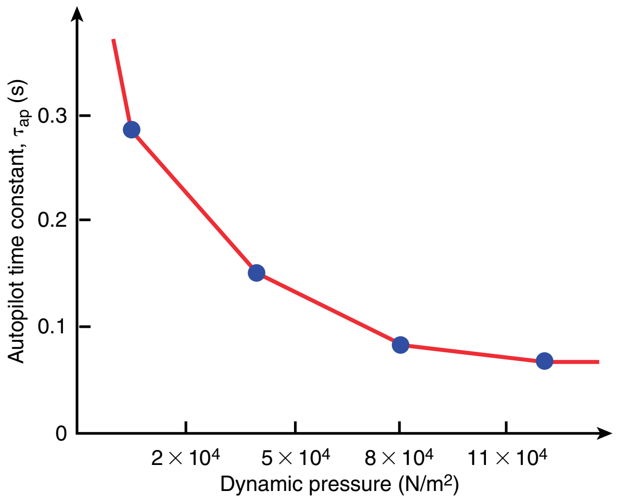


Figure 2. Example of parameter scheduling. The autopilot time constant is a function of dynamic pressure. Optimum values are designed at certain dynamic pressure points, and intermediate values are calculated via interpolation during flight. The red curve represents scheduled parameter values, and the blue dots represent analytically derived, or tuned, parameter values.

Initial Conditions

Iterative optimization algorithms like SPSA require a set of initial values for the parameters to be optimized. Determining appropriate initial values can be difficult for certain G&C algorithms. Although analytical results may provide nearly optimal starting values for certain parameters, this is rarely the case. As previously noted, one of the motivations for an automated numerical search is that the designer's understanding of the performance sensitivities of certain parameters can be quite poor. Algorithms such as IGC are complex and contain a large number of tunable parameters that affect performance in nonintuitive ways, making the choice of initial values difficult.

Requirements for initial parameter values are strict. For complex and highly nonlinear systems, a successful automated search cannot begin from any arbitrary set of parameter values. First, such an arbitrary set of values may yield such suboptimal performance that the results are essentially meaningless. For instance, if the missile spins out of control, any change in the locally computed gradient of performance with respect to parameter change contains no useful information. Therefore, an initial parameter set must, at a minimum, enable the missile to fly stably and to home in on the target.

A stable missile, however, is not sufficient. The tuning parameters also must be initialized to values that are sufficiently near their optimal values. If the missile is stable but the tuning parameters are initialized with values much different from the optimal values, then the optimization algorithm may converge to a locally optimal value set that is substantially different from the globally optimal set. Thus, without an appropriate initial parameter set, an automated search may not converge to the parameter values that achieve the desired maximum performance.

Because of the stated requirements for initial conditions of the automated search and the difficulties in meeting these requirements, many tuning problems may require a manual search (e.g., a coarse-grid search) to establish acceptable initial conditions because the automated search is expected to fine-tune the parameters to an optimal set of values. The search, however, should not be blind. A coarse-grid search over the parameter space should begin the process of finding initial values for the automated search. Parameter sets that yield the best results then must be evaluated to eliminate sets that may be unacceptable for secondary reasons. At this point, the designer must essentially guess which of the remaining initial condition candidates is believed to be closest to the globally optimal set. Scrutiny of internal simulation states can eliminate parameter values that yield bizarre or unexpected results. For example, if a set of parameters yields poor internal estimates of the line of sight between the missile and target (a key guidance quantity), the parameter set is rejected. It ultimately may be necessary to attempt automated tuning with various sets of initial conditions.

Parameter Search Space

Another difficulty associated with the automated tuning of G&C algorithms is that many parameter values have limited ranges. For example, because the autopilot time constant is, by definition, limited to the positive real axis, the tuning algorithm should never produce a negative value. However, this bound may not be sufficient. The autopilot time constant may have a lower bound greater than zero, which is necessary to ensure the numerical stability of the autopilot. These types of bounds are common in G&C algorithms. Guidance filters, for instance, also may become numerically unstable with certain parameter choices. It is important to define acceptable ranges for each tuning parameter over which the missile-engagement scenarios are being optimized for performance.

Choosing acceptable boundaries does not usually pose undue difficulty. The most challenging boundaries to select are not immediately apparent through analysis, such as those preventing numerical instability. These bounds can be found through a manual search. The exact boundary often is not necessary; it is usually sufficient to bound each parameter within a range that contains the optimal parameter value. A simple grid search may identify extreme parameter values for which performance is poor but with which the G&C algorithms can function without any numerical issue.

Once identified, the parameter bounds are easily accommodated in the implementation of SPSA used here. The mechanics of how these constraints are incorporated into SPSA are discussed in *Parameter Constraints*, later in this article.

PERFORMANCE METRICS

The SPSA optimization algorithm is driven by a single, scalar-valued cost function. For the optimization to work well, this cost function must capture the designer's view of the optimal performance of the algorithm. Ideally, a cost function captures all of the important system characteristics to be optimized and no others. In addition, the choice of a cost function must take smoothness and ill conditioning into account. Overly noisy or discontinuous cost functions impede the convergence of the SPSA algorithm to viable optimums because of the erratic parameter perturbations they induce.

For our problem domain, the capacity to negate a threat under widely varying engagement conditions defines the efficacy of the missile with a particular G&C configuration. With this basic goal in mind, any tuning of the G&C algorithm must seek to increase the probability that the guided missile destroys a given threat.

Three statistics that are used to quantify the concept of successful intercept are probability of hit (PH), probability of damage (PD), and adjusted probability of damage (APD). (A successful intercept for missiles that use kinetic warheads, i.e., hit-to-kill missiles, generally is characterized by the "miss" distance to a selected aimpoint on the target.) Estimates of these quantities are statistically derived from the simulation-based Monte Carlo analysis of particular intercept scenarios. The simplest of the three quantities, PH, refers to the probability that the closest point of approach (CPA) of the missile to the target (i.e., miss distance) falls within the lethal radius of the warhead. PH is calculated as the percentage of runs in a Monte Carlo set with the CPA less than the warhead lethal radius. PH provides a good measure of guidance performance but falls short of capturing the essence of a successful intercept because a CPA within the lethal warhead radius does not guarantee target destruction or even damage.

A separate endgame simulation is required to generate PD statistics using outputs generated by the primary missile simulation. Depending on a variety of factors and more complicated than PH, PD provides an accurate measure of fragmenting warhead lethality. Assuming that the G&C system successfully guides the missile close to the target, the warhead fuze generally detects the target and initiates the warhead detonation sequence. At the time of fuzing, endgame characteristics—such as instantaneous missile and target velocities and positions, missile body rates, relative angles between the missile and target, and missile and target angles of attack (AOAs)—influence the amount and type of damage sustained by a target. In addition to these engagement-specific characteristics, the type of target and warhead employed in the missile also affect the target damage. High-fidelity endgame simulations model the warhead/target interaction at intercept based on complex mathematical codes and warhead test data. Based on these considerations, the endgame simulation predicts the probability of target damage for an individual engagement to yield a PD value, which is averaged over a Monte Carlo set to obtain an estimate of the PD for the chosen intercept point.

