Engineering with Computers
DOI 10.1007/s00366-010-0192-8

ORIGINAL ARTICLE

Data driven surrogate-based optimization in the problem solving environment WBCSim

S. Deshpande · L. T. Watson · J. Shu · F. A. Kamke · N. Ramakrishnan

Received: 9 November 2009 / Accepted: 18 June 2010

Abstract Large scale, multidisciplinary, engineering designs are always difficult due to the complexity and dimensionality of these problems. Direct coupling between the analysis codes and the optimization routines can be prohibitively time consuming due to the complexity of the underlying simulation codes. One way of tackling this problem is by constructing computationally cheap(er) approximations of the expensive simulations that mimic the behavior of the simulation model as closely as possible. This paper presents a data driven, surrogate-based optimization algorithm that uses a trust region-based sequential approximate optimization (SAO) framework and a statistical sampling approach based on design of experiment (DOE) arrays. The algorithm is implemented using techniques from two packages—SURFPACK and SHEPPACK that provide a collection of approximation algorithms to build the surrogates and three different DOE techniques—full factorial (FF), Latin hypercube sampling, and central composite design—are used to train the surrogates. The results are compared with the optimization results obtained by directly coupling an optimizer with the simulation code. The biggest concern in using the SAO framework based on statistical sampling is the generation of the required database. As the number of design variables grows, the computational cost of generating the required database grows rapidly. A data driven approach is proposed to tackle this situation, where the trick is to run the expensive simulation if and only if a nearby data point does not exist in the cumulatively growing database. Over time the database matures and is enriched as more and more optimizations are performed. Results show that the proposed methodology dramatically reduces the total number of calls to the expensive simulation runs during the optimization process.

Keywords Problem solving environment · Wood-based composite materials · Visualization · Optimization · Surrogate · Response surface approximation · Sequential approximate optimization · Experiment management · Trust region strategy

1 Introduction

Large scale, multidisciplinary, engineering design problems require physical experiments and/or simulations to evaluate a design objective as a function of design parameters. For many real world problems, however, a single simulation can take several minutes, hours, or even days to complete. As a result, routine tasks, such as design optimization, design space exploration, and sensitivity analysis could become almost impossible since they might require hundreds or even thousands of simulations. One way of tackling this problem is by constructing computationally cheap(er) approximations of the expensive simulations, that mimic the behavior of the simulation model as closely as possible. These approximations are known as

S. Deshpande (✉) · L. T. Watson · J. Shu · N. Ramakrishnan
Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0106, USA
e-mail: shubhgd@cs.vt.edu

L. T. Watson
Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0106, USA
e-mail: ltw@ieee.org

F. A. Kamke
Department of Wood Science and Engineering, Oregon State University, Corvallis, OR 97331, USA

Published online: 14 July 2010

surrogates, response surface approximations (RSAs), metamodels, or emulators.

This paper discusses a data driven, surrogate-based optimization algorithm illustrated by a scientific problem solving environment (PSE), WBCSim, which increases the productivity of wood scientists conducting research on wood-based composite (WBC) materials. WBCSim integrates legacy FORTRAN 77 and new Fortran 90 simulation codes with a Web-based graphical front end, an optimization tool, an experiment management component, a computational steering capability, and various visualization tools. As discussed in [31], WBCSim has evolved steadily from a prototype PSE, intended as a research tool and a Web interface for legacy Fortran programs, to a commercial quality PSE. The current version of WBCSim has enhanced visualization and simulation capabilities, more realistically modeling manufacture. The more advanced models in WBCSim, such as the hot pressing Fortran 90 code or its visualization/optimization tools, can take hours to run on a fast (DEC AXP 21064 or SUN Sparc) workstation. Nonlinear optimization algorithms cannot be applied directly to these complex simulation models, as it can be prohibitively time consuming and cost ineffective. The solution discussed here is to provide a computationally inexpensive representation of the underlying system. A strong motivation behind the implementation of a data driven, surrogate-based optimization algorithm for WBCSim is the availability of a sophisticated experiment management component, which efficiently manages the simulation execution and experiment data, providing a systematic way to automatically store and retrieve the simulation data [29]. The existing simulation run data can be retrieved to construct a surrogate function for the entire simulation or parts of it, thereby replacing costly simulation executions with cheap(er) surrogate function evaluations.

