https://ntrs.nasa.gov/search.jsp?R=20170012303 2018-07-24T15:28:05+00:00Z

Automation of POST Cases via External Optimizer and "Artificial p2" Calculation

Patrick D. Dees* and Mathew R. Zwack†
Jacobs ESSSA Group, Huntsville, AL, 35806, United States

Diane K. Michelson‡
SAS Institute, Cary, NC, 27513, USA

During conceptual design speed and accuracy are often at odds. Specifically in the realm of launch vehicles, optimizing the ascent trajectory requires a larger pool of analytical power and expertise. Experienced analysts working on familiar vehicles can produce optimal trajectories in a short time frame, however whenever either "experienced" or "familiar" is not applicable the optimization process can become quite lengthy. In order to construct a vehicle agnostic method an established global optimization algorithm is needed. In this work the authors develop an "artificial" error term to map arbitrary control vectors to non-zero error by which a global method can operate. Two global methods are compared alongside Design of Experiments and random sampling and are shown to produce comparable results to analysis done by a human expert.

Nomenclature

ACO	Advanced Concepts Office
$AP2$	Artificial p2
DE	Differential Evolution
EVD	Extreme Value Distribution
PDF	Probability Density Function
$POST$	Program to Optimize Simulated Trajectories
PS	Particle Swarm
$SAP2$	Scaled Artificial p2
UR	Uniform Random

I. Introduction

During conceptual design of complex systems, speed and accuracy are often at odds with one another. Many aspects of the design are fluctuating rapidly due to the interaction of analysis disciplines and competing design paths. Nevertheless, accurate data must be collected from which to down-select designs.[1] Downstream metrics such as reliability, safety, manufacturability, and operations cost are heavily impacted by down-selection during this phase,[2–5] leading to a commitment of up to 80% of the Life Cycle Cost of the system.[2] Thankfully design freedom is high during the conceptual phase, and so pitfalls normally only uncovered past Phase A can be identified and avoided early while they are cheap.[6] However, due to the disciplinary and institutional changes commonly occurring during the conceptual phase, conceptual studies should be on the order of weeks to a month in order to remain relevant.[7,8] Therefore enabling the conceptual designer to produce accurate data in a timely manner is tantamount to program viability.

For launch vehicles in particular, trajectory analysis and optimization is a large hurdle. Tools such as the industry standard Program to Optimize Simulated Trajectories (POST) have traditionally required an expert in the loop for setting up inputs, executing the program, and analyzing the output. While an experienced

*Trajectory Engineer, Advanced Concepts Office, George C. Marshall Space Flight Center/ED04, Huntsville, AL, AIAA Member.

†Systems Engineer, Advanced Concepts Office, George C. Marshall Space Flight Center/ED04, Huntsville, AL, AIAA Member.

‡Principal Analytic Training Consultant, SAS Education

analyst presented with a familiar vehicle can produce optimal performance figures in a timely manner, as soon as the experienced or familiar adjectives are invalid the process can become lengthy. In addition, an experienced analyst working on a similar vehicle may go into the analysis with preconceived ideas about what the vehicles trajectory should look like, which can result in sub-optimal performance reports. Therefore to enable the conceptual designer to produce accurate data in a timely manner we require a vehicle-agnostic, validated, and automated method.

II. Approach

Within NASA Marshall Space Flight Center's Advanced Concepts Office (ACO), the trajectory software in use is the Program to Optimize Simulated Trajectories (POST).[9] It provides the dual analyses of direct shooting trajectory analysis and constrained local optimization. During a run, POST works with user specified independent and dependent variables to optimize a user specified variable's value. The selected optimizer first propagates the vehicle defined within an input file and initial conditions for its flight through a series of phases or events then seeks to improve the trajectory. At first this is done by minimizing the difference between the currently propagated trajectory and the desired one via dependent variable setup and a sum squared error value represented by the variable p2. If p2 enters the feasible region ($p2 \leq 1$) then focus shifts to optimizing the desired variable. The p2 space is in general non-linear and multi-modal[10,11] and POST's current optimization routines are local. Therefore whichever optima is closest to the initial values of the independent variables will be the one reported assuming the optimizer is given sufficient iterations to resolve it. Due to the multi-modal response, a reported optima has only a chance of being the global optimum. From observation the authors have found feasible regions to be disjoint, housing one or more optima, each with a characteristic spread of the optimized variable.

To give more accurate data for any vehicle under analysis we desire finding the global optimum from among the collection of local optima. In order to gain confidence on capturing the global optimum a global optimization algorithm must be applied. However, the direct shooting method by which POST propagates trajectories puts up a roadblock here. Within the POST input file are events acting as gates through which the trajectory must pass to perform the desired mission. Each phase is triggered by one or several variable values satisfying equality on inequality constraints. For example these events can include launch tower clearance at a certain height, dropping stages when their propellant is exhausted, and inserting into orbit via specified orbital parameters. POST allows these events to be primary (required) or secondary (optional). If the initial values of the independent variables are such that the initial trajectory POST propagates does not pass through each primary event, then POST immediately returns with a p2 value of exactly zero and an output message of "Unusable Nominal Trajectory". When analyzing a new vehicle or a set of vehicles the independent variable space will tend to be large in order to capture all potential modes of ascent. Therefore a large proportion of the corresponding p2 space will be identically zero in regards to values of the independent variables, severely hindering the use of a global optimization method without *a priori* knowledge of the non-zero region. If a global method was used on this space as is, the unusable region will be the clear winner as the error of usable trajectories only asymptotically approaches zero rather than achieving it exactly. If done by a transform, recoding the unusable space value of exactly zero to a large number ($\sim 10^{15}$), then only those members of an initial population who just so happen to start in the usable region will be useful. With the observation that the usable region is small in regards to the full investigation space,[12] we conclude that a significant amount of time would be wasted bringing all those points in the unusable space to where they are useful.

In previous work[13] a brute force version of global optimization was employed via uniform random sampling of the independent variables. In this work we present an "artificial p2" (AP2) calculation which assigns a non-zero value to any arbitrary set of independent variables, enabling the use of global optimizers with POST.

A. Artificial p2

The p2 calculation within POST is a sum squared error using the current values of the specified dependent variables at the desired part of the trajectory.[14] When an unusable independent variable or *control vector* is input, not all desired events are present for calculation of p2 in POST's normal manner, and the program exits returning a p2 value of exactly zero. The AP2 calculation simply takes the information which *is* available

in the propagated trajectory and calculates the sum squared error in the same manner. For example, if the dependent variables are associated with phases one, two, and three, however the input control vector only produces a trajectory which propagates through phases one and two, the AP2 calculation simply uses the information in phase two to stand in for the missing phase three data. For example, if phase three is defined as orbital insertion and constrained by orbital elements apogee and perigee, the AP2 calculation would simply use the values for apogee and perigee in phase two as stand in values.

