American Journal of Aerospace Engineering
2014; 1(2): 8-15
Published online July 20, 2014 (http://www.sciencepublishinggroup.com/j/ajae)
doi: 10.11648/j.ajae.20140102.11

Application of response surface methodology during conceptual design for mission analysis

Johar Daudi

Department of Aerospace Engineering, School of Engineering, University of Glasgow, UK

Email address:
daudij@yahoo.com

To cite this article:
Johar Daudi. Application of Response Surface Methodology during Conceptual Design for Mission Analysis. *American Journal of Aerospace Engineering.* Vol. 1, No. 2, 2014, pp. 8-15. doi: 10.11648/j.ajae.20140102.11

Abstract: The design of complex engineering problems requires computation of several optimal parameters that is generally very time consuming and computationally expensive process. When using computationally expensive simulation programs/algorithms in engineering design for optimization, sometimes it becomes impractical to rely exclusively on simulation codes only. This study involved the designing of a suitable metamodel by using Kriging response surfaces which will be used for global optimization purposes. The study also covers the implementation of Kriging mathematical model in the form of a computer algorithm which is written in MATLAB.

Keywords: Surrogate/Meta Model, System Function, Design of Experiments

1. Introduction

Jones et al (1998) establishes an important foundation regarding global optimization of simulators and their use in engineering optimization problems where the number of function evaluations is limited by time or cost. In regard to this perspective, it is suggested to fit particular response surfaces to data collected by evaluating the function at a few points. [1]

Within the engineering community, response surfaces are gaining ever more popularity as a way of developing fast surrogates for time-consuming computer simulations. By running the simulations at a set of points (experimental design) and fitting response surfaces to the resulting input-output data, fast surrogates are obtained that can be used for optimization. These are computational cheap and time is saved by exploiting the fact that all runs used to fit the surfaces can be done in parallel, that runs can be started before one has even formulated the problem, and that runs can be reused when solving modified versions of the original problem

There is a lot of difference in some given data sample and the information that samples has in it. It is extremely useful if statistical analysis can be employed to the given data to extract useful information on it. Kriging, belongs to the group of geostatistical techniques that interpolate the value of a random variable at an unknown place in the design domain, based on the information it gathers from the given set of observed values. It is used as a method of predicting values of a function which has continuous distribution. The real strength of this method is based on the fact that it evaluates statistical variation in values in two respects i.e. distance and direction while predicting values to minimize error.

Based on the works of Daniel Gerhardus Krige, Georges Matheron, a French mathematician developed this Kriging method.

2. Mathematical Model

There are several methods of generating response-surfaces / surface-fits. Response surfaces are not only a mere representation of the input data but these also represent trends/patterns in the given data. Some of the methods that are generally used for surface fitting are as follows; [2]

1. Minimum Curvature
2. Kriging
3. Inverse Distance
4. Linear
5. Profiles
6. Quintic
7. Polynomial
8. Bidirectional
9. Triangulation

Out of these methods, Kriging is gaining ever more

popularity for metamodeling and design of experiments due to it's inherit characteristics which will be discussed below.

2.1. Kriging Method

Kriging is a statistical based interpolation method which takes into account i) distance and ii) degree of variation between known data points when predicting values in unknown design space. The Kriging predicted-value is weighted linear combination of the known observed values around the point to be estimated. While predicting values it tries to minimize the error-variance and furthermore it provides a statistical estimation of the error at each interpolated point, providing a measure of confidence in the predicted value. [3]

A random variable can be expressed as given in the following expression, [4]

$$Y(x) = f(x) + Z(x) \tag{1}$$

where $Y(x)$ is a random variable on the (x) - parameter. $Y(x)$ is the interpolated point as given by Kriging corresponding to the function $f(x)$ (which in some cases is considered as constant and taken equal to mean value) and $Z(x)$ as error deviation of the predicted value from the true function. Mostly interpolation functions regard $Z(x)$ as independent. But in Kriging metamodel, $Z(x)$ is modeled as realization of stochastic process with mean-zero, variance and non-zero covariance, considered as dependent and are modeled as a 'zero-mean' Gaussian process. Thus Eq.1 can be written as