As discussed in [20, 21] the surrogate models can be integrated within optimization tools in two ways: (1) using global approximations, where a RSA is developed over the entire design space, or (2) using local approximations, where RSAs are built within a local region around the current design. The global approximations require a more complex model to mimic the underlying system, consequently, the cost of developing a global surrogate model is higher than for local approximations. In general, a single optimization is performed while employing global approximations, whereas local approximations require a series of optimizations, each one performed within a local region around the current design. When using local RSAs, a sequential approximate optimization (SAO) methodology can be used. The basic concept of the SAO framework is to apply nonlinear optimization to an approximation at the current design point subject to local move limits. The design space is sampled around the current design point at the beginning of each SAO iteration to generate the dataset required for constructing a surrogate model using regression analysis.

The implementation here is based on the second approach using the SAO framework and a statistical sampling approach based on design of experiment (DOE) arrays as reported by Rodrígues et al. [24, 25]. At each SAO iteration, a DOE array is used to select a set of design points for sampling. Each design point is evaluated either by retrieving a previously stored simulation run or by running the simulation code at the design point, if it does not exist in the database. The resulting dataset is used to build a surrogate model. An optimization is performed using this approximation model within local move limits. The surrogate and move limits are updated after every iteration using a trust region strategy until convergence is achieved.

The organization of the paper is as follows: Sect. 2 reviews related work in PSEs and WBCSim in particular, describes the surrogate-based optimization and SAO framework, and discusses database support within the context of PSEs. Section 3 describes the proposed surrogate-based optimization methodology in detail, and presents a pseudocode for the optimization algorithm. The experimental results are discussed in Sects. 4, and 5 offers concluding remarks.

2 Background

2.1 Problem solving environment and WBCSim

A problem solving environment is a system that provides a complete, usable, and integrated set of high level facilities for solving problems from a prescribed domain [9, 15]. A PSE commonly addresses various issues: Internet accessibility to legacy codes, visualization, experiment management (EM), multidisciplinary support, recommender systems, collaboration support, optimization, high performance computing, preservation of expert knowledge, design extensibility, and pedagogical uses [39]. PSEs were first introduced in the simpler problem domains such as partial differential equations (ELLPACK and its descendants [5], for solving two and three-dimensional elliptic partial differential equations) and linear algebra (Linear System Analyzer [4] for manipulating and solving large-scale sparse linear systems of equations). Since then, many PSEs have been introduced to address problems in diverse domains, such as Gismo [3], created at Washington University, for modeling all aspects of a satellite's design and performance; a PSE developed by Chen et al. [6], to simulate physically realistic, complex dust behaviors useful in interactive graphics applications for education,

entertainment, or training; Expresso [32], a microarray experiment management PSE, designed to assist biologists in planning, executing, and interpreting microarray experiments; L2W [7], a PSE for land use change analysis; JigCell model builder [1, 37, 38], a PSE to define chemical kinetic models as a set of reaction equations, and many more. Watson et al. [39] provide a thorough summary of the key attributes of a PSE, and also a comparative study of a PSE with other similar computing environments: a decision support system (DSS) and a geographical information system (GIS).

The review here is focused on the work regarding multidisciplinary optimization support provided by PSEs. A number of PSEs have been introduced that combine analysis codes with optimization methods in a flexible manner, along with a visualization tool for viewing the optimization results. iSIGHT [34] is a PSE that provides a generic shell environment for multidisciplinary optimization. LMS optimus [11] is a system that provides a front end to set up a problem, select a method suitable to the problem, and analyze the results. DAKOTA [8] is a framework that provides a flexible, object-oriented, and extensible PSE with an integrated interface for a variety of optimization methods. S^4W [19, 33] is a collaborative PSE for the design and analysis of wideband wireless communication systems. VizCraft [12] is a PSE that provides a graphical user interface to a widely used suite of analysis and optimization codes to aid aircraft designers during conceptual design of a high-speed civil transport. WBCSim (discussed next) is a prototype PSE for WBCs manufacturing that provides support for a sophisticated optimization component along with various visualization tools. Among these PSEs, S^4W uses surrogate functions for its WCDMA simulator to estimate the bit error rates [19].