Figure 1 below shows the effect of applying the AP2 calculation. Along the x-axis are the components of the control vector $\vec{u}$ for an example vehicle. Each point represents a single run of POST. In red are usable runs, those whose initial $\vec{u}$ resulted in a trajectory which propagated through each of the required phases. In blue are those which returned as unusable and for whom the AP2 calculation was applied. At first glance it is of note that many of the control vector components have little to no relationship with p2. The first component has the most relationship with p2, stretching outward from the usable region to remain flat for most of the component's range.

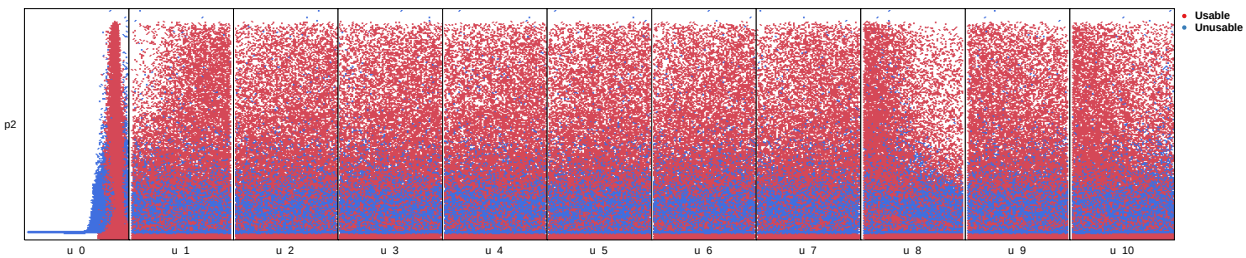


Figure 1: p2 versus control vector with artificial p2 calculation

As was discussed previously, large flat areas are not of much use in finding the usable region. To combat this problem, a modification is made to the AP2 calculation to scale in relation to the expected burn time of the vehicle.

B. Scaled Artificial p2

Knowing the initial propellant loads, thrusts, and specific impulses of the stages and engines making up the vehicle gives an "ideal" burn time which can be used to induce a gradient in the AP2 calculation. In the event that $\vec{u}$ crashes the vehicle or causes it to shoot farther up than desired, the burn time will be less than expected. The AP2 calculation is then modified as

$$scaled\ artificial\ p2 = \frac{ideal\ burn\ time}{burn\ time} * artificial\ p2$$

to allow these effects to introduce curvature in the value. Sampling the same example vehicle then produces the data in Figure 2 below.

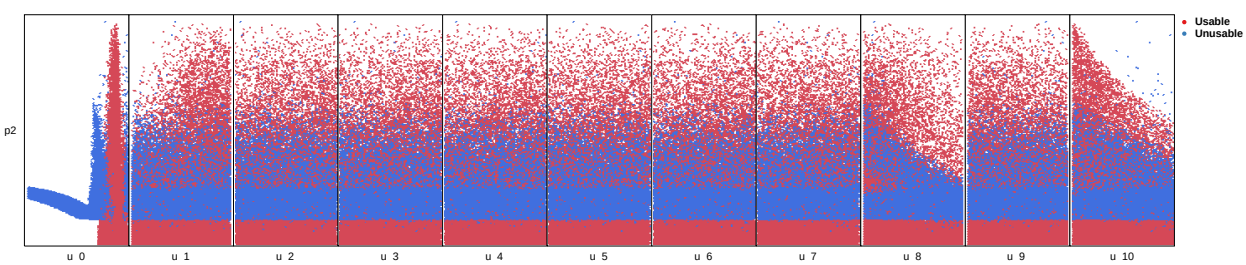


Figure 2: p2 versus control vector with scaled artificial p2

As with AP2, the scaled artificial p2 (SAP2) calculation has several components which do not have a noticeable effect on the value. However, several more do have an effect, producing a gradient which leads to the usable region. While global methods do not in general use gradient information, many do use information from previous iterations. For example, PS utilizes an individual particle's best position and the overall swarm's best position to update the individual particle's velocity. By introducing the slope toward

the usable region, the time spent evaluating points in the unusable space will be reduced by drawing global optimization method members toward the usable space.

C. Comparison

To compare calculations, the unusable runs are collected and colored by calculation type. It is easily seen in Figure 3 below that in general SAP2 is of higher magnitude, and more spread out in the unusable region far from the usable region. The higher magnitude is due to the SAP2 calculation using the burn time of the vehicle. The shorter the burn time (further away from the usable space) the larger the ratio of ideal to observed burn time and therefore a larger SAP2 value. The spread and gradient are due to the fact that a vehicle can only crash so hard. For example, if a vehicle's orbital insertion is constrained by apogee and perigee in phase three but in phase two the vehicle crashes there is a hard minimum for the apogee and perigee numbers used in the AP2 calculation. However for SAP2 the burn time is affected by each of the components of $\vec{u}$ and so these differences manifest in a spread. The gradient shows that this component in particular has a large influence on the overall burn time.