$$Y(x) = f(x) + Z(x) = \sum_{j=1}^{k} B_j f_j(x) + Z(x) \tag{2}$$

where $f_j(x)$ is referred to as basic functions, B_j are corresponding coefficients. Now the co-variance of $Z(x)$ is modeled as

$$\begin{aligned} &Cov\left[Z\left(x^i\right), Z\left(x^j\right)\right] = \sigma^2 R \\ &R = \left[R\left(x^i, x^j\right)\right] \end{aligned} \tag{3}$$

where σ^2 is the process variance, R is correlation matrix and $R\left(x^i, x^j\right)$ is correlation function between any two of n-sampled points x^i and x^j. R is (n x n) symmetric matrix with ones along the diagonal. There are a number of ways to model this correlation function. But generally the most widely used correlation function is Guassian correlation function, which is given as

$$R\left(x^i, x^j\right) = \exp\left[-\sum_{k=1}^{n} \theta \left|x_k^i - x_k^j\right|^P\right] \tag{4}$$

where 'n' represents the design variables, $\theta > 0$ and $p > 0$ but for Guassian correlation function p =2, Value of 'p' varies between 1 – 2, 2 being selected for continuous functions.

Now the predicted estimate of $\hat{y}(x)$ of the response surface $y(x)$ at untried values of x is given by the following expression

$$\hat{y} = \hat{B} + r^T(x) R^{-1} (y - f\hat{B}) \tag{5}$$

where y is the column vector of size $(n \times 1)$ of the given input data (observed values), f is the column vector of ones of the size of $(n \times 1)$ when $f(x)$ is considered as constant and equal to mean value of output as mentioned earlier. B in the above expression is calculated by using the following expression; [4]

$$\hat{B} = \left(f^T R^{-1} f\right)^{-1} \left(f^T R^{-1} y\right) \tag{6}$$

In the equation (5), $r^T(x)$ is the correlation vector of length (n) between an untried value of x in the design space and the given input values of x (i.e. $x^1, x^2 ..... x^n$) so $r^T(x)$ can be written as follows,

$$r^T(x) = \begin{bmatrix} R\left(x, x^1\right) \\ R\left(x, x^2\right) \\ ..... \\ ..... \\ R\left(x, x^n\right) \end{bmatrix} \tag{7}$$

The variance σ^2 between $\hat{B}$ and y is calculated by the

$$\sigma^2 = (1/n)\left[\left(y - f\hat{B}\right)^T R^{-1} \left(y - f\hat{B}\right)\right] \tag{8}$$

As it has been discussed that Kriging based metamodels also give a measure of error in the metamodel predicted value, that is using statistical interpretation as given by the expression in Eq.9, an approximated error between original data values and the predicted value can be calculated which in turn can show, the level of confidence in the prediction. The expression for mean squared error (MSE)

$$\phi(x) = \sigma^2 \left[1 + u^T \left(f^T R^{-1} f\right)^{-1} u - r^T R^{-1} r\right] \tag{9}$$

where $u = f^T R^{-1} r - f$

The expression of MSE, is statistically zero at the observed data points, and as it goes/moves away from the observed data points in the design space, the error will be more due to lower values of correlation between untried new points (in design space) and the observed data points, which in turn shows the uncertainty in predicted values or

represents lower confidence level in metamodel-based predicted value. It should be kept in mind that Eq. 9 will only give statistical approximation of error.

2.2. Computational Aspects

The computation of variance σ^2 as given by Eq. 8 cannot be carried out directly due to computational problems. So $\hat{B}$ should be calculated by orthogonal transformation as least square solution. [4]

$$\tilde{f}\, B \simeq \tilde{Y} \tag{10}$$

Least Square solution can be found in the following manner.

1. Compute "Economy Size" QR factorization of $\tilde{f}$,

$$\tilde{f} = QG^T \tag{11}$$

where Q has orthonormal columns and G^T is upper triangle.