APD is derived from PD. APD is simply the statistical PD value adjusted to discount engagements that have exceeded the maximum acceleration limit of the missile airframe. In reality, physical airframe properties dictate a maximum acceleration achievable by a missile, but missile simulations do not typically terminate intercept scenarios on the basis of maximum calculated accelerations. By erroneously counting PD values for engagements that would result in missile structural failure, PD averages can be optimistic. The calculation of APD treats all engagements in a Monte Carlo set that exceed the acceleration limit as having PD values of zero. After this adjustment, the PD is averaged over the entire Monte Carlo set to yield the APD of the set.

Miss distance and the related PH statistic are the simplest cost functions used to express missile performance. However, they can be misleading for at least two reasons. First, the ultimate objective is for the warhead to be able to destroy the target, and this lethality depends on secondary factors as well as on miss distance. Thus, smaller miss distances do not always translate into a higher PD. In addition, miss-based cost functions do not take into account stability, limits on the expenditure of control energy, or the structural integrity of the airframe during flight.

The APD statistic described above appears reasonable in the context of the previous factors because it penalizes overly aggressive G&C designs that tax the structural integrity of the airframe. However, because the APD statistic is discontinuous and can exhibit ill conditioning, it is difficult to use as a cost function for automated tunings. APD statistics are bimodal by definition, since simulation runs that do not meet acceleration limiting requirements have zero performance, while simulation runs meeting the requirements retain their PD value, which usually is either one or zero. Unless large numbers of simulation performance statistics are averaged to estimate mean APD, a cost function composed exclusively of APD is extremely discontinuous between tuning iterations, making SPSA gradient approximations difficult. One method of circumventing this problem is to scale the raw PD measure functionally using information on the magnitude of the maximum achieved acceleration during an engagement, producing a "pseudo-APD" measure. With this technique, the scaling function reduces the PD value exponentially, and continuously, as the maximum acceleration approaches the missile acceleration limit.

Other system characteristics must be considered in the tuning process. For example, a tuning that is driven solely by miss distance or lethality can require an unrealistically high expenditure of control energy. In addition, nonlinear effects such as fin rate limiting, fin position limiting, and acceleration limiting break the control loop of a system, leading to unpredictable sensitivities to tuning adjustments. The addition of augmenting cost-function components that penalize nonlinear or overly aggressive system behaviors can yield designs more reasonable than those based on miss distance or lethality alone. Typically, these secondary cost components are scaled and combined with miss distance to form a composite cost function. Some examples of useful secondary cost components are summarized below.

- **Maximum acceleration cost:** The maximum acceleration cost is computed as a function of the maximum lateral acceleration achieved during missile flight. Appropriate scaling of this cost component can encourage more stable missile behavior by penalizing large achieved accelerations, thereby minimizing the nonlinear and saturation effects of excessive acceleration limiting. Care must be taken, however, to avoid overly penalizing missile acceleration, because doing so adversely affects miss distance.

- **Maximum fin rate cost:** This cost component is a function of the maximum achieved fin rate of the missile. Typically, maximum fin rates above a critical value specified in the tuning setup are increasingly penalized according to a second-order polynomial. Judicious adjustment of the cost-function shape and weighting relative to the primary miss-based cost component forces the tuning to progress in a direction that avoids fin rates that exceed the maximum achievable hardware rates.

- **Fin activity cost:** Related to the maximum fin rate cost concept, the fin activity cost component is the integral of the squares of the achieved fin rates throughout the engagement. As a "roll-up" measure, the fin activity cost of an engagement provides information on the total fin energy expended during an engagement but can mask any indication of the magnitude or duration of fin position or rate saturation.

- **Fin command error cost:** Also a cumulative measure, this cost component consists of the error between controller commanded and achieved fin positions integrated over time. Engagements resulting in saturated fin rates and positions cause larger fin command error and, thus, large roll command error cost. As with a fin activity cost component, the integral form of this type of cost component obfuscates the distinction between small but persistent errors and large, transient errors.

Many practical issues arise when considering the implementation of a cost function in automated tuning. Chiefly, these issues pertain to the proper conditioning of the cost and sufficient sampling of the cost space as well as to computational limitations and considerations. Engagements with excessive misses or large secondary cost statistics tend to disproportionately drive the evolution of an optimization through excessive contribution to the tuning cost, leading to poor overall system performance. To avoid this problem, it is common to truncate cost statistics to a maximum value. This cost-function limiting helps to ensure that only relatively well-behaved simulation results drive a tuning. Computer system failure during the course of a tuning can produce erroneous cost measurements that skew tuning results in a similar fashion to ill-behaved engagements. To address this difficulty, automated optimization algorithms implemented here employ fault-handling logic to identify and disregard failed cost-function evaluations attributable to computer system collapses, calculating cost based only on successful simulation runs.

A tuned G&C controller should perform near optimally for a large variety of engagement scenarios and noise conditions, necessitating the simultaneous tuning of many different simulated intercept scenarios. To this end, a single SPSA cost measurement often involves the averaging of several individual costs calculated for a group of simulated engagements. The selected group of tuning engagements can either be static over the course of a tuning or be randomly drawn from a pool of candidates at each tuning iteration. Thus, calculating the average cost of multiple engagements for each tuning iteration may require several individual simulation runs and extensive computational resources. Another possibility in combining cost statistics for multiple engagements is to use a chosen percentile of the statistics as the overall cost. For example, a tuning of the median cost of a group of engagements would use the 50th percentile cost value of the group at each iteration as the tuning cost. This approach partially ameliorates problems associated with the undue influence of outliers on the mean cost of a group of engagements.

SPSA OPTIMIZATION ALGORITHM

For a given G&C algorithm and an appropriately designed cost function, the automated-tuning task largely becomes a matter of properly setting up and executing the optimization algorithm, in this case the SPSA. The basic form of the SPSA obtains two measurements of the cost function by randomly perturbing the tuning parameters and uses the cost measurements to approximate the gradient of the cost function with respect to the tuning parameters. The SPSA gradient approximation then is used to update the parameters in the direction of the gradient. This basic form of the algorithm works well for relatively simple applications, but for the tuning of missile G&C algorithms, several practical issues present themselves. A description of the basic form of SPSA, along with several implementation issues and the solutions that work well for the missile G&C problem domain, are discussed below.