WBCSim is a prototype PSE for WBCs simulations that integrates a set of high level components for making both legacy and new Fortran codes widely accessible. WBCSim qualifies as a PSE because it provides Internet access to Fortran codes via the Web, is equipped with visualization and optimization tools, has a sophisticated experiment management (EM) component, a computational steering capability, and has support for collaboration and high performance computing being added. WBCSim currently supports five simulation models:

1. Composite material analysis (CMA). The CMA model was developed to assess the stress–strain behavior and strength properties of laminated fiber-reinforced materials (e.g., plywood) [31].
2. Oriented strand board mat formation (OSB). The mat formation model [43] creates a three-dimensional spatial structure of a layered WBC (e.g., oriented strand board and waferboard) and also calculates certain mat properties by superimposing a mesh on the mat structure.
3. Hot compression (HC). The hot compression model simulates the hot pressing of a flake mat, created by the mat formation model, in a batch press, using two-dimensional heat and mass transfer theory. It calculates the internal environmental conditions, such as the temperature, moisture content, and pressure changes, as well as adhesive cure during the mat consolidation process [43, 44].
4. Radio-frequency pressing (RFP). This model simulates heat and mass transfer in wood, resulting in the consolidation of wood veneer into a laminated composite, when subject to power input by an alternating electric field [26].
5. Rotary dryer simulation (RDS). The RDS model was developed as a tool that assists in the design and operation of a rotary drying system for wood particles [16, 17].

Goel et al. [10] first described the three tiered software architecture for WBCSim. The current implementation of WBCSim follows the same architecture with the addition of an EM component and support for XML datasheets [30], and a computational steering capability. The three tiers in the architecture correspond to (1) the client layer—user front end, (2) the server layer—a Web server and a PHP module, and (3) the developer layer—the Fortran code and various visualization and optimization tools running on the server.

The current implementation of WBCSim supports the optimization package DOT (design optimization tool) [36] based on sequential quadratic programming and the method of feasible directions. Two of the five models supported in WBCSim—RDS and RFP—are linked to DOT. This paper describes a computationally inexpensive surrogate based optimization method that intends to improve the underlying system performance while applying optimization algorithms to the computationally expensive more advanced models (e.g., hot compression) in WBCSim. Note that while the optimization results here are for the RDS model (since the HC model is not yet linked to DOT), the motivation for and ultimate application of this surrogate based optimization work is using DOT with the expensive HC model. A typical RDS simulation takes 545 ms versus 2 h for a HOT simulation.

The various visualization tools that are integrated in WBCSim include Virtual Reality Modeling Language [2], Wolfram's Mathematica [40], and the UNIX utility WhirlGif. Shu et al. [31] present a detailed treatment of these tools.

WBCSim has an efficient experiment management component that integrates a Web-based graphical front end,

server scripts, and a database management system to allow scientists to easily save, retrieve, and perform customized operations on experimental data [29].

WBCSim has been enriched with a recent addition of XML datasheets to unify its implementation layers [30]. An XML datasheet is tailored for each of the five models mentioned above. The WBCSim interface layer, the server scripts, and the database management system all use the same XML datasheet for a particular model. The use of XML reduces redundancy and improves the usability and maintainability of the client, server, and developer layers. A computational steering capability for the hot pressing process simulation has also been added. Now the user can view temperature, pressure, and moisture content profiles within the mat during the hot pressing simulation, and interactively modify the press schedule or abort the simulation. Such steering significantly enhances user productivity and insight into the manufacturing process.

WBCSim has evolved in various ways over many years, and has now become a sophisticated, mature PSE, equipped with a complete suite of high-level tools that make it a uniquely valuable system for the WBC industry. Yet, its original goals remain the same: (1) to increase the productivity of WBC research and manufacturing groups by improving their software environment, and (2) to continue serving as an example for the design, construction, and evaluation of small-scale PSE. The surrogate-based optimization algorithm presented in this paper intends to contribute towards these goals by significantly enhancing the system performance for optimization.