2. Check that G and $\tilde{f}$ has a full rank. If not, this is an indication that the chosen regression functions were not sufficiently linearly independent, and computation should stop with giving an error on 'Ill Matrix'. or else compute the least squares solution by back substitution in the system

$$G^T \hat{B} = Q^T \tilde{Y} \tag{12}$$

The auxiliary matrices can also be used to compute the process variance σ^2 which was given in Eq.8.

$$\sigma^2 = (1/n)\left\| \tilde{Y}_{:} - \tilde{f}\, \hat{B}_{:} \right\|_2^2 \tag{13}$$

and the MSE is computed using the following expression.

$$\phi(x) = \sigma^2 \left[1 + \left\| G^{-1} u \right\|_2^2 - \left\| \tilde{r} \right\|_2^2 \right] \tag{14}$$

with

$$\begin{aligned} \tilde{r} &= C^{-1} r \\ u &= f^T R^{-1} r - f \end{aligned} \tag{15}$$

3. Design of Experiment (DOE)

Conducting experiments on the obtained metamodel implies in analyzing where and how to select the inputs (x*) at which to evaluate the original/system function, in order to minimize/reduce statistical uncertainty of predicted values.

There are several approaches which deal with the criteria of selecting new search points in the design space. Some of these methods have been taken and analyzed from available literature, "A Taxonomy of Global Optimization Methods Based on Response Surfaces" by Donald R. Jones (Journal of Global Optimization 21: 345–383, 2001, Kluwer Academic Publishers.)

As mentioned previously that Kriging has a statistical interpretation besides just predicting the assumed function values which is, it provides an estimate of the uncertainty/error (i.e. MSE) in the interpolator (i.e. metamodel based prediction). This error is employed in some criteria (or can be called Auxiliary Functions) developed for searching new points where actual function will be evaluated.

There are two different approaches in selecting new search points. They are 1-stage and 2-stage methods. 1-stage methods do not perform fitting of response surface to the given observed data. Instead, they find new search points by evaluating 'hypotheses' about the location of the optimum. The 'worth' of the hypothesis that the optimum occurs at a point x* with function value (f*) may be determined by examining the properties of the best-fitting response surface that passes through the observed data and the point (x*, f*).

In the 1st-stage of 2-stage method, a metamodel is generated based on the given sample of data (also known as observed data) and all parameters are estimated. Now in the 2nd -stage of this 2-stage method, assuming the estimated parameters to be 'true representers' of actual response surface, new search points are found which satisfy the auxiliary function. One thing to be noted is that metamodel parameters are sensitive to the given initial sample data that may or may not give the true shape of the actual/system function, if the sample is sparse, it would result in greater number of iterations to be performed to get the precise shape of actual/system function. During this, 2-stage method has been followed.

4. Objective Function Evaluation Criteria

Following two criteria were analyzed.

4.1. Maximizing the Probability of Improvement

This method is among the most commonly employed methods for finding a point in the design space where the probability of improving (PI) a function beyond some 'Target' T is greater. The uncertainty in the value as given by actual function and predicted value (given by the metamodel) is expressed as to 'be like' the realization of a random variable 'Y(x)' with mean 'y (x)' and standard error 's(x)'.

Now let the current best function value be 'fmin', then target T value for the improvement has be some number T < fmin. Thus the probability of improving (PI) by such amount is simply the probability that Y(x) < T . Assuming uniform distribution, the probability expression is given as,

$$PI = \Phi\left(\frac{T - \hat{y}(x)}{s(x)}\right) \quad (16)$$

where $\Phi(.)$ is the standard cumulative distribution function and 'T' has been taken as given per reference [1].

$$T = f\min - 0.25\left|f\min\right| \quad (17)$$

which carries out 'Search for' 25 % improvement in $f\min$ value.

One of the main features of using probability of improvement is that under certain assumptions the iterations will be very dense. An improved version of this form is Expected Improvement.

4.2. Maximizing Expected Improvement

The Expected Improvement (EI) method is based on the criteria of evaluating how much improvement is expected if sampling is done at a particular point. [1]

Let Y(x) be a 'normally distributed' random variable representing uncertainty about the function's value at a point x, with mean *y(x)* and variance given by the Kriging predictor.

The expected improvement function is described as given in eq. 18. This function is useful as it provides a balance between exploration of regions with high uncertainty and exploitation of the most promising regions of design variable space.

$$EI[x] = s(x)\left[u(x)\Phi(u(x)) + \phi(u(x))\right] \quad (18)$$

where

$$u(x) = \frac{f\min - y(x)}{s(x)} \quad (19)$$

f min is the lowest objective function value of the sampled points.

$\Phi(u(x))$ and $\phi(u(x))$ is the standard cumulative distribution function and density function (having mean = 0 and standard deviation =1) and $\hat{y}(x)$ is predicted value from, s(x) is the root mean squared error in the prediction of x.