Basic SPSA Algorithm

To frame the problem, let us define $\boldsymbol{\theta}$ to be a vector of p real parameters to be optimized, $L(\boldsymbol{\theta})$ to be the real-valued, scalar cost function of the parameters, i.e.,

$$L(\boldsymbol{\theta}) \Big| \Re^p \rightarrow \Re \,, \tag{5}$$

and $\mathbf{g}(\boldsymbol{\theta})$ to be the gradient of the cost function:

$$\mathbf{g}(\boldsymbol{\theta}) = \partial L(\boldsymbol{\theta}) / \partial \boldsymbol{\theta} \,. \tag{6}$$

The goal of the optimization problem is to determine the set of parameters $\boldsymbol{\theta}^*$ that minimizes the cost function $L(\boldsymbol{\theta})$. Assuming that the cost function is sufficiently smooth and that only a global minimum exists, the mathematical objective of the optimization problem is to find the $\boldsymbol{\theta}^*$ that satisfies a zero-derivative condition for the cost function:

$$\mathbf{g}(\boldsymbol{\theta}^*) = \frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \Big|_{\boldsymbol{\theta}^*} = 0 \,. \tag{7}$$

Thus, for the single global minimum problem, the gradient of the cost function assessed for the optimal vector of tuning parameters $\boldsymbol{\theta}^*$ is zero-valued. Because we do not have direct access to the cost function, only noisy cost-function measurements of the form

$$y(\boldsymbol{\theta}) = L(\boldsymbol{\theta}) + \text{noise} \tag{8}$$

are available, and the gradient in Eq. 7 must be approximated as a function of the measurements y.

What differentiates SPSA from the more traditional finite-difference counterparts is the way these gradient approximations are made. A two-sided finite-difference algorithm perturbs each of the p parameters one at a time, both positively and negatively, and forms the gradient approximation component-wise by dividing the change in the cost function at the perturbed parameters by the perturbation amount, thus requiring $2p$ cost-function measurements. SPSA, however, forms the gradient approximation with only two cost-function measurements, regardless of the number of parameters. This economy is accomplished by randomly but simultaneously perturbing each parameter twice to obtain two measurements of the cost function. Each parameter is perturbed independently by a random amount chosen from a zero-mean probability distribution that satisfies certain conditions.[9] A simple and typical choice for the probability distribution is the equally weighted ±1 Bernoulli probability density function (PDF). Note that the common uniform and normal PDFs do not satisfy the conditions specified in Ref. 9 and cause the algorithm to fail. The ith component of the SPSA gradient approximation at the kth optimization iteration is thus formed as

$$\mathbf{g}_{ki}(\boldsymbol{\theta}_k) = \frac{y(\boldsymbol{\theta}_k + c_k \boldsymbol{\Delta}_k) - y(\boldsymbol{\theta}_k - c_k \boldsymbol{\Delta}_k)}{2 c_k \boldsymbol{\Delta}_{ki}} \,, \tag{9}$$

where $\boldsymbol{\Delta}_k$ is the p-dimensional random perturbation vector described above. After having obtained the gradient approximation at the kth optimization iteration, the estimate of the optimum parameter vector is updated using the standard recursive SA update equation,

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - a_k \mathbf{g}_k(\boldsymbol{\theta}_k) \,, \tag{10}$$

where a_k is a positive scalar gain sequence chosen to ensure convergence of the algorithm (see *SPSA Convergence*). Equation 10 is iterated until the termination criteria are met. Termination criteria are usually heuristic; typically, the algorithm is terminated if there has been little change in the cost function (or in $\boldsymbol{\theta}_k$) over several successive iterations or if a prespecified maximum number of iterations has been reached.

Gradient Averaging

When the noise levels in the cost-function measurements are high, which is usually the case for medium- to high-fidelity guided missile simulations, the gradient approximations formed using SPSA are potentially of poorer quality compared to those formed with finite-difference algorithms and can cause unstable optimization performance. This difficulty is overcome by using a level of gradient averaging, at the expense of additional cost-function measurements.[10] Gradient averaging is accomplished by making several successive approximations given by Eq. 9 (with independent values of the random perturbation amount $\boldsymbol{\Delta}_k$ for each approximation) and averaging the resulting gradient estimates component-wise. In this case, the gradient approximation becomes

$$\mathbf{g}_k(\boldsymbol{\theta}_k) = q^{-1}\sum_{i=1} \mathbf{g}_k^{(i)}(\boldsymbol{\theta}_k), \tag{11}$$

where $\mathbf{g}_k^{(i)}\ (i = 1,\ldots,q)$ are the q gradient estimates formed as in Eq. 9. Thus, for an averaged gradient approximation consisting of q gradient estimates, $2q$ cost-function measurements are required per optimization iteration. Empirical evidence suggests that a small number of gradient estimates, $q = 3$, for example, are sufficient to reduce the effective noise contribution and yield satisfactory optimization performance. Even with these additional cost-function measurements (now $2q = 6$ per optimization iteration), the SPSA algorithm still affords significant time savings compared to the $2p$ cost-function measurements required by finite-difference algorithms for the potentially large p associated with G&C algorithms.

Normalization via Parameter Scaling

Because the SPSA algorithm forms the gradient approximation by perturbing all of the parameters simultaneously by the same amount, difficulties arise when parameters have widely different magnitudes. The perturbation amount, critical to the performance of the optimization algorithm, must be large enough so the gradient approximations are not driven solely by measurement noise but small enough to ensure good-quality gradient approximations.

This concern is addressed by applying a matrix scaling to the parameter vector to map the parameters to values with similar magnitudes. The optimization algorithm operates on the normalized values, which then are transformed back into the real parameter space before being sent to the simulation to obtain the cost-function measurements. One simple choice for the scaling is a matrix whose *i*th diagonal entry equals the reciprocal of the initial value of the *i*th parameter:

$$\mathbf{S} = \begin{bmatrix} \frac{1}{\boldsymbol{\theta}_{0_1}} & & & 0 \\ & \frac{1}{\boldsymbol{\theta}_{0_2}} & & \\ & & \ddots & \\ 0 & & & \frac{1}{\boldsymbol{\theta}_{0_p}} \end{bmatrix}. \tag{12}$$

This scaling matrix yields normalized parameters, all with unity initial values. Using this matrix as a starting point, the individual diagonal entries then can be fine-tuned to allow a more or less aggressive search of the parameter space as desired by the user.

SPSA Convergence

Formal convergence of the SPSA is shown in Ref. 9 to be conditioned on the two gain sequences a_k and c_k in Eqs. 9 and 10. Typically, these gain sequences take the following forms:

$$\begin{aligned} a_k &= \frac{a}{(A + k)^{\alpha}} \\ c_k &= \frac{c}{(k)^{\gamma}} . \end{aligned} \tag{13}$$

For these sequences, $\alpha = 1$ and $\gamma = 1/6$ are asymptotically optimal values, but implementation and numerical evidence show that choosing smaller values usually yields better performance by generating larger parameter step sizes. In Ref. 10, $\alpha = 0.602$ and $\gamma = 0.101$ are reported to be the lowest theoretically valid values and work well in practice. For tuning missile G&C algorithms, it has been found that reducing the value of α even lower can often help the tuning process continue to move parameters to more favorable values before the a_k sequence overly decays.