2.2 Surrogate-based optimization and a SAO framework

The more advanced models in WBCSim such as the hot pressing model (a two-dimensional nonlinear partial differential equation) are quite complex and can take hours to run on a fast (DEC AXP 21064 or Sun Sparc) workstation. Applying a nonlinear optimization algorithm directly to these complex models can be prohibitively time consuming due to the complexity of the underlying simulation codes. One way of alleviating this burden is by integrating RSAs or surrogate functions with nonlinear optimizers to reduce the CPU time required for the optimization of complex multidisciplinary systems. RSAs provide a computationally inexpensive lower-fidelity representation of the underlying simulation. In large scale multidisciplinary engineering design, the construction and use of such surrogates has become a standard practice.

As discussed above, two trends have emerged to integrate surrogate functions with a nonlinear optimizer: (1) using a global RSA, or (2) using local RSAs. One mechanism for utilizing local RSAs is the SAO framework. In SAO, simple RSAs that are valid for a local region are built for the objective function and the constraints. An optimization algorithm is applied to this approximation within the local trust region defined by local move limits. The surrogate functions and trust region (local move limits) are updated at every iteration until convergence is achieved. Different SAO strategies have been developed [20–25, 27, 41, 42], depending on the sampling approaches used and move limit methods implemented. This paper presents an algorithm that implements a SAO framework using DOE-based sampling and a trust region method to adjust the move limits. The only constraints for the WBCSim models being optimized are simple bound constraints on the variables, hence RSAs are only required for the objective function.

The algorithm starts with iteration $k = 0$ at some feasible design point. The move limits are defined around the design point and a database is generated for the local trust region using a DOE array. A RSA that is valid near the current design point is built using the generated database. A nonlinear optimization is then performed using this approximation. When the optimization returns a new candidate point, a trust region test is applied to decide the acceptance of the approximation and to define the next move limits. Based on the trust region ratio, the new candidate point is either accepted or rejected, new move limits are defined, and optimization proceeds until convergence is achieved. Figure 1 presents a flowchart for a general SAO framework.

The main concept of a trust region method is to monitor how well the approximation agrees with the true objective function using a trust region ratio ρ, the ratio of the actual improvement in the objective function to the RSA predicted improvement. For a detailed description of the trust region methodology, computation of the trust region ratio, and the adjustment of move limits, see [20].

The most expensive step in the SAO framework is the generation of a database for a local trust region at every iteration. The design points to be evaluated can be generated either using some optimization-based sampling as described in [25, 28, 41, 42], or using traditional DOE arrays as described in [22–24, 27]. Each design point can be evaluated by running the simulation code at that point. Several DOE strategies have been developed to generate efficient surrogate models. Among the common techniques that have been used to generate RSAs are traditional factorial designs (full and fractional factorial), central composite designs (CCD), orthogonal arrays (OAs), and space filling techniques, such as Latin hypercube and its extensions. More complex experimental designs such as D-optimal designs have been introduced to address the limitations of traditional DOE techniques. The SAO framework generally eschews such complex experimental designs. This paper presents an optimization algorithm and

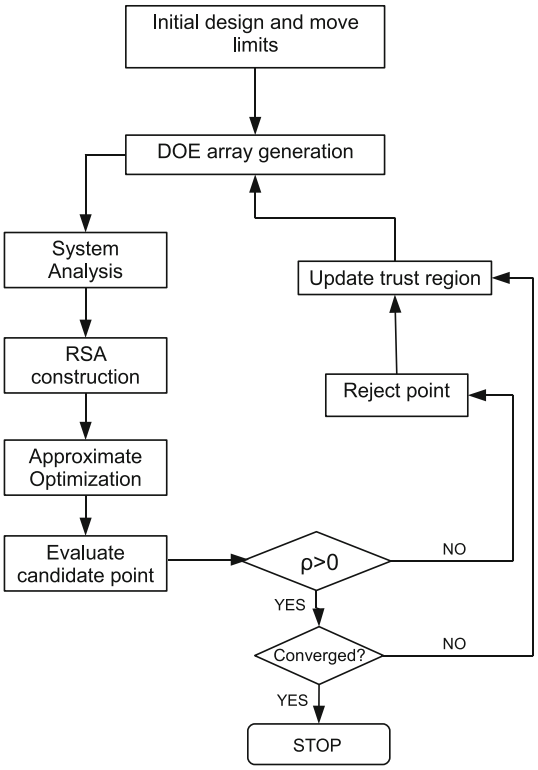

Fig. 1 Flowchart for a general SAO framework

a comparative study using various traditional DOE sampling techniques.