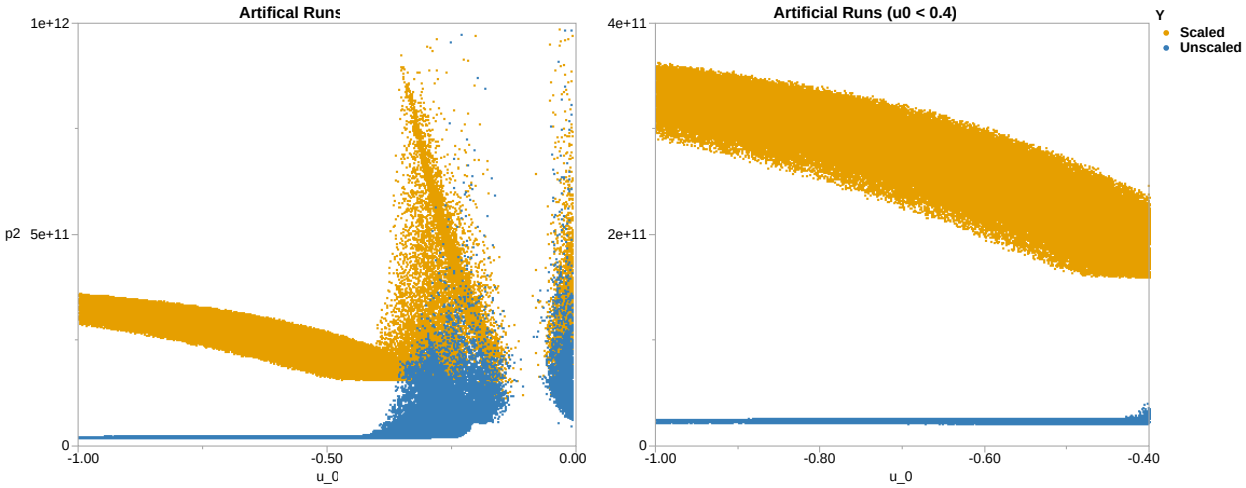


Figure 3: Comparison of scaled and unscaled artificial p2

III. Comparison Trials

In this section we present a series of trials designed to test the application of several global optimization methods utilizing the SAP2 calculation to optimizing the ascent trajectory of a launch vehicle using POST. Three global optimization methods are selected: Uniform Random sampling (UR), Differential Evolution (DE),[15] and Particle Swarm (PS).[16] These are selected due to the author's familiarity and their availability through the scipy.optimize[17] and pyswarm[18] libraries. As each method in testing incorporates stochastic steps in their procedure, each trial is repeated ten times to get a fair statistical spread. All data reported in Results will then be based upon these repeated trials.

The multiPOST software developed by the authors, presented in [13] and extended in [12, 19, 20] is used for all trials and limited to 80 simultaneous threads running across several Windows 7 machines.

A. Trial Levels

To asses the algorithms' abilities a full factorial experimental design was run. Trials are conducted setting the iterations each algorithm has to work with to $[5, 10, 15]$. These will be referred to as *global iterations*. To assess the role of POST's internal optimizer (if any) in tandem with the global optimizer, trials will also be run setting POST's iterations to $[5, 10, 15]$. These will be referred to as *local iterations*. A general guideline to initial population size[21] and Design of Experiments (DOE) sampling[22] is to use $10x$ the number of dimensions. The initial population size of DE & PS methods are then set accordingly. The total number

of control vectors evaluated for these global methods will then be *Global Iterations* $* N$. UR trials are set to 1000 control vectors evaluated as a comparison.

Table 1: Comparison Trials Setup

Method	**Global Iterations**	**Local Iterations**	**N**
Differential Evolution	5, 10, 15	5, 10, 15	120
Particle Swarm	5, 10, 15	5, 10, 15	120
Uniform Sampling	N/A	5, 10, 15	1000

B. Trajectory Description

The trajectory used is a 2.5 stage to orbit cargo vehicle the authors have previous experience with, and use the previously analyzed and "manually" run trajectory as the point of comparison and stand-in for global optimality. The vehicle has two solid rocket boosters attached running simultaneously with a liquid engine Core stage. In-line with the Core stage is a liquid engine Upper Stage. The vehicle mission is to ascend to Low Earth Orbit, jettisoning the cargo shroud whenever a free molecular heating rate limit is satisfied, remain in orbit for two revolutions, then depart for a Trans-Lunar Injection burn. During ascent are a gravity turn, booster staging, Core staging, cargo shroud jettison, and start up of the Upper Stage. At mission end the Upper Stage is required to hold back a percentage of its total ΔV as propellant for a Flight Performance Reserve. Dependent variables take the form of maximum dynamic pressure and axial acceleration constraints, orbital insertion constrained to a particular circular orbit, and Trans-Lunar Injection modeled as a specific value of specific orbital energy C3. Independent variables are nine pitch rates spread across the ascent, payload mass, launch azimuth, and upper stage propellant load for a total of 12. Finally, the optimized value undergoing maximization is the payload mass delivered.

IV. Results

For each setting of algorithm type, number of global iterations, and number of local iterations, we are interested in the trends in, distribution of and models for several responses: time required, number of results returned, and payload delivered. Trends are displayed in the form of variability gauge charts containing box plots. Each box plot consists of a central rectangle whose upper and lower ends represent the third and first quantiles respectively, showing the inner quartile range. The horizontal line within the central box shows the median. Vertical lines extending from the central box show the maximum and minimum of the data. Distribution fitting is performed using JMP Software from SAS[23] utilizing maximum likelihood estimation, corrected Akaike Information Criterion,[24] and visual inspection of Probability Density Function (PDF). Several effects of interest are found to follow the Johnson Sl distribution. Time-to-event data is often modeled using a lognormal distribution,[25] which is a special case of a system of distributions known as the Johnson system. The Johnson system is able to fit any combination of location, scale, skewness, and kurtosis, for data that are unbounded, interval-bounded, or bounded on one side (Johnson Sl). The probability density function (PDF) of the Johnson Sl is defined as

$$f(x) = \frac{\delta}{|x-\theta|}\psi[\gamma + \delta log(\frac{x-\theta}{\sigma})]$$

where ψ is the standard normal pdf and $x > \theta$ if $\sigma = 1$, or $x < \theta$ if $\sigma = -1$. The parameters θ, σ, γ and δ are estimated from data using maximum likelihood estimation.

For each metric models are created to help elucidate the performance of each method as they pertain to the levels of each trial.

Model interpretation can be found by examining Table 2. The full model is a full quadratic model in the number of global and local iterations. That is,

$$E[Y] = \beta_0 + \beta_1 * (Global\ Iterations) + \beta_2 * (Local\ Iterations)$$

$$+\beta_{12} * (Global\ Iterations) * (Local\ Iterations) + \beta_{11} * (Global\ Iterations)^2 + \beta_{22} * (Local\ Iterations)^2$$

where E[Y] represents the expectation or average value of Y. R^2 represents the proportion of variation in Y explained by the model. The root mean squared error (RMSE) represents the unexplained variation. For example, for the Differential Evolution Total Time response, 98% of the variation in Total Time is explained by the model, and the unexplained variation is 107.5 minutes.

The estimates of the β parameters are found using ordinary least squares regression.[26] The interpretation of the intercept is the average value of the response when the predictors are at their zero level. In our experiment, the intercept has no physical interpretation and will be ignored. The interpretation of the slopes of the main effect terms is the average change in the response when increasing the predictor by one unit. For example, increasing the number of global iterations for the Differential Evolution algorithm by 1 leads to an increase of 128.8 seconds of Total Time on the average. However, the variation in the response is also affected by the quadratic and interaction effects. They represent combinations of effects. For example, the change in Total Time due to changing the number of local iterations depends on the number of global iterations. The estimate of the interaction effect is 14.2.