The perturbation gain c should be proportional to the level of the measurement noise in $y(\boldsymbol{\theta})$. A noisier cost function requires a larger value for c so that the perturbations are large enough to generate gradients that are not driven by noise alone. One benefit of applying the unity-magnitude scaling described above, where all scaled parameters have unity initial values, is that the value of c corresponds to a percentage value for the initial perturbation. A value of $c = 0.1$, for example, results in a ±10% initial perturbation for each parameter. Practice has shown that small perturbations (5% or less) are sufficient for cost functions with relatively low noise levels. Particularly ill-behaved cost functions, however, may require perturbations on the order of 30–50%.

The parameter update gain a and the optimization stability parameter A in the a_k sequence also can be chosen in a semiautomatic fashion.[10] The parameter A should be chosen so that it is equal to no more than 10% of the expected number of optimization iterations. Then, the parameter a should be chosen so that a_0 times the initial gradient magnitude, $|\mathbf{g}_0(\hat{\boldsymbol{\theta}}_0)|$, is approximately equal to the largest desired change in magnitude in the elements of $\boldsymbol{\theta}$ during the early iterations. The initial gradient magnitude is obtained by forming several gradient averages at the initial $\boldsymbol{\theta}_0$ (with c chosen as described above).

Parameter Constraints

An additional practical concern is how to impose parameter constraints. For G&C algorithms, minimum and maximum parameter values often must be enforced. These constraints can either be real or imposed by the designer. An example of a real parameter constraint is that an autopilot time constant cannot be negative. An example of an imposed constraint is if the designer wishes the measurement noise covariance value to remain within some region of the known sensor noise characteristics.

A simple modification to the SPSA algorithm is presented in Ref. 11, which handles this constrained optimization case. The constraints are enforced both in the parameter update step in Eq. 10 and in the perturbations that form the gradient approximation in Eq. 9. For the parameter update step, any parameter that breaches the boundary of the allowable parameter space is projected back to the closest point on the boundary (Fig. 3). Regarding the perturbations, if the perturbed value violates any of the constraints, the point in the parameter space about which the gradient approximation is formed is projected to a nearby point so that the constraints are enforced.[11]

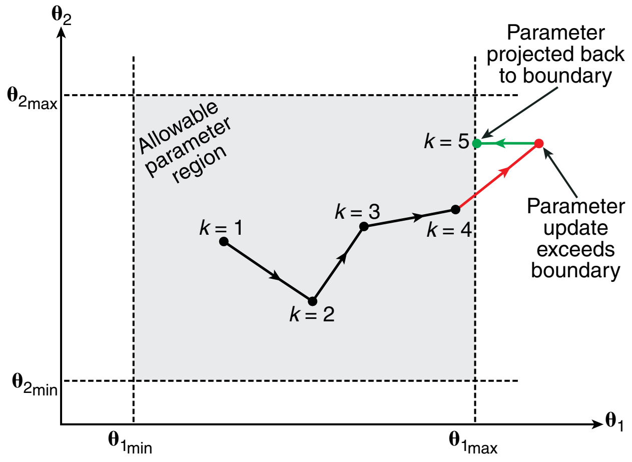


Figure 3. Example of parameter constraint enforcement for a case with two tuning parameters. At iteration $k = 5$, the optimization algorithm updates the parameter to a value outside the allowable bounds. To enforce the parameter constraint, the parameter value is projected back to the closest point on the allowable parameter boundary.

EXAMPLE

The efficacy of an automated, simulation-based approach to G&C algorithm tuning is demonstrated by applying the SPSA technique to the IGC algorithm.[1] The IGC system is a good candidate for automated simulation-based tuning because of the lack of analytical design techniques, the large number of adjustable algorithm parameters, and the interdependence of these parameters with regard to algorithm performance.

The IGC concept considers the complete integration of the missile guidance filter, guidance law, and autopilot. Traditional G&C systems contain separate, decoupled algorithms for each of these features (Fig. 4).

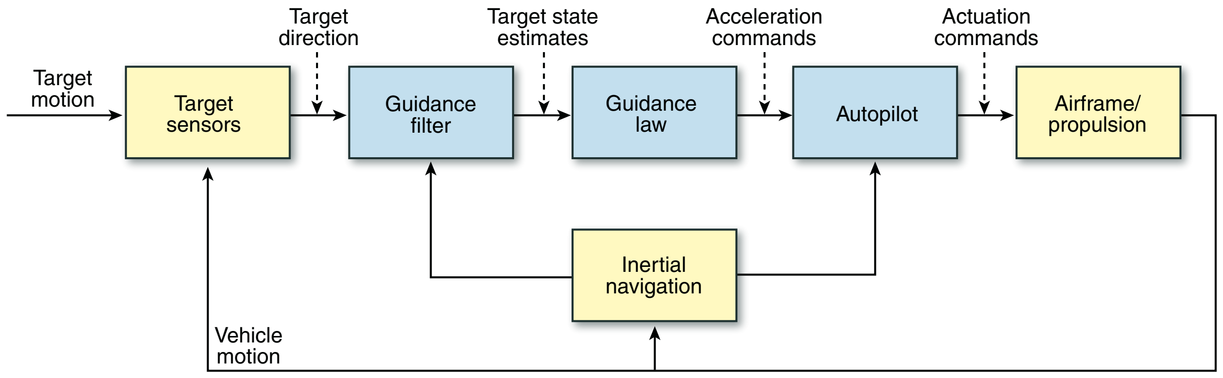


Figure 4. A classical G&C architecture consists of decoupled guidance filter, guidance law, and autopilot components.

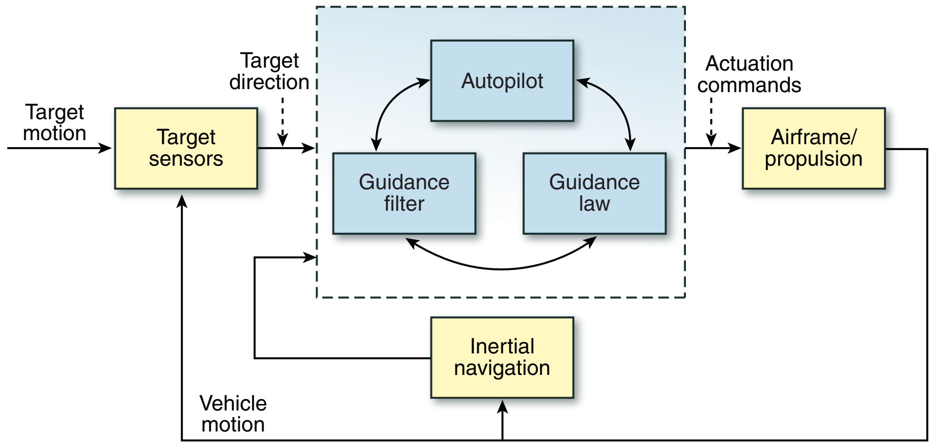


Figure 5. A fully integrated G&C architecture combines the three traditionally separate guidance filter, guidance law, and autopilot components into a single integrated algorithm.