Numerous algorithms exist to generate RSAs that interpolate or fit data points. The SAO framework here uses techniques from two packages that provide a collection of approximation algorithms—SURFPACK [13, 14] developed at Sandia National Laboratories, and SHEPPACK developed by Thacker et al. [35]. SHEPPACK is a Fortran 95 package containing five versions of the modified Shepard algorithm: quadratic (Fortran 95 translations of ACM Algorithms 660, 661, and 798), cubic (Fortran 95 translation of ACM Algorithm 791), and linear variations of the original Shepard algorithm. SURFPACK provides a library of surrogate modeling methods including low order polynomials (linear (POLY1), quadratic (POLY2), and cubic (POLY3), KRIGING interpolation, multivariate adaptive regression splines (MARS), and artificial neural networks (ANN).

2.3 Database

Experiment management is a crucial component of any PSE. It provides a systematic and efficient way to store, retrieve, and manage experimental data. WBCSim is equipped with a sophisticated EM tool, which consists of customized user interfaces, server scripts, and an open source DBMS, Postgres. The EM tool not only supports all the features from the previous file-based system, but also significantly improves WBCSim user productivity, usability, and system maintenance in various ways by providing support for storing simulation inputs and outputs, retrieval of existing simulation runs instead of running a brand new simulation when inputs and/or outputs exist in the database, filtering the experiment data, and comparing stored simulation outputs. See [29] for a detailed description of the implementation of an EM component for WBCSim.

WBCSim has a full fledged working EM component for the simulation models OSB and RDS, which has support for optimization as well and hence is a testbed for the surrogate-based optimization algorithm presented in this paper. A strong motivation for the implementation of a data driven surrogate-based optimization algorithm in WBCSim is the availability of the required infrastructure. In SAO, a database is generated for each iteration by evaluating the objective function and constraints (called a "system analysis") at each DOE sample point. An EM component saves the cost of a simulation run when the data point already exists in the database (a simulation run is required if and only if the data point does not exist in the database). After running the simulation, the point and results are stored in the database for future references. Thus, over time, the database matures and is enriched as more and more optimizations are performed. This data-driven approach significantly reduces the total number of expensive simulation runs required and improves the underlying system performance. Based on the trust region test, if a candidate point is accepted, the new move limits are decided based on the trust region ratio as described in [20]. This paper describes an algorithm that adjusts the new trust region such that it makes maximal use of existing data points from previously generated trust regions, hence reducing the number of new data point evaluations (simulation runs) required for generating a new trust region. Thus, a data driven approach takes advantage of the fact that expensive simulations are run only when data points do not exist in the cumulatively growing database.

3 Optimization algorithm

Move limits define a "local design space" around the current point X_k, taken as the intersection of the trust region box $\left\{X \mid \|X - X_k\|_\infty \leq \Delta\right\}$ with the box $[L, U]$ defined by the variable bounds $L \leq X \leq U$. The user defined bounds L and U are used to scale each design variable between -1 and 1 replacing $X_k[i]$ by $\frac{X_k[i]-((U[i]+L[i])/2)}{(U[i]-L[i])/2}$ and making $[L, U] = [-1, 1]$. Assume henceforth that all the design variables are thusly scaled. A precise description of the data driven surrogate-based optimization algorithm described in the previous section follows.

Algorithm DDSAO

Input: p (a user specified start point), $minmax$ (min or max selection for the optimization), Δ (trust region radius), l, u (local move limits), N (number of design variables), DB (simulation database).
Output: optimum design point and objective function value.
Parameters:δ (trust region adjustment threshold), ϵ (tolerance for the absolute change in the true function and the surrogate), $\tilde{\delta}$ (relative error in the surrogate), $\hat{\delta}$ (error tolerance for the improvement in the successive iterates), η (lower limit on the trust region radius).