The estimates are calculated using data from the experimental runs. Another set of experimental runs would give different, but similar results. The variation in the estimates due to sampling is captured in the standard error. The t Ratio uses the estimate and its standard error to test the hypothesis that the true effect is actually zero, that is, the term is not significant in explaining variation in the response. A high t Ratio gives evidence for the significance of the term. High t Ratios correspond to low p-values. The p-value ($Prob > |t|$) is the probability of rejecting the hypothesis of zero effect when it is really true. We used a cutoff level of significance of 0.05 to determine significance. All estimates with $p < 0.05$ have been removed in Table 2.

Trial summary data is presented in Table 3, displaying levels and high level statistics on the Time Required, N Returned, and Payload Delivered metrics to be discussed in detail in the following subsections.

		Term	Estimate	Std Error	t Ratio	Prob > |t|	Summary of Fit	
DE	Total Time	Intercept	-1388.1000	43.8886	-31.63	<.0001	RSquare	0.9801
		Global Iterations	128.8133	2.7758	46.41	<.0001	RSquare Adj	0.9791
		Local Iterations	110.1500	2.7758	39.68	<.0001	Root Mean Square Error	107.5047
		(Global Iterations-10)*(Global Iterations-10)	2.9613	0.9616	3.08	0.0028	Mean of Response	1050.8890
		(Global Iterations-10)*(Local Iterations-10)	14.2440	0.6799	20.95	<.0001	Number of Observations	90
	N Results	Intercept	-522.6333	38.3193	-13.64	<.0001	RSquare	0.9109
		Global Iterations	66.6700	2.4235	27.51	<.0001	RSquare Adj	0.9067
		Local Iterations	19.7900	2.4235	8.17	<.0001	Root Mean Square Error	93.8626
		(Global Iterations-10)*(Global Iterations-10)	2.4687	0.8395	2.94	0.0042	Mean of Response	383.111
		(Global Iterations-10)*(Local Iterations-10)	3.6030	0.5936	6.07	<.0001	Number of Observations	90
	Difference from Global	Intercept	1241.9755	92.5998	13.41	<.0001	RSquare	0.6632
		Global Iterations	-36.8940	5.8565	-6.3	<.0001	RSquare Adj	0.6474
		Local Iterations	-61.0769	5.8565	-10.43	<.0001	Root Mean Square Error	226.8222
		(Global Iterations-10)*(Local Iterations-10)	5.3313	1.4345	3.72	0.0004	Mean of Response	338.7909
		(Local Iterations-10)*(Local Iterations-10)	4.5915	2.0288	2.26	0.0262	Number of Observations	90
PS	Total Time	Intercept	-1575.6220	51.8370	-30.4	<.0001	RSquare	0.9800
		Global Iterations	159.3037	3.4825	45.74	<.0001	RSquare Adj	0.9793
		Local Iterations	142.7841	3.4857	40.96	<.0001	Root Mean Square Error	131.4472
		(Global Iterations-10.0581)*(Local Iterations-10.1744)	13.9804	0.8537	16.38	<.0001	Mean of Response	1479.2790
							Number of Observations	86
	N Results	Intercept	-747.9409	67.7664	-11.04	<.0001	RSquare	0.9553
		Global Iterations	162.8334	3.8953	41.8	<.0001	RSquare Adj	0.9531
		Local Iterations	22.5150	3.9116	5.76	<.0001	Root Mean Square Error	149.5093
		(Global Iterations-10.0581)*(Global Iterations-10.0581)	4.6688	1.3408	3.48	0.0008	Mean of Response	1152.6860
		(Local Iterations-10.1744)*(Local Iterations-10.1744)	-2.6359	1.3416	-1.96	0.0529	Number of Observations	86
	Difference from Global	Intercept	431.1059	45.6127	9.45	<.0001	RSquare	0.6077
		Global Iterations	-10.2491	2.7869	-3.68	0.0004	RSquare Adj	0.5883
		Local Iterations	-25.4950	2.8001	-9.11	<.0001	Root Mean Square Error	105.1726
		(Global Iterations-10.0581)*(Local Iterations-10.1744)	1.5566	0.6831	2.28	0.0253	Mean of Response	130.9853
		(Local Iterations-10.1744)*(Local Iterations-10.1744)	3.7716	0.9602	3.93	0.0002	Number of Observations	86
UR	Total Time	Intercept	0.2378	0.1206	1.97	0.0586	RSquare	0.5998
		Local Iterations	0.0723	0.0112	6.48	<.0001	RSquare Adj	0.5855
							Root Mean Square Error	0.2497
							Mean of Response	0.9611
							Number of Observations	30
	N Results	Intercept	18.9000	3.0031	6.29	<.0001	RSquare	0.5209
		Local Iterations	1.2100	0.2452	4.93	<.0001	RSquare Adj	0.4854
		(Local Iterations-10)*(Local Iterations-10)	-0.1900	0.0849	-2.24	0.0338	Root Mean Square Error	5.4830
							Mean of Response	27.833
							Number of Observations	30
	Difference from Global	Intercept	1953.0177	218.8696	8.92	<.0001	RSquare	0.7078
		Local Iterations	-138.4769	17.8706	-7.75	<.0001	RSquare Adj	0.6862
		(Local Iterations-10)*(Local Iterations-10)	14.3340	6.1906	2.32	0.0284	Root Mean Square Error	399.5994
							Mean of Response	807.1476
							Number of Observations	30

Table 2: Data from trials

	Global	Local	N Evaluated	Trial Data Used	Time (minutes) μ	Time (minutes) σ	N Results μ	N Results σ	Difference from Global (lbm) μ	Difference from Global (lbm) σ
Differential Evolution	5	5	600	10	217	10	48	11	1036	360
	5	10	600	10	432	8	76	19	484	347
	5	15	600	10	646	21	87	29	131	74
	10	5	1200	10	495	60	259	98	606	392
	10	10	1200	10	992	102	352	109	196	124
	10	15	1200	10	1518	148	415	91	52	11
	15	5	1800	10	804	73	529	104	405	199
	15	10	1800	10	1699	156	753	148	107	56
	15	15	1800	10	2656	193	928	132	33	7
Particle Swarm	5	5	600	9	315	33	255	73	392	175
	5	10	600	9	664	91	399	84	141	67
	5	15	600	9	1014	145	448	87	51	26
	10	5	1200	9	716	59	927	120	307	197
	10	10	1200	10	1422	142	1110	144	48	19
	10	15	1200	10	2179	213	1147	147	27	3
	15	5	1800	9	1163	109	1819	250	209	188
	15	10	1800	10	2272	140	2039	219	34	8
	15	15	1800	10	3262	157	2108	120	24	2
Uniform Random		5	1000	10	0.6	0.2	20	4	1619	575
		10	1000	10	1.0	0.2	31	7	568	381
		15	1000	10	1.3	0.3	32	5	234	60