An integrated architecture, on the other hand, considers all three components in a single, unified algorithm, thereby exploiting any functional interaction between the guidance and flight-control functions (Fig. 5). The IGC paradigm also provides a natural framework for simultaneous optimization of all three functions.

At the heart of the IGC algorithm is the integrated plant model. As a fully integrated G&C algorithm, the plant model includes homing loop kinematics and target acceleration states (i.e., the guidance elements) as well as missile dynamics and actuator states (i.e., the flight-control elements):

$$
\mathbf{x}(t)=\left[\begin{array}{c}
\left.\begin{array}{c} r_y^{\mathrm{G}}(t)\\ \dot{r}_y^{\mathrm{G}}(t)\\ r_z^{\mathrm{G}}(t)\\ \dot{r}_z^{\mathrm{G}}(t)\end{array}\right\}\text{Relative kinematics}\\ \hdashline
\left.\begin{array}{c} \alpha(t)\\ \beta(t)\\ p(t)\\ q(t)\\ r(t)\\ \bar{\varphi}_{\mathrm{err}}(t)\end{array}\right\}\text{Body motion}\\ \hdashline
\left.\begin{array}{c} \delta_r(t)\\ \dot{\delta}_r(t)\\ \delta_p(t)\\ \dot{\delta}_p(t)\\ \delta_y(t)\\ \dot{\delta}_y(t)\end{array}\right\}\text{Actuator}\\ \hdashline
\left.\begin{array}{c} a_{T_y}^{\mathrm{G}}\\ a_{T_z}^{\mathrm{G}}\end{array}\right\}\text{Target}
\end{array}\right]
\quad
\mathbf{y}(t)=\left[\begin{array}{c}
r_y^{\mathrm{G}}(t)\\ r_y^{\mathrm{G}}(t)\\ \hdashline
a_{m_y}(t)\\ a_{m_z}(t)\\ \hdashline
p(t)\\ q(t)\\ r(t)\\ \varphi_{\overline{\mathrm{err}}}(t)\\ \hdashline
\delta_r(t)\\ \delta_p(t)\\ \delta_y(t)\\ \hdashline
F_{x_0}(t)\\ g(t)
\end{array}\right]
\quad
\mathbf{u}(t)=\left[\begin{array}{c}
\delta_{r_c}(t)\\ \delta_{p_c}(t)\\ \delta_{y_c}(t)
\end{array}\right]
\quad
\mathbf{z}(t)=\left[\begin{array}{c}
r_y^{\mathrm{G}}(t)\\ \dot{r}_y^{\mathrm{G}}(t)\\ r_y^{\mathrm{G}}(t)\\ \dot{r}_y^{\mathrm{G}}(t)\\ \hdashline
\alpha(t)\\ \beta(t)\\ p(t)\\ q(t)\\ r(t)\\ \bar{\varphi}_{\mathrm{err}}(t)\\ \hdashline
\dot{\delta}_r(t)\\ \dot{\delta}_p(t)\\ \dot{\delta}_y(t)\\ \delta_{r_c}(t)\\ \delta_{p_c}(t)\\ \delta_{y_c}(t)
\end{array}\right]
\quad (14).
$$

Equation 14 illustrates that the IGC state vector $\mathbf{x}(t)$ captures both guidance and missile dynamics. The relative kinematics and target states capture the target-missile relative geometry and target acceleration to capture the guidance portion of the problem. The body motion and actuator states model the missile dynamic states related to flight control. The input vector $\mathbf{y}(t)$ contains commonly available measured or reconstructed quantities such as missile accelerations, body rates, and the missile/target relative positions. The output of the IGC plant $\mathbf{u}(t)$ consists of the pitch-yaw-roll angular tail position commands (traditionally generated by an autopilot). The performance output vector $z(t)$ contains the relative kinematic, missile dynamics, and control signal elements needed in the design synthesis to specify IGC performance.

Essentially, the job of the IGC algorithm is to translate the sensed inputs $\mathbf{y}(t)$ into tail actuation commands $\mathbf{u}(t)$. To accomplish this task, the IGC plant model is discretized and brought into a linear-looking, state-dependent form at each sample instant. The resulting state-space formulation is

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k + \mathbf{D}_k\mathbf{w}_k \\ \mathbf{y}_k &= \mathbf{C}_k\mathbf{x}_k + \mathbf{E}_k\mathbf{u}_k \\ z_k &= \mathbf{H}_k\mathbf{x}_k + \mathbf{G}_k\mathbf{u}_k . \end{aligned} \tag{15}$$

A solution to a soft-constrained linear-quadratic dynamic game then is solved, subject to Eq. 15, to yield the tail actuation commands. (The interested reader is directed to Ref. 1 for a full discussion.)

The key adjustable parameters in the IGC algorithm are entries in the state matrices in Eq. 15 and fall into four distinct parameter classes. The process disturbance weights (entries in the $\mathbf{D}_k$ matrix) allow some adjustment of the controller to help mitigate the effects of modeling uncertainty. Likewise, the measurement disturbance weights (entries in $\mathbf{E}_k$) allow adjustment of the controller bandwidth to mitigate the effects of sensor noise on the system. The performance output weights (entries in $\mathbf{G}_k$ and $\mathbf{H}_k$) affect the penalization of individual states and control effort under the cost integral of the dynamic game formulation.[1] Plant modeling parameters (entries in $\mathbf{A}_k$), such as the roll control and target time constants, are tuned to provide acceptable performance. The significant tuning parameters are shown in Fig. 6, with those in red being the first-order performance drivers that are included in this example tuning.