The first part on the right hand side of Eq.18 emphasizes on searching around the current minimum value, while on the other hand the second part emphasizes on searching in regions of high uncertainty, thus provides an automatic balance between exploitation and exploration which is a very attractive criteria for searching..

As according to literature [1], using the criteria of Expected Improvement has certain advantages.

1) It avoids the need specify a desired improvement (i.e., the target *T* of the previous section).
2) For different cases, the iterations in this method can be dense.
3) It provides stopping criteria for the program, '*stop when the expected improvement from further search is less than some small positive number'* [1].

By achieving the stopping criteria, it is meant that the metamodel has been sampled at sufficient points i.e. system function has been evaluated at enough number of points, and now it can be used with confidence to predict the global minimum of the metamodel model.

5. Algorithm Development

The main architecture for the code development and implementation is explained in Figure-1. In the implementation of search strategy and optimizing the auxiliary function, a global optimizer has been used whereas for the optimization of correlation parameter (θ-theta) Matlab function *'fminsearchbnd'* has been used which as proven to computationally cheap function.

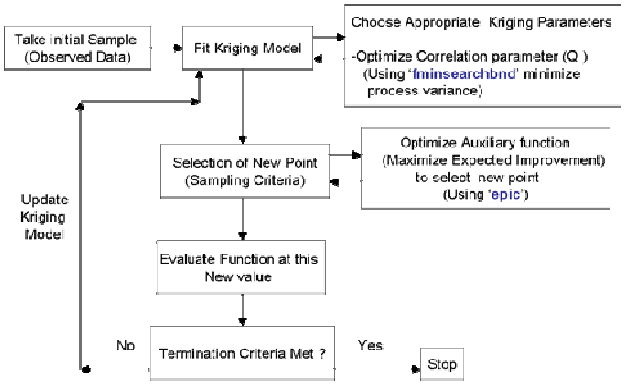


Figure 1. *Architecture for Code Development*

Optimizing the auxiliary function to predict the next iteration point is computational expensive and at the end of optimization of auxiliary function, the actual function is evaluated at the point which optimized auxiliary function. At the end of each iteration, the optimized points and the value of the system function is added to the data samples and Kriging meta-model is updated (this includes optimization / fine tuning of theta). This continuous up-dating of the Kriging metamodel at every iteration; improves the Kriging metamodel for better convergence.

When the condition for stopping criteria has been satisfied, it implies that metamodel has been sampled at sufficient points and now it can used with confidence to predict the global minimum of the metamodel model.

In this code, the stopping criteria has been implemented in 2-ways, either the code can be stopped after all the iterations have been performed or while the value of objective function falls below a certain threshold value. Another check on objective function has also being implemented that is to see if the value of objective function is not changing for certain number of iterations e.g. keep a check and monitor last 10-values of objective function and if they are not changing then the code will be stopped suggesting that the

metamodel has been converged.

It was implemented and analyzed for several test cases and the results were carefully studied and cross-matched with reference values of taken test case, it was seen that the stopping criteria was working well and giving satisfactory results.

6. Validation of Algorithm & Case Studies

After the development of any computer algorithm and written code, the main issue is to validate the results generated by the algorithm. Unless and until proven by several hundred of iterations, careful analysis and comparison with already published results, the results of the developed algorithm cannot be trusted with a greater degree of confidence.

For case studies and analysis, various test functions have been taken from the Dixon-Szeg, An introduction Towards Global Optimisation, North-Holland, Amsterdam 1978, which are considered to be bench mark for global optimization.

As a matter of preference which auxiliary function (PI or EI) should be used, both have been implemented and analyzed. The concluding remarks about the use of recommended criteria are given later.

7. Low thrust Trajectory and Propellant Optimization

Use of electric engines as primary sources of propulsion has given a new concept to future interplanetary missions, but at the same time has increased the complexity of trajectory design. Low-thrust trajectories are generated by shaping the trajectory through a set of parameterized pseudo-equinoctial elements. The characterization of solution space for a particular set of planetary missions and launch date selection are then carried out through a global optimization method, which in this case is being carried out by the developed algorithm.