Table 3: Data from trials

A. Time Required

UR is far and away the quickest method as can be marginally seen in Figure 4, the longest trial requiring just under 1.5 minutes. The reason why this method can evaluate 1,000 runs in such a short time is due to the small usable space this vehicle exhibits. Across each trial, the number of usable $\vec{u}$ randomly averages to about 6.7%. Figure 6 shows the distribution of seconds required to evaluate an unusable run, which follows a Johnson Sl distribution. The mean found is 0.61s, with standard deviation of 0.38s. For a 1,000 sample UR run with a 6.7% usable rate, 80 available simultaneous threads and the time spread, evaluating the unusable runs will take from 3.5-10.5s. The remaining time is then dominated by the relationship between the time required for POST to perform an iteration and the number of local iterations allowed. This is shown in Figure 5, which depicts a roughly linear trend with local iterations. This behavior is backed up by the model selected which found only a first-order term with local iterations to be significant. The coefficient estimate for the relationship with local iterations is at 0.0723 minutes or 4.34 seconds. At the 5 local iteration level this puts the average usable run at 21.69s which falls in line with the chart. From this data we then have a measure of roughly 0.61s to evaluate an unusable run or rather a run without iterations, and 4.34s per local iteration to evaluate a usable run.

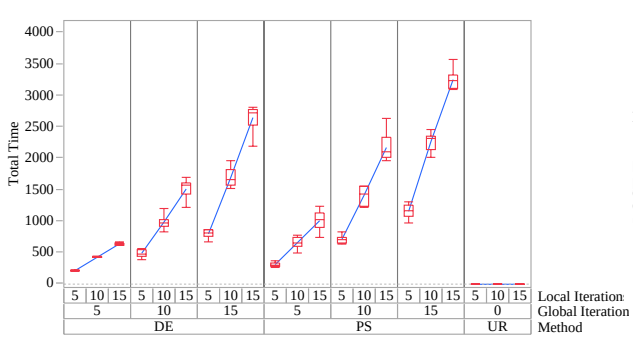


Figure 4: Total time required

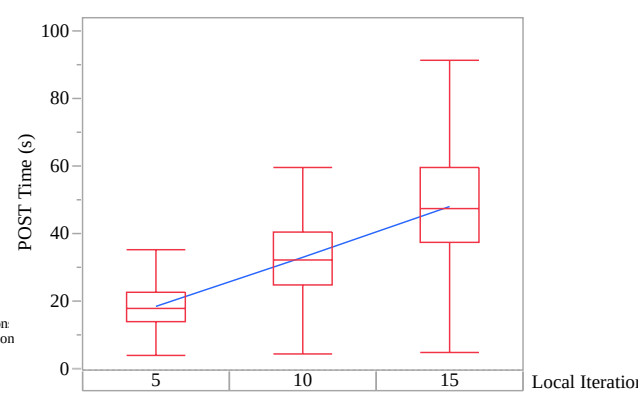


Figure 5: Time required to evaluate usable run

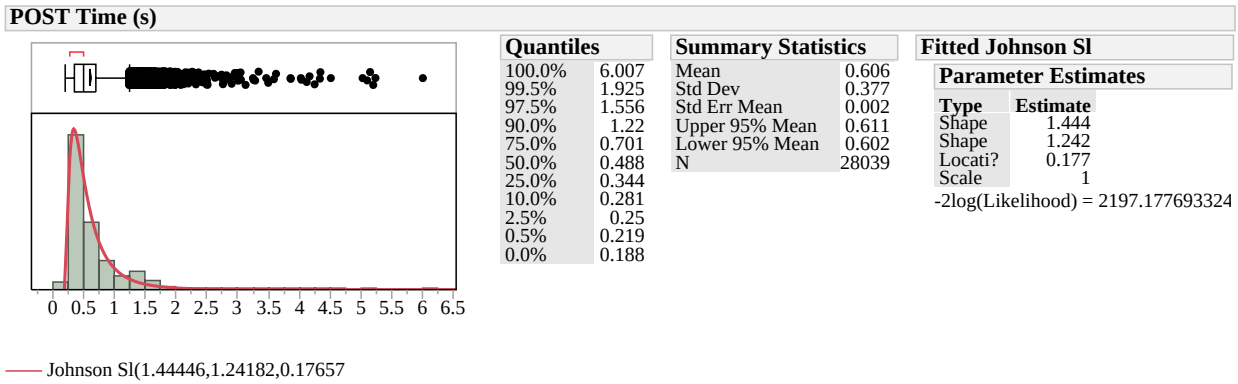


Figure 6: Time required to evaluate usable run

Both DE & PS exhibit relationships with global and local iterations. In each case the method is more sensitive to global iterations, which makes sense as more global iterations means more runs in general. The relationship with local iterations comes very close however, which also makes sense as the driving factor in the total time is how long POSTs local optimizer takes to run local iterations. Both exhibit a significant positive interaction term. This makes sense, as the number of local iterations linearly increases the time required per usable run, and the higher the global iterations the higher the percentage of the population that is within the usable region. The vertex at ten iterations tells us that this behavior should be expected to further increase quadratically outside the levels of this study. DE also exhibits a quadratic relationship with global iterations, leading us to expect that increasing global iterations beyond 15 may lead to a situation where DE takes longer PS.

B. Number of Results

The next metric of interest is the number of feasible data points returned by the methods, shown in Figure 7. As with time required, increasing either the global or local iterations increases the number of converged points returned. UR & PS share a negative quadratic relationship with local iterations. This shows us that there is a point of diminishing returns with the number of local iterations used, the vertex of which is at ten local iterations. This can be seen dramatically for the UR trials in Figure 8.