The IGC system is implemented in a 6-degree-of-freedom (6-DOF) terminal homing simulation that forms the basis for the tuning of these systems. This simulation incorporates a generic, fully coupled, nonlinear aerodynamics model (via table lookup), structural filtering, and fin command-processing delays. The inertial measurement unit gyro and accelerometer models include second-order dynamics, additive Gaussian noise, and scale factor and misalignment errors. The gyro model also includes a drift component. Actuators are modeled with fin rate and position limits and assume second-order

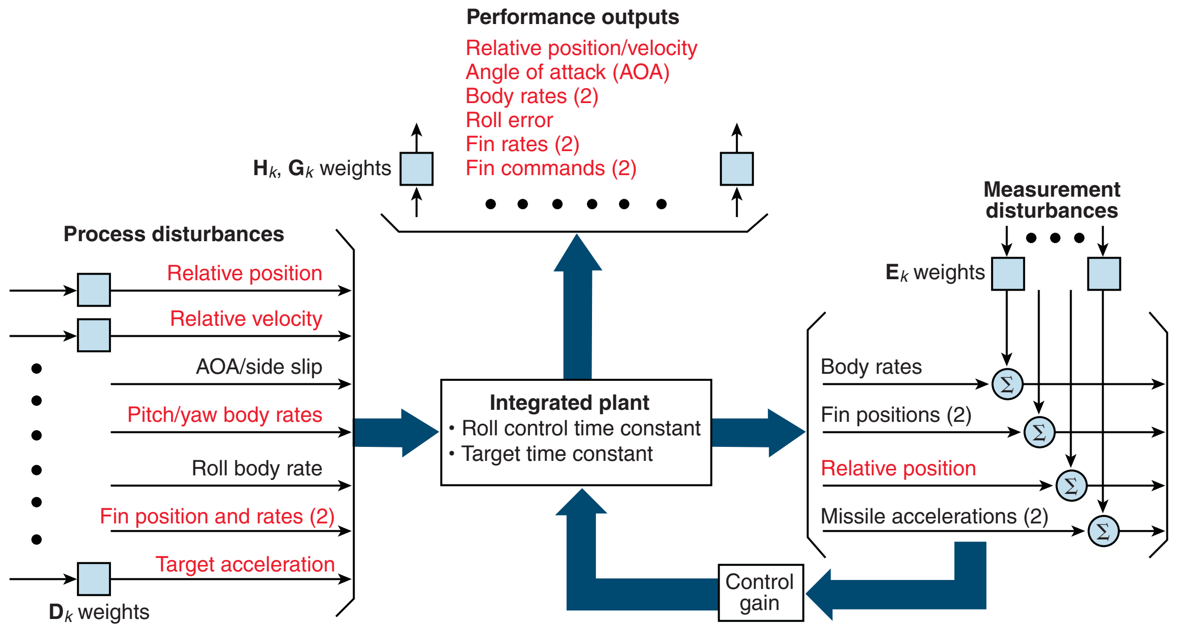


Figure 6. The IGC system consists of performance output, process disturbance, and measurement disturbance weights as well as integrated plant parameters. The parameters included in this example that are first-order performance drivers are shown in red.

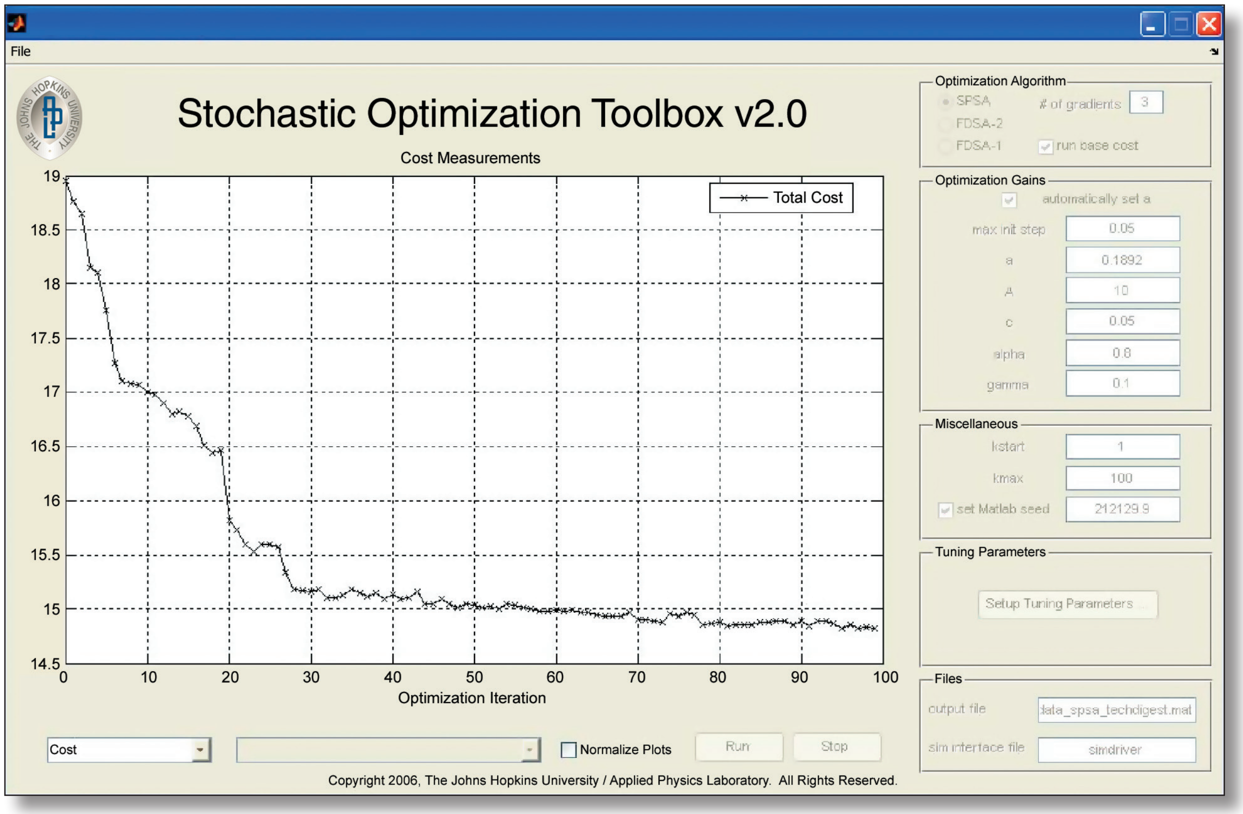
dynamics. Seeker angle measurements are corrupted by glint noise, radome-induced boresight errors (via table lookup), and angle noise consistent with an active RF seeker. Gaussian noise is added to true range and range rate to form range and range rate measurements. The 6-DOF simulation tactical ballistic missile (TBM) target generator models ballistic slowdown and coning maneuvers with variable acceleration capabilities and maneuver periods.

For this example tuning, a single terminal homing engagement is simulated. The threat is a TBM executing a coning maneuver. The initial interceptor/threat relative geometry is chosen to yield a realistic heading error. Because the IGC algorithm gains are scheduled with altitude, the intercept point for this example is chosen so that the missile flight is restricted to within a single altitude regime. Only the IGC gains associated with that altitude regime are included in the tuning.

The cost function for this example is formulated as a weighted sum of miss distance (or CPA), a measure of fin activity, and the maximum achieved lateral acceleration of the missile. The fin activity component is formed as the sum of the integrals over time of the squared fin angular accelerations in the pitch, yaw, and roll channels. The maximum acceleration component is computed as the maximum achieved lateral acceleration during flight. These figures of merit are averaged over 25-run Monte Carlo sets for each cost-function measurement. The weightings are set such that miss distance is the dominant component, followed by maximum acceleration and finally fin activity.