Initialize trust region radius Δ to 20% of the diameter of the entire design space;
$\delta := 0.1 * \Delta$;
$\epsilon := 1.0E-8$;
$\tilde{\delta} := 1.0E-5$;
$\hat{\delta} := 1.0E-5$;
$\eta := 1.0E-2$;
$count := 0$;
$k := 0$;
$convergence := false$;
if a point within Δ of p does not exist in DB OR DB contains no points in $[-1, 1]$ **then**
begin
run simulation at p;
insert p and corresponding objective function value $f(p)$ for p into DB;
end
select a point X_k from DB as a start point, where the point lies within the design space bounds $[-1, 1]$ and has optimum value $f(X_k)$;
define the local design space move limits $[l, u]$ on all the design variables around X_k using Δ.
while not *convergence* **do**
begin
generate DOE array A of size n for local design space around X_k;
for $i := 1$ **step** 1 **until** n **do**
begin
if a point within Δ/n of $A[i]$ does not exist in DB **then**
begin
perform SA at $A[i]$ to get $f(A[i])$;
insert $A[i]$ and $f(A[i])$ into DB;
end
end
build a surrogate model $\tilde{f}(X)$ using all DB points within 2Δ of X_k;
call DOT to optimize $\tilde{f}(X)$ in $[l, u]$ yielding X_{k+1};
$\rho := \dfrac{f(X_k) - f(X_{k+1})}{\tilde{f}(X_k) - \tilde{f}(X_{k+1})}$;
if $\rho \leq 0$ **then**
begin
if $|f(X_k) - f(X_{k+1})| \leq \epsilon$ **then**
$convergence := true$;
else
begin
reject X_{k+1};
$\Delta := 0.25 * \Delta$;
reset local move limits using Δ;
end
end
else
begin
accept X_{k+1}; $k := k + 1$;
if $\frac{|\tilde{f}(X_k) - f(X_k)|}{|f(X_k)|+1} \leq \tilde{\delta}$ **then**
for $i := 1$ **step** 1 **until** N **do**
if $(X_k[i] = l[i] = -1)$ OR $(X_k[i] = u[i] = 1)$ OR $(l[i] < X_k[i] < u[i])$ **then**
$count := count + 1$;
if $(\frac{\|X_k - X_{k-1}\|_\infty}{\|X_{k-1}\|_\infty+1} \leq \hat{\delta})$ OR $(count = N)$ OR $(|\tilde{f}(X_k) - \tilde{f}(X_{k-1})| \leq \epsilon)$ **then**
$convergence := true$;
else
begin
if $\rho \leq 0.25$ **then**
begin
$\Delta := \Delta * 0.25$;
if $\Delta \leq \eta$ **then**
throw an error message and exit;
end
else if $\rho > 0.75$ **then**
$\Delta := \Delta * 2$;
end
end
for $i := 1$ **step** 1 **until** N **do**
begin
if $X_k[i]$ is within δ of $l[i]$ **then**
begin
$u[i] := l[i] + \delta$;
$l[i] := l[i] - \Delta$;
end
else if $X_k[i]$ is within δ of $u[i]$ **then**
begin
$l[i] := u[i] - \delta$;
$u[i] := u[i] + \Delta$;
end
else
begin
$l[i] := (l[i] + u[i])/2 - \Delta/2$;
$u[i] := (l[i] + u[i])/2 + \Delta/2$;
end
if $l[i] < -1$ **then**
$l[i] := -1$;
if $u[i] > 1$ **then**
$u[i] := 1$;
end
end
return X_k and $f(X_k)$;