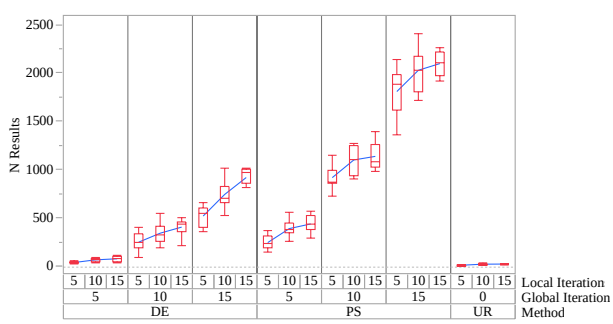


Figure 7: Number of converged trajectories reported

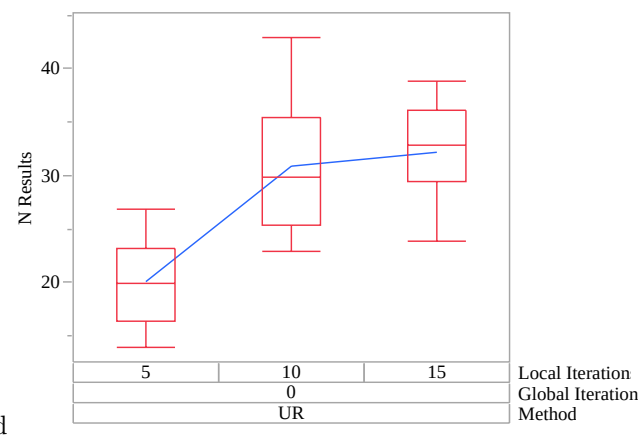


Figure 8: Number of converged trajectories reported

Unlike Time Required, N Returned is much more sensitive to the number of global iterations for DE & PS. This shows that the global methods are doing a good job of finding the feasible region within the usable region, as in general the number of local iterations used is not sufficient to resolve a minima. If the situation were reversed it would be telling us that the local optimization step was doing more to bring the population into the feasible region. Both DE & PS have a positive quadratic relationship with global iterations, showing that the maximum efficacy of global iterations has not been captured by these trials. If there is a point of

diminishing returns for global iterations it is beyond 15. Additionally DE has an interaction term, showing that it has a stronger relationship with the local optimization step. This makes sense, as the Time Required metric showed DE taking less time to complete and it returns less data meaning that it is slower to reach the feasible region. It is therefore utilizing the local optimization step more directly to reach optima.

C. Payload Delivered

The final metric of interest is presented as the difference from the global optima resulting from the authors previous analysis. Performance across trials is shown in Figure 9 below. UR at low local iterations has a lot of variability and tends to give a poorer answer than one given by either PS or DE. As the number of local iterations increases however, the answer quickly improves. At its highest local iterations level UR is on par with mid-level performance of either DE or PS. For each model there is a negative linear relationship and a positive quadratic relationship with local iterations centered at ten local iterations. This behavior mirrors that seen in the N Results metric: that there is a point of diminishing returns in performance past ten local iterations for all of the methods. All methods exhibit a stronger relationship with local iterations than other terms (excluding the intercept), which makes sense as the local iterations in these trials fine-tune the initial conditions selected by either UR sampling or the populations, iteration, and method of the global algorithms. In addition both DE & PS exhibit an interaction term, showing that there is a relationship between the global and local optimization steps. This makes sense, as many optimization schemes will use a final local optimization step to fine-tune the answer.

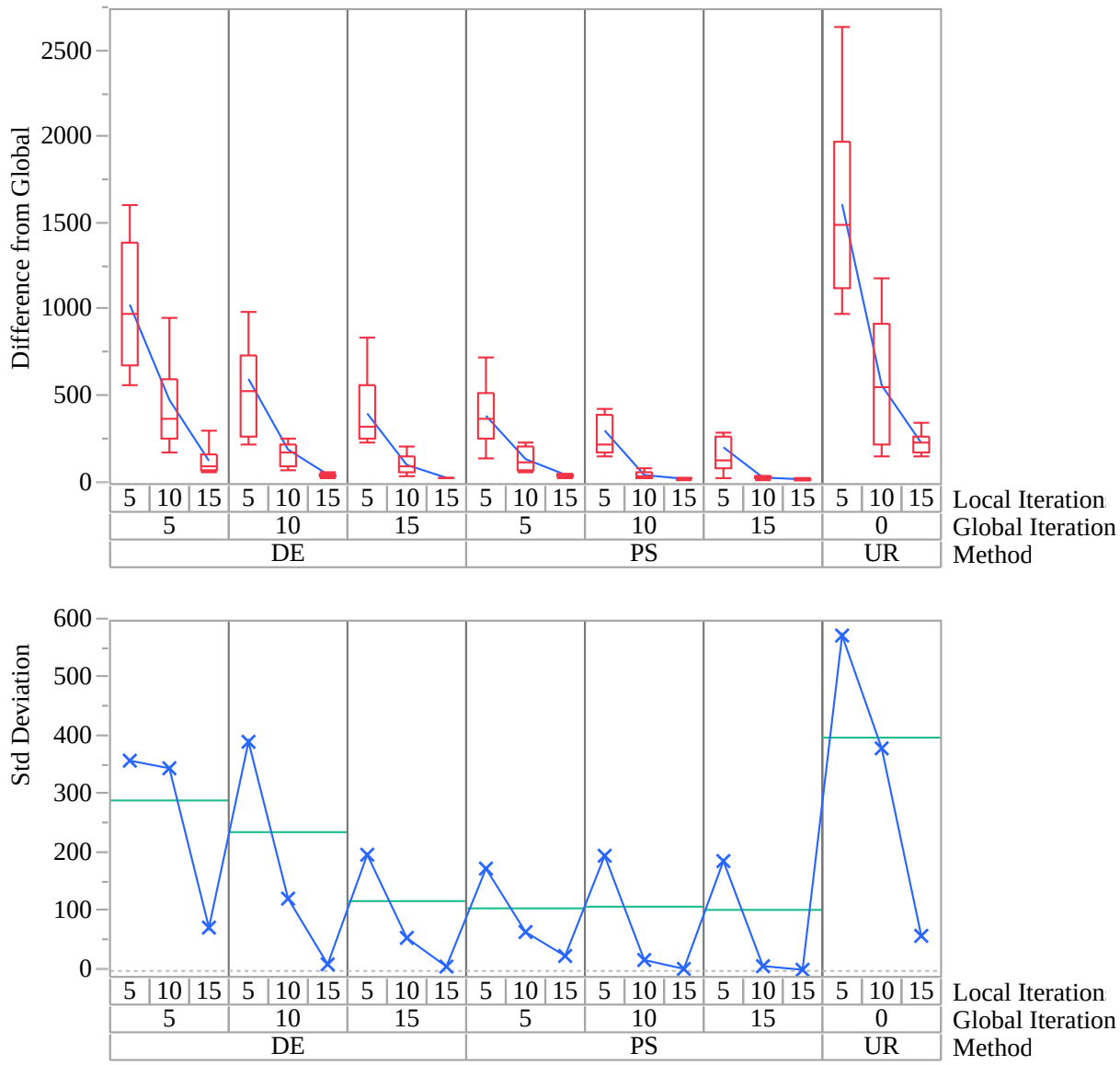


Figure 9: Difference from global optima

PS shows two interesting behaviors, better shown also with Figure 9 in the bottom portion depicting the variation in standard deviation for the Difference from Global data. The green line shows the mean of the standard deviations in each grouping of global iterations. While DE shows a decrease in standard deviation as global iterations, PS is virtually level. This suggests that there is little improvement to be gained in increasing the global iterations past five for PS. In addition it can be seen that there is little improvement in increasing the number of local iterations past ten, an observation noted in the previously presented metrics. Finally, UR shows the largest relationship between local iterations and standard deviation. This behavior once again highlights the effect of a small usable region, and even smaller feasible region. At low local iterations, a $\vec{u}$ has to be very near the feasible region to have any chance of passing within, effectively reducing the space which will produce "usable" information. As local iterations increase, the samples have to be less lucky as the effective usable region grows larger with each available local iteration.