Each SPSA gradient is formed as the average of three independent gradient estimates. Because the initial values for the IGC tuning parameters range over 7 orders of magnitude, unity-magnitude scaling is employed so the mapped parameters all have unity initial values. With unity-magnitude scaling, the value for the SPSA gain parameter *c* is set to 0.05 to yield a 5% initial perturbation amount for each parameter. The value for *a* is automatically determined so that the most any parameter can change on the first iteration does not exceed 5%. As is typical, the values for α and γ are experimentally determined to yield stable algorithm behavior and good convergence properties.

Stochastic Optimization Toolbox version 2.0, developed at APL, is the software tool used to implement the SPSA algorithm for this example. The optimization was allowed to run for 100 iterations. The resulting composite cost-function evolution is shown in Fig. 7. The SPSA algorithm was successful in substantially reducing the cost function over the 100 optimization iterations, yielding an approximately 20% reduction.



Optimization Algorithm
SPSA ● # of gradients 3
FDSA-2
FDSA-1 ✓ run base cost

Optimization Gains
✓ automatically set a
max init step 0.05
a 0.1892
A 10
c 0.05
alpha 0.8
gamma 0.1

Miscellaneous
kstart 1
kmax 100
✓ set Matlab seed 212129.9

Tuning Parameters

Files
output file data_spsa_techdigest.mat
sim interface file simdriver

Copyright 2006, The Johns Hopkins University / Applied Physics Laboratory. All Rights Reserved.

Figure 7. Screenshot of Stochastic Optimization Toolbox version 2.0. The graphical user interface allows the setting of SPSA and simulation parameters as well as the viewing of optimization data. The composite cost-function evolution is shown. After 100 optimization iterations, the cost has been reduced by approximately 20%.

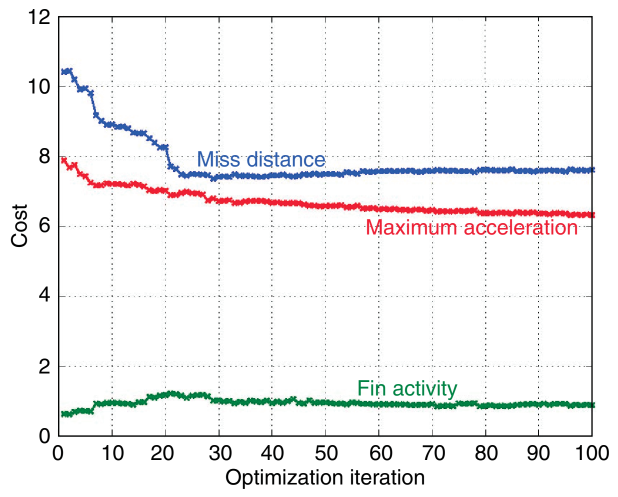


Figure 8. Evolution of the cost-function components during optimization. Miss distance was chosen to be the largest contributor to the cost, followed by maximum missile acceleration and fin activity.

Figure 8 illustrates the importance of the appropriate selection of the cost-component weightings. The weightings on the miss distance and maximum acceleration components caused them to dominate the optimization, and these two components exhibit aggressive downward trends over the first 20 or so iterations. The fin activity component was weighted to be a relatively minor contributor to the overall cost, and its value actually increased over the first 20 or so iterations. After that point, however, the fin activity cost began a modest reduction, and the miss distance cost actually increased slightly. A different set of weightings, particularly one where the fin activity component was weighted more heavily, would have yielded substantially different tuned parameter values.

The movement of three of the IGC gains is illustrated in Fig. 9. The normalized values are shown, so all three parameters have unity initial values. The AOA performance weight (AOA_PERF_WEIGHT) is used in the IGC algorithm to penalize large missile AOAs. It is a significant contributor to all three cost components. The smaller the value, the more maneuverable the missile can be, which can improve miss distance; a smaller value, however, also will generally increase the fin activity and maximum acceleration cost components. This parameter increases smoothly by approximately 80% over the course of the optimization. The target process disturbance weight (TARG_PDIST_WEIGHT_BKP) affects the overall bandwidth of the IGC system and is also a first-order performance driver. As seen in Fig. 9, its value increases by about 30% over the optimization. The roll angle error performance weight (ROLLERR_PERF_WEIGHT) is used to adjust the missile roll response. This parameter is known to contribute significantly less to the three cost-function components of this example. As long as its value is within a reasonable region, system performance is not very sensitive to changes in its value, as is evident because its value does not change much over the course of the optimization.

The final values for the tuning parameters generated above can be evaluated over a larger set of Monte Carlo runs for the engagement scenario used in the tuning. For example, Fig. 10 shows the miss distance cumulative probability distributions for the original and tuned IGC algorithms based on 100-run Monte Carlo sets. The

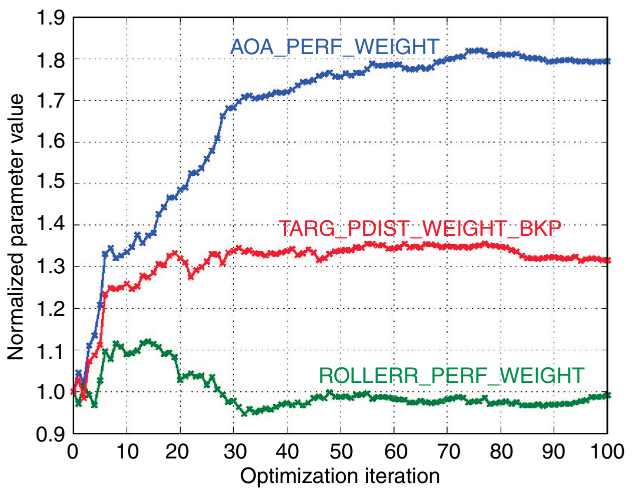
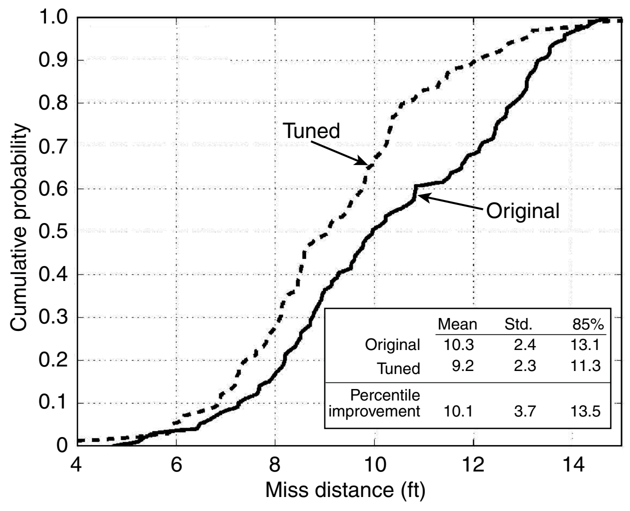


Figure 9. Progression of three IGC parameters during optimization. The AOA_PERF_WEIGHT and TARG_PDIST_WEIGHT_BKP parameters, key IGC performance drivers, moved significantly, whereas the ROLLERR_PERF_WEIGHT, a less critical parameter, moved very little.