To perform the Low-Thrust Approach analysis, an algorithm developed by the Space Research Group, Department of Aerospace at University of Glasgow has been used which is based on the method presented by Professor Massimiliano Vasile and Professor P. De Pascale in "An Approach to the Preliminary Design of Low Thrust Multiple Gravity-Assist Trajectories". One single iteration of developed algorithm takes about 10 minutes to run, which is computationally very expensive if considered for optimization purpose.

The parameters that are a part of design formulation for low thrust analysis involve the following,

a. Efficiency of the engine
b. Efficiency of the power system
c. Specific power at 1AU [W/m^2]
d. Launch date [MJD2000]
e. Time of flight [day]
f. Area of the solar arrays [m^2]
g. Specific power, or power to thrust ration [W/N]
h. Wet mass at launch [kg]

which will result in findings for appropriate/optimized values of

a. Mp : Propellant mass (kg)
b. NR : Number of revolutions giving the best result (integer)

The lower and upper bounds of the parameters to be optimized within the given limits are stated below in Table-1. The wet mass at launch has been fixed at 1000 kg for this case analysis. The objective function to be optimized here is the propellant mass (kg).

The case of Low thrust given in Table 1; has 7 parameters to be optimized, which represents a very large search space. So the criteria of stopping had been changed to iteration-based stop i.e. if the total number of iterations has been reached stop the code.

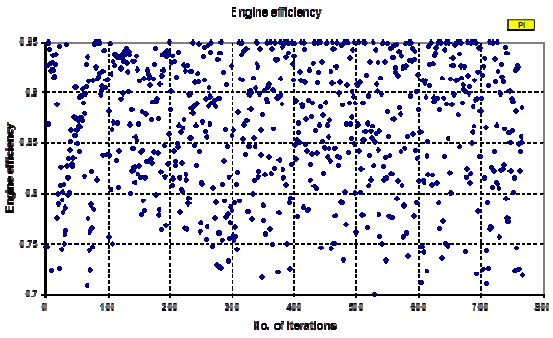
Table 1. *Low Thrust Parameters*

Engine Efficiency	0.7– 0.95	Launch Date MJD2000	2650-5475
Power Subsystem Efficiency	0.6-0.9	Time of Flight	100-800
Specific Power at 1AU (W/m2)	240-380	Area of Solar Planes (m^2)	5-20
Specific Power (W/N)	17-37	Wet Mass at Launch (kg)	1000

Total no of iterations has been taken equal to 800. Initial sample size was taken as 10. The case of low thrust was optimized using both PI and EI. It is to be kept in mind that PI-based runs were stopped at 767, where EI completed whole 800 iterations.

The results obtained after the end of 800 iterations are given below. Yellow highlighted box represents the case for PI and the magenta one represents the case for EI.

Figure-2 set for both PI and EI represent that search has been performed almost uniformly and both the criteria, try to touch the upper limit of 0.9 for engine efficiency. For the case of PI, in the beginning, it doesn't reach 0.9 and is searching more widely and by the end, since the PI is decreasing, it finds 0.9 more often whereas for EI it locates 0.9 throughout the beginning but keeps on exploring the search space.



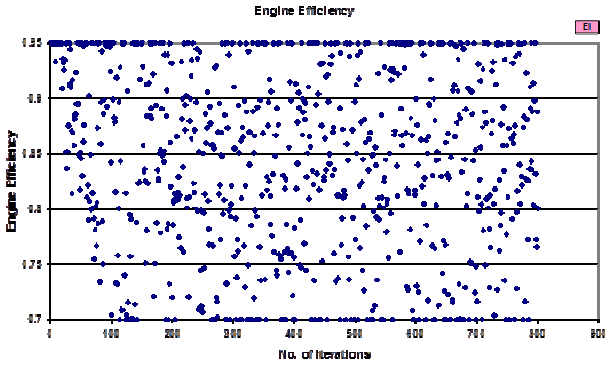


Figure 2. *Engine Efficiency*

Figure-3 shows the search pattern for power efficiency, it can be seen for the case of PI, again the search has been performed quite widely. However the most interesting pattern is seen with EI, where in the beginning, the values are more or less in concentrated form and after a number of iterations, it begins to search more globally. The reason can be attributed to the fact that if initial sample is sparse and deceptive, then it will give small estimates of standard error. Thus, it performs an exhaustive search around the initial best point before beginning to search more globally.