4 Experimental results and discussion

4.1 Design of experiments

Design of experiments (DOE) is a statistical-based approach for systematically and efficiently designing and analyzing experiments to determine the relationship between different factors affecting a process and the response of the process. A particular setting of design variables describes a typical experimental run, and a particular combination of runs defines an experimental design. The choice of the DOE methods is motivated by the six different approximation methods used to build a surrogate and the testbed PSE rotary dryer simulation (RDS) used for the optimization. The RDS model simulates drying behavior of the wood particles in a rotary dryer as discussed in [16, 17]. The RDS optimization has 13 variables that define the inlet conditions of the hot gases and wet wood particles, as well as the physical dimensions of the drum and lifting flanges, flow rates, and thermal loss factor for the dryer. The model predicts the particle moisture content, temperature, cumulative time, gas composition, and energy consumption. The experiments here used 3 of the 13 variables (temperature of drying gases, flow rate of inlet drying gases, and drum rotation speed) as design variables, and cumulative time as the objective function to be minimized. It was observed during the exploratory analysis that the objective function for evaluating the cumulative time is more sensitive to the flow rate of inlet drying gases than to the other variables. Hence the full factorial design is formulated to cater to this specific case. The class of central composite designs (CCD) is the most popular class of second-order designs, hence a face centered CCD was chosen. The KRIGING model used to build surrogates is known to work better with the space-filling designs, and the Latin hypercube design being one of the favorite space-filling designs, was chosen as the third DOE method.

Figure 2 is a two-dimensional representation of the three DOE methods used. A full factorial (FF) design array of sample size 12 is generated with 3 levels for the design variable corresponding to the flow rate of the drying gases and two levels for the other two. All the possible high/low combinations of all the three design variables form a FF design of size $3 \times 2 \times 2 = 12$ as shown in Fig. 2. A Latin hypercube sampling (LHS) is a space filling technique in which the design space is divided into n non overlapping intervals and one value from each interval is then selected at random to generate an array of n k-tuplets. A two-dimensional representation of the LHS design on a 10×10 grid is shown in Fig. 2 where $n = 10$ and $k = 3$. The third DOE technique implemented is the face centered central composite design (CCD) with eight corner points, six points with a point at the center of each face, and a point at the center of the local design space. Thus a DOE array of sample size 15 is generated as shown in Fig. 2.

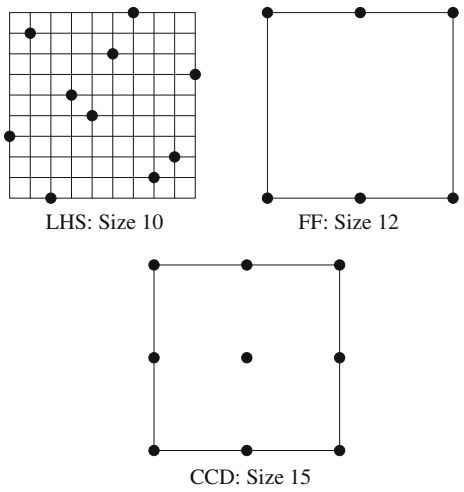

Fig. 2 A 2D projection of the DOE methods used. The FF points are $\{-1, 0, 1\} \times \{-1, 1\} \times \{-1, 1\}$, and the CCD points are $\{-1, 0, 1\}^3 \setminus$ {points with exactly one 0 coordinate}

4.2 A typical scenario

Figure 3 gives a graphical representation of a possible convergence scenario of the optimization algorithm through a series of iterations. The algorithm starts at X_0 with $\Delta = D = 0.2 \times$ (diameter of the entire design space) defining the bold outlined box (not drawn to scale) in Fig. 3. DOT is called to optimize $\tilde{f}(x)$ within this local trust region, and returns a new candidate point at X_1 and $\rho > 0.75$. Hence X_1 is accepted and as it is towards the right boundary of the current local region, the new local