	Mean	Std.	85%
Original	10.3	2.4	13.1
Tuned	9.2	2.3	11.3
Percentile improvement	10.1	3.7	13.5

Figure 10. Miss distance cumulative probability results for original and tuned IGC systems. Better miss distance performance is indicated by curves farther to the left and more vertical. The tuned IGC system yields better performance, with an approximately 10% improvement in mean miss distance, 4% improvement in standard deviation, and 13% improvement in 85th-percentile miss.

tuning results in a 10% improvement in mean miss distance. In this case, additional tuning would likely yield additional performance improvements, perhaps with modified weightings on the cost component contributions to the overall cost.

CONCLUDING REMARKS

As modern guided missile systems continue to evolve to counter advances in next-generation threats, the task of optimizing the performance of the associated G&C algorithms becomes more challenging. Traditional manual and ad hoc tuning techniques are increasingly being rendered suboptimal or even unusable. The inherent complexities and challenges associated with optimizing G&C algorithms makes this problem domain well suited to automated, simulation-based optimization techniques in general and to the SPSA algorithm in particular.

Consideration of the broad range of complexities and adjustable variables involved in the problem domain exposes the full difficulty of achieving a viable system design through this method. Inherent noise in cost-function measurements, nonlinear parameter sensitivities, multivariable dependencies, domain constraints, and complex and possibly conflicting tuning objectives all contribute to the challenge of executing a successful automated tuning. Despite these obstacles, research and empirical studies have led to considerable progress in the understanding of automated G&C tuning and the development of systematic tuning methods. In particular, a better understanding of SPSA parameters and their effects as well as cost-function design, parameter scheduling, domain restriction, and initial tuning conditions have been applied successfully to several applications in support of sponsor programs at APL.

Notwithstanding these accomplishments, substantial opportunities for the improvement of simulation-based automated G&C tuning remain. Most significantly, it is likely that the cost functions described herein for G&C applications do not have a single, global minimum, as assumed by the basic form of the SPSA. Methods of extending this basic form to promote global convergence have been developed[12] but have not yet been applied to the G&C problem domain to assess their suitability.

In addition, the creation of hybrid tuning algorithms to more efficiently and thoroughly explore large, complicated cost spaces holds promise for additional advances. A hybrid algorithm would consist of the combination of SPSA with an alternative optimization algorithm potentially more adapted to global searches of the cost space. Such an algorithm would initially use the global optimization method to isolate a probable optimum subspace in the problem domain and then apply SPSA to converge to the optimum tuning within the subspace. Two well-known potential algorithms for global search are genetic algorithms and simulated annealing. Although existing research has applied both types of algorithm individually for purposes of G&C tuning,[4–6] the synthesis of these algorithms with SPSA remains an unexamined option.

REFERENCES

[1]Palumbo, N. F., Reardon, B. E., and Blauwkamp, R. A., "Integrated Guidance and Control for Homing Missiles," *Johns Hopkins APL Tech. Dig.* **25**(2), 121–139 (2004).

[2]Zarchan, P., *Tactical and Strategic Missile Guidance*, American Institute of Aeronautics and Astronautics (1997).

[3]Reardon, B. E., Palumbo, N. F., and Casper, S. G., "Simulation-Based Performance Optimization of Missile Guidance and Control Algorithms," *AIAA/MDA Technology Conf.* (Aug 2002).

[4]Oshman, Y., and Shaviv, I., "Optimal Tuning of a Kalman Filter Using Genetic Algorithms," *AIAA Guidance, Navigation, and Control Conf. and Exhibit*, Denver, CO, AIAA Paper 2000-4558 (Aug 2000).

[5]Powell, T. D., "Automated Tuning of an Extended Kalman Filter Using the Downhill Simplex Algorithm," *AIAA J. Guid. Control Dyn.* **25**(5), 901–908 (Sept–Oct 2002).

[6]Tekinalp, O., and Bingol, M., "Simulated Annealing for Missile Optimization: Developing Method and Formulation Techniques," *AIAA J. Guid. Control Dyn.* **27**(4), 616–626 (July–Aug 2004).

[7]Robbins, H., and Munro, S., "A Stochastic Approximation Method," *Ann. Math. Stat.* **22**, 400–407 (1951).

[8]Kiefer, J., and Wolfowitz, J., "Stochastic Estimation of the Maximum of a Regression Function," *Ann. Math. Stat.* **23**, 462–466 (1952).

[9]Spall, J. C., "Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation," *IEEE Trans. Autom. Control* **37**(3), 332–341 (1992).

[10]Spall, J. C., "Implementation of Simultaneous Perturbation Algorithm for Stochastic Optimization," *IEEE Trans. Aerosp. Electron. Syst.* **34**(3), 817–823 (1998).

[11]Sadegh, P., "Constrained Optimization via Stochastic Approximation with a Simultaneous Perturbation Gradient Approximation," *Automatica* **33**(5), 889–892 (1997).

[12]Maryak, J. L., and Chin, D. C., "Global Random Optimization by Simultaneous Perturbation Stochastic Approximation," *Johns Hopkins APL Tech. Dig.* **25**(2), 91–100 (2004).

The Authors

Brian E. Reardon is a member of the APL Senior Professional Staff in the Air and Missile Defense Department (AMDD). He received B.S. and M.S. degrees in electrical engineering from Drexel University in 2000. He joined the Guidance, Navigation, and Control Group at APL that same year and has been working on missile guidance and control systems and high-fidelity simulation development. **Justin M. Lloyd** is a member of the APL Senior Professional Staff in the Guidance, Navigation, and Control Group of AMDD. He holds a B.S. in mechanical engineering from North Carolina State University and an M.S. in mechanical engineering from Virginia Polytechnic Institute and State University. Currently, Mr. Lloyd is pursuing his Ph.D. in electrical engineering at The Johns Hopkins University. He joined APL in 2004 and conducts work in optimization; advanced missile guidance, navigation, and control; and integrated controller design. **Ron Y. Perel** is a member of the APL Associate Professional Staff in AMDD. He received B.S. degrees in aerospace engineering and mathematics from the University of Maryland College Park in 1997 and an S.M. degree in aerospace engineering from the Massachusetts Institute of Technology in 1999. That same year, he joined APL, where he has been working on missile simulation development as well as missile guidance/control design and analysis. Further information on the material presented in this article can be obtained through Mr. Reardon. His e-mail address is brian.reardon@jhuapl.edu.

Brian E. Reardon

Justin M. Lloyd

Ron Y. Perel

The *Johns Hopkins APL Technical Digest* can be accessed electronically at **www.jhuapl.edu/techdigest**.