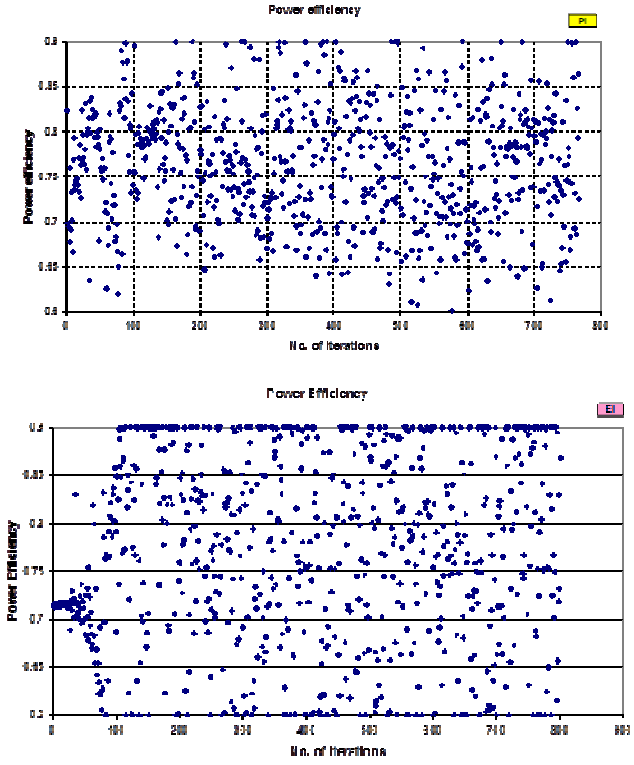


Figure 3. *Power Efficiency*

The search pattern is almost same as for the case of engine efficiency, both the criteria stress on using the upper limit 380 (W/m^2) as shown for Figure-4. For the case of EI, search has been performed more globally in comparison with PI.

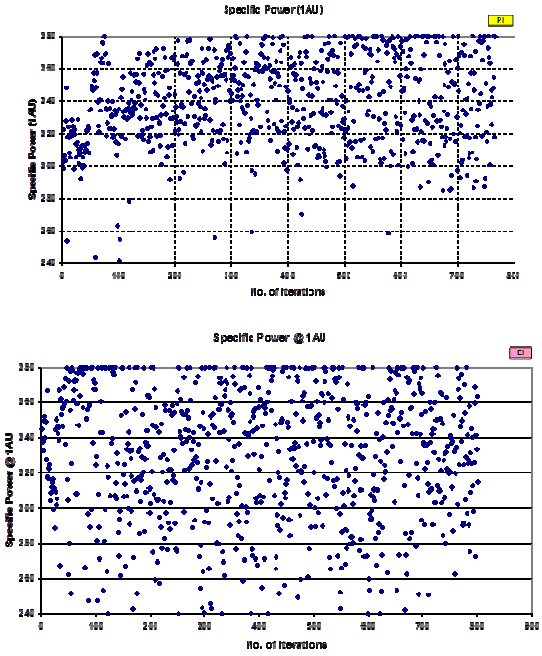


Figure 4. *Specific Power*

Figure-5: In case of optimizing launch day, it has been observed that here two criteria choose different values, for PI the search is more concentrated around a range bracket of 4000 & 4200 MJD2000 where as for EI case, it searches both the lower and upper end of given bounds but is more concentrated towards the upper end of bound.

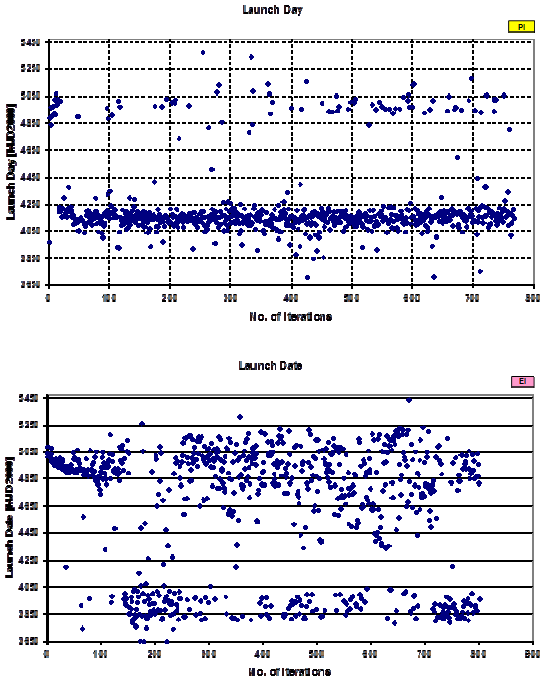


Figure 5. *Launch Day*

Figure-6 shows that PI is again concentrated towards the upper boundary limit whereas EI performs a more global search. Here again one fact can be noted that the initial search in the case of EI is more concentrated and then it gets