V. Conclusion

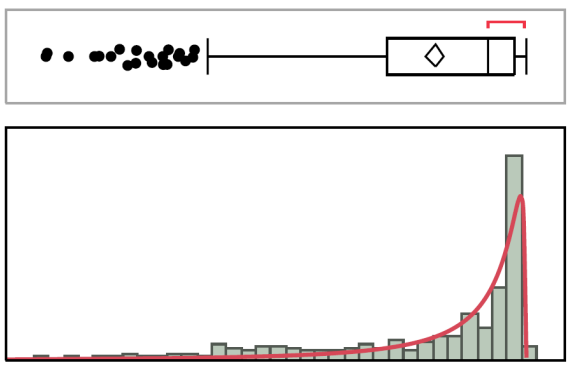
In light of the trial data we come to the conclusion that PS is more effective at exploring this optimization space. The characteristically longer time PS takes to execute is driven by its population moving into the usable region more quickly and thus each global iteration takes longer. It returns more data, once again because the swarm converges to the usable region more quickly. The diminishing returns seen in regard to local iterations and N Returned also shows that the global method brings its population further clumped into the feasible region within the usable region effectively. PS works off of a velocity update function[16] which can be strongly affected by large differences in magnitude in a particle's current position fitness value and the global best. In the unusable and usable infeasible regions p2 can range from 10^{12} to 1 and so particles close to the feasible region exercise a strong pull on those in the unusable region. By comparison, DE simply chooses the most fit from its population and produces new members via crossing and mutation, resulting in a slower crawl toward the usable and feasible regions. Because PS gets to the feasible region more quickly it spends more time resolving feasible minima and therefore has a higher chance of finding the global optimum. Diminishing returns in N Results and Payload Delivered were found in local iterations past ten, however improvement is found beyond ten global iterations and so our recommendation is to use PS at ten local iterations and as high of global iterations as time allows.

A question we intend to follow up on is the effect of computational resources on the time required for the methods. For UR it is expected that increased resources would bring the Time Required metric to a spread defined by the average time POST requires for iterating on usable runs. This would likely manifest as a reduction of the intercept in the UR trials Total Time model to one more consistent with the spread of time required to evaluate a single unusable run as shown in Figure 6 in the previous section. With more resources the number of runs evaluated by UR could increase as high as time available allows. This would likely increase the N Returned as the percentage of usable runs would stay constant, however looking across UR trials and at the gross data returned by all its trials, it is not expected that increased resources would help UR in the Payload Delivered metric. Individually versus as a group there is not much difference in Payload Delivered. For PS & DE, Time Required performance would increase up to the point where the number of threads available was equal to the population size. At this point there would be more threads than cases to run at a time. Past this point the population size could be increased beyond the minimum 10*number parameters used in this study which would retain the same time required spread but is expected to improve both the N Returned and Payload Delivered metrics. With all methods it is expected that decreased resources would cause time required to increase as 1/x, as each individual execution has to wait in a longer and longer queue.

The goal of this work has been to demonstrate the use of the scaled artificial p2 calculation in conjunction with global optimization methods applied to the problem of ascent trajectory optimization using the direct shooting method and local optimizers within POST. Using a global method with the SAP2 calculation is useful whenever a vehicle under consideration is not familiar, or the analyst is not experienced. By moving along the curves within the unusable space mapped by SAP2 a global method can suss out whether a vehicle has a feasible space at all. This kind of analysis could be done with fewer local iterations, greatly reducing the amount of time required. In this case however the number of global iterations should be increased to make up for the loss of help from the local optimizers. In this manner verification and validation utilizing the data returned can be used for debugging a POST input deck when the vehicle design is known to be feasible or determining whether a conceptual vehicle is able to perform its prescribed mission.

In the end, each of UR and the pair of PS & DE are found to have their place in conceptual design investigation of Earth-to-orbit trajectory performance. In the case where the vehicles under analysis are chosen by DOE to take part in a large trade-space study such as the applications in [13, 19], then UR is the way to go. If the vehicles under analysis are to be used in the creation of a surrogate model describing the entirety of the design space then the off-optimality of any one vehicle reported will be smoothed out by the end model. This is handy, as the variability in the answer delivered by UR is in general higher than that of the other global methods at the same number of local iterations. UR works very fast, discards potential dead ends quickly, provides an answer close to the other methods tested, and is well suited for looking at multiple vehicles simultaneously. A fair amount of calories are burned in evaluating unusable cases however, and the UR method is improved significantly by utilizing *a priori* knowledge of the feasible region. The reader is referred to [12] for more information on this development. If however the point of analysis is to provide more detail on a specific vehicle concept, then using an overall global optimization scheme with the SPA2 calculation is a better bet.

As the global optimization algorithms progress through their global iterations, each method's procedure brings its population further toward and into the usable region. Therefore as iterations proceed the average length of time for an iteration will increase as more members evaluate usable trajectories. This is the main mechanism for the high execution times of the global methods. Due to the global methods' behavior of iteratively increasing their populations' usable pass rates they also tend to return a lot more information. These points returned are not in general allowed enough local iterations to resolve a minima but do return data from within the feasible space. Instead of returning a single data point, the global methods allow the automated return of a myriad of data within the feasible region of a vehicle's response space, returning a distribution describing the potential performance of the vehicle. With this distribution in hand, the conceptual designer can now describe both optimal and off-optimal performance. The distribution of data which comes back from across the feasible space of the analyzed vehicle is then not simply a collection of local optima as is the case of UR, but also the in-between points which can represent the limitations of non-idealized control systems, manufacturing inefficiencies, off-spec engine components, and other concerns which only come to light later in the design cycle. By having this additional data there is higher confidence in the distribution fit via tighter bounds on mean, standard deviation, and all model fit parameters. This can be seen comparing the distributions in Figures 10 and 11 below. The amount of data returned by PS brings the confidence intervals on the Johnson Sl model fit parameters much tighter and from visual inspection does seem to model the data far more closely.