dispersed to search more globally, as it was found in the case of power efficiency above. The reason can be attributed to initial sample.

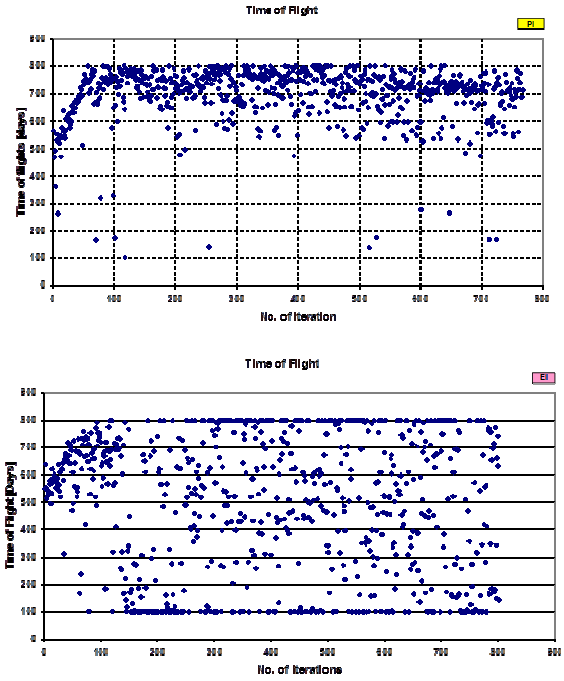


Figure 6. *Time of Flight*

Both the criteria show trend towards using the upper limit of area for solar arrays i.e. 20 m^2. Same trend is being followed that is choosing the upper limit of specific power of 37 W/N.

Figure-7 if seen carefully shows that EI criteria finds minimum early in the beginning of iterations as compared to PI.

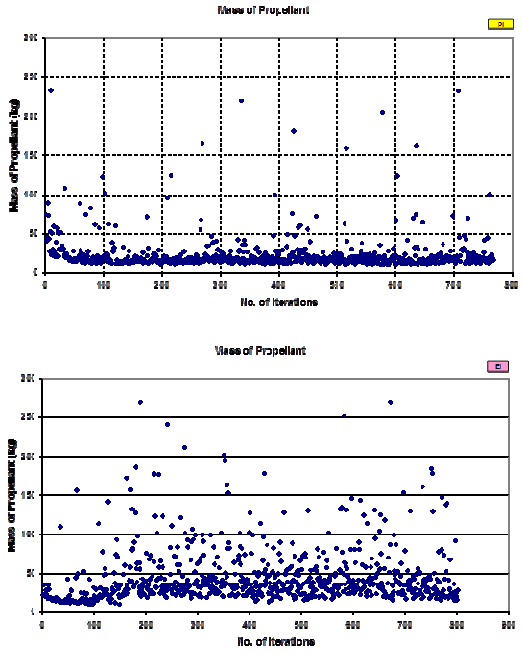


Figure 7. *Mass of Propellant*

As for the case of number of revolutions both PI and EI show a concentrated trend towards selecting 1.

It has been most interesting to see the variation of objective function i.e. variation of PI and EI. Carefully analyzing Figure-8; for PI it is evident that in the beginning probability of finding lower values is high, which forms a zig-zag pattern having certain peaks, but overall the trend is decreasing as the number of iterations increase. If the number of iterations would have been higher than the current selected of 800, it trend would have touched zero-line, implying no further improvement.

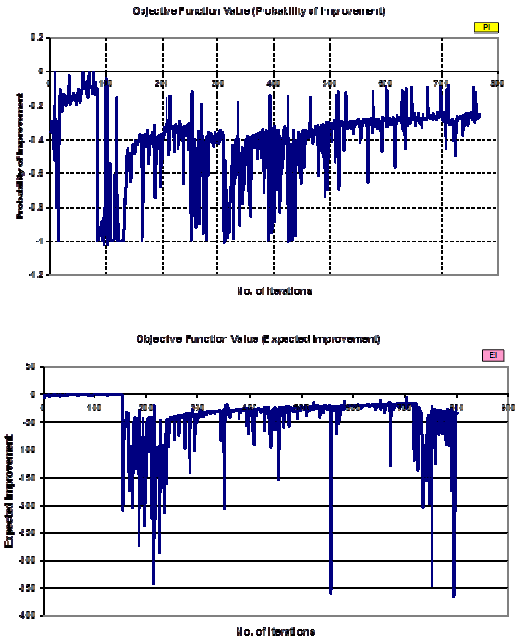


Figure 8. *Objective Function*

For the case of EI, it has been observed that initially the expected improvement is very low. The main influencing factor is the initial sample which can be sparse and deceptive, giving small estimates of error. Consequently, making the algorithm to search near to the already best available and carrying out exhaustive search before beginning to search more globally. However, it can be seen that the overall trend is decreasing as the number of iteration increase.

The final optimized values of input variables is given by Table 2, where both the criteria emphasize on using values near upper bound of engine efficiency, power subsystem efficiency, specific power, area of solar arrays.

Table 2. *Optimized Parameters*

Parameters	PI	EI
Engine Efficiency	0.95	0.949
Power Subsystem Efficiency	0.821	0.706
Specific Power at 1AU , W/m^2	369.4	380
Specific Power (W/N)	37	36.68
Launch Date MJD2000	4147.64	4893.25
Time of Flight	753.96	766.05
Area of Solar Planes (m^2)	20	20
Propellant Mass, kg	11.08	1
No of Revolutions	10.99	1

It has been seen that since this low thrust case has a large search space, so for future analysis the total number of iteration should be increased from 800 to higher value.

8. Conclusion & Recommendations

After the development and implementation of the computer algorithm, different test cases were analyzed and following observations and conclusions were made.

1) If the initial data sample is sparse then lots of iterations are utilized in fine tuning the Kriging model.
2) While selecting the auxiliary function of Probability of Improvement PI, it was seen that PI is affected by the value of the target 'T'. If the PI is very small, the search will be initially performed locally before searching globally which means more iteration will be required for convergence which is expensive talking in terms of time and computational effort. On the vice versa, if 'T' is very high, the search will be carried out in more global way and more of iterations will be carried out in fine tuning the model and consequently finding the optimum point. This search criteria works well with simple cases but for complex cases it fails to find the true global minimum.
3) Regarding the method of EI, it has been shown in Locateli's proof; the expected improvement method does find the global minimum [1]. But as stated in the literature and also observed while analyzing the performance, for different test cases it takes large number of iterations to converge. It has been seen that the initial sample is very deceptive, giving very small estimates of the standard error. Resultantly, only points that are close to the current best point have high expected improvement. So it carries out 'exhaustive search' near the initial best point before it begins to search globally.
4) It has been seen that the value of correlation (theta) has an effect on the predicted values and thus objective function value as well. The limits of theta need to be defined. Based on the stop criteria the model used to converge prematurely, however changing theta refined the model well, yielding true global minimum. It has been observed that initial value and variation of EI can be attributed due to initial sample and model fit and most importantly to the nature of function/problem.
5) If the search space is too large, then it is suggested that criteria of iteration-based stop should be used.
6) The auxiliary function, expected improvement showed robustness in accurately locating the global minimum; it is recommended that this criteria should be followed while carrying out future analysis.

Reference

[1] A Taxonomy of Global Optimization Methods Based on Response, DONALD R. JONES, General Motors Corporation, Journal of Global Optimization 21: 345–383, 2001, Kluwer Academic Publishers.

[2] http://www.hesc.it/tnt/tutorial/surfmodl-ENG.pdf

[3] http://lazarus.elte.hu/hun/digkonyv/havas/mellekl/vm25/vma07.pdf

[4] DACE, A MATLAB Kriging Toolbox Version 2.0, August 1, 2002, Søren N. Lop-Haven, Hans Bruun Nielsen, Jacob Søndergaard