Summary Statistics

Std Dev	3655
Std Err Mean	165.3
N	489

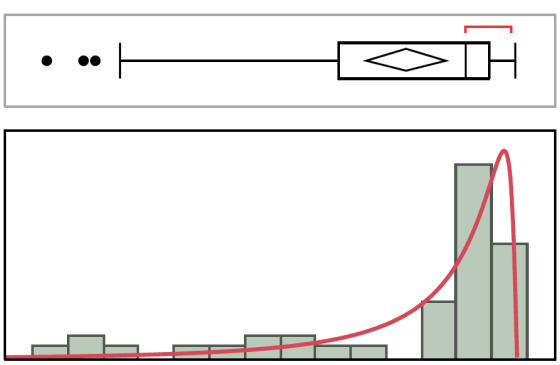
Fitted Johnson Sl

Parameter Estimates

Type	Estimate	Lower 95%	Upper 95%
Shape	-5.23	-5.59	-4.87
Shape	0.725	0.679	0.774
Scale	-1		

-2log(Likelihood) = 8764.50

Figure 10: Johnson Sl fit to Particle Swarm trial with 5 Global, 10 local iterations



Summary Statistics

Std Dev	3710
Std Err Mean	559
N	44

Fitted Johnson Sl

Parameter Estimates

Type	Estimate	Lower 95%	Upper 95%
Shape	-6.1	-7.98	-4.21
Shape	0.824	0.618	1.097
Scale	-1		

-2log(Likelihood) = 793.31

Figure 11: Johnson Sl fit to Uniform Random trial with 15 local iterations

Keen-eyed readers may spy an issue with the above distribution fits. As mentioned earlier in this document the authors have empirically noted the presence of multiple disjoint feasible regions within the response space of several vehicles. For this particular vehicle there seem to be three resolved by the trials. We do not present any method for handling the use of mixture models at this time, but we do hope to address this issue in a later publication.

References

[1]NASA, *NASA Systems Engineering Handbook*, National Aeronautics and Space Administration, December 2007, NASA/SP-2007-6105 Rev 1.

[2]Blair, J., Ryan, R., Schutzenhofer, L., and Humphries, W., "Launch Vehicle Design Process: Characterization, Technical Integration, and Lessons Learned," Tech. rep., National Aeronautics and Space Administration, May 2001.

[3]Dulac, N. and Leveson, N., "Incorporating safety risk in early system architecture trade studies," *Journal of Spacecraft and Rockets*, Vol. 46, No. 2, 2009, pp. 430 – 437.

[4]Ullah, R., Zhou, D.-Q., Zhou, P., Hussain, M., and Amjad Sohail, M., "An approach for space launch vehicle conceptual design and multi-attribute evaluation," *Aerospace Science and Technology*, 2011.

[5]Zwack, M., *A Conceptual Reliability Growth Approach for Comparison of Launch Vehicle Architectures*, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, December 2014.

[6]Wertz, J. and Larson, W., *Reducing Space Mission Cost*, Microcosm Press, 1996.

[7]Raymer, D., *Aircraft Design: A Conceptual Approach*, American Institute of Aeronautics and Astronautics, Inc., 4th ed., 2006.

[8]Mulqueen, J., Maples, D., and Fabisinski, L., "Tailoring Systems Engineering Processes in a Conceptual Design Environment: A Case study at NASA Marshall Spaceflight Center's ACO," *INCOSE International Symposium*, Rome, Italy, July 2012.

[9]Waters, E., Garcia, J., Beers, B., Philips, A., Holt, J., and Threet, G., "NASA Advanced Concepts Office, Earth-To-Orbit Team Design Process and Tools," *AIAA SPACE 2013 Conference & Exposition*, 2013.

[10]Nelson, D., *Qualitative and Quantitative Assessment of Optimal Trajectories by Implicit Simulation (OTIS) and Program to Optimize Simulated Trajectories (POST)*, Master's thesis, Georgia Institute of Technology, April 2001.

[11]Steffens, M., Edwards, S., and Mavris, D., "Capturing the Global Feasible Design Space for Launch Vehicle Ascent Trajectories," *AIAA SciTech*, Kissimmee, FL, January 2015.

[12]Zwack, M. R. and Dees, P. D., "Improvement of Automated POST Case Success Rate using Support Vector Machines," *AIAA Space Conference and Exhibition*, 2017.

[13]Dees, P., Zwack, M., Steffens, M., Edwards, S., Diaz, M., and Holt, J., "An Expert System-Driven Method for Parametric Trajectory Optimization During Conceptual Design," *AIAA SPACE Conference and Exposition*, Pasadena, CA, August 2015.

[14]Powell, R., Striepe, S., Desai, P., Braun, R., Brauer, G., Cornick, D., Olson, D., Peterson, F., and Stevenson, R., *Program To Optimize Simulated Trajectories (POST) Vol II: Utilization Manual*, 1997.

[15]Storn, R. and Price, K., "Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, 1997.

[16]Kennedy, J. and Eberhart, R., "Particle Swarm Optimization," *IEEE*, 1995.

[17]SciPy, "Scipy Differential Evolution," 2017, Retrieved 01/2017 from https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.diffe

[18]Lee, A., "Particle Swarm Optimization (PSO) with constraint support," Retrieved 03/2017 from http://pythonhosted.org/pyswarm/.

[19]Dees, P., Zwack, M., Steffens, M., and Edwards, S., "Augmenting Conceptual Design Trajectory Tradespace Exploration with Graph Theory," *AIAA SPACE Conference and Exposition*, 2016.

[20]Zwack, M. R. and Dees, P. D., "Application of Design of Experiments and Surrogate Modeling wiwith the NASA Advanced Concepts Office, Earth-to-Orbit Design Process," *AIAA Space Conference and Exhibition*, 2016.

[21]Storn, R., "On the usage of differential evolution for function optimization," *Biennial Conf. of the North American Fuzzy Information Processing Society*, 1996.

[22]SAS Institute Inc, *JMP 13 Design of Experiments Guide, Second Edition*, SAS Institute Inc, 2017.

[23]JMP, Version 13. SAS Institude Inc., "Cary, NC," 1989-2007.

[24]Akaike, H., "Information theory and an extension of the maximum likelihood principle," 1971, pp. 267–281.

[25]National Institute of Science and Technology, "NIST/SEMATECH e-Handbook of Statistical Methods," 2017, Retrieved 07/2017 from http://www.itl.nist.gov/div898/handbook/.

[26]Box, G., Hunter, W., and Hunter, J., *Statistics for Experimenters: Design, Innovation, and Discovery, 2nd Edition*, John Wiley and Sons, Inc., 2nd ed., 2005.