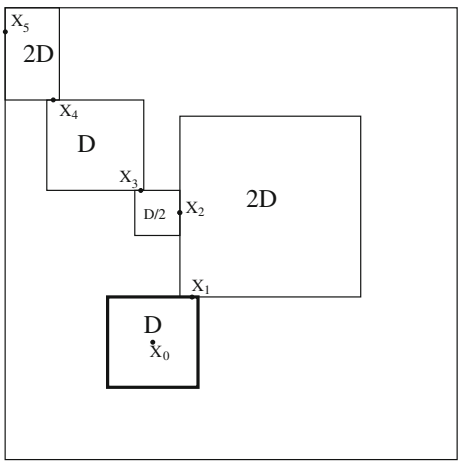

Fig. 3 A possible convergence scenario

design space is defined towards the right of the current box using $\Delta = 2D$. DOT returns the point X_2 and $0 < \rho \leq 0.25$. Hence the trust region radius is reduced to $\Delta = 2D/4 = D/2$. As X_2 is towards the center of the boundary of the local trust region, the new local design space is defined around X_2 as shown in Fig. 3. The next candidate point is at X_3, which is towards the left, and $\rho \geq 0.75$. Hence the new trust region is defined towards the left of the current box using $\Delta = D$. DOT returns a new candidate point at X_4 again towards the left of the current local trust region and with $\rho > 0.75$. Hence, the new local design space is defined towards the left of the current trust region using $\Delta = 2D$, and truncated to be within the feasible set. The algorithm converges at X_5, which is at the boundary of the entire design space for the optimization problem.

4.3 Results

The optimization algorithm DDSAO discussed in Sect. 3 was applied to the RDS model using six different surrogate construction methods from the two packages SURFPACK (POLY1, POLY2, KRIGING, and ANN) and SHEPPACK (QSHEP3D and LSHEP). Refer to [14, 35] for a detailed treatment of these methods. Each of the surrogates was constructed using three different DOEs: FF, LHS, and CCD. The optimization problem was to minimize the objective function $f(x)$ (cumulative time) using three design variables (temperature of drying gases, flow rate of inlet gases, and drum rotation speed), subject to the bound constraints $L \leq X \leq U$ where $L = (500, 1, 5)$ and $U = (600, 2, 6)$, and a start point $X_0 = (550, 1.5, 5.5)$. The optimization was performed first by linking the simulation code directly to the DOT optimizer, and then using the optimization algorithm DDSAO of Sect. 3. In both cases, all the control parameters to DOT were set to their default values except for the relative finite difference step (FDCH). While performing the optimization by having DOT directly call the simulation code, the finite difference step (FDCH) was set to 0.02. Smaller values of FDCH caused premature convergence of DOT; thus DOT's behavior is sensitive to the problem, the starting point, and various tuning parameters. While using the optimization algorithm DDSAO, FDCH was set to its default value of 0.001 for all approximation methods except QSHEP3D using the FF design and LSHEP using the CCD for which it was set to 0.02, again to avoid premature convergence. FDCH is the finite difference step size as a fraction of the design variable being perturbed and is used for internal gradient calculations by DOT. The values set for FDCH for the experiments here seem to exhibit reasonable results. While using the optimization algorithm DDSAO, all the constants initialized in the algorithm are set to their default values except for the trust region radius Δ, which was changed to 10% of the diameter of the entire design space for the FF and CCD designs. Comparing 10 versus 20%, 20% was better for LHS and 10% was better for FF and CCD. The initial trust region radius thus depends on the problem and the experimental design used.

When directly coupled with the simulation code, DOT returned the point (550.413, 2, 6), the objective function value $f(x) = 259.3$, and required 22 simulation runs ($\#f(x)$). The same experiment was performed using the optimization algorithm DDSAO for comparison with the aforementioned results. Executing the optimization algorithm DDSAO once defines a single *run*. Executing the simulation code once defines a single *simulation run*. Executing the outermost while loop from the DDSAO algorithm once defines a single SAO *iteration*. Thus, a single run of DDSAO algorithm constitutes multiple SAO iterations, and a single SAO iteration can have multiple simulation runs. A very first run of the algorithm was carried out with no data in the simulation database. Multiple runs of the algorithm were carried out using the gradually growing simulation database in order to observe the change in the total number of simulation runs required.

Tables 1, 2, and 3 report the experimental results for the optimization algorithm DDSAO using the six surrogate types constructed from the FF, LHS, and CCD designs. Each row in these tables corresponds to a single run of the optimization algorithm DDSAO, and records the true

Table 1 Full factorial (FF) results

FF	$f(x)$	$\tilde{f}(x)$	$\#f(x)$	$\#\tilde{f}(x)$
QSHEP3D	248.900	248.899	66	156
	248.800	248.800	21	27
LSHEP	248.900	248.900	47	97
	248.900	248.900	11	4
	248.800	248.800	6	10
POLY1	248.900	249.713	46	41
	248.800	250.398	11	12
POLY2	434.800	502.000	14	10
	252.400	1230.880	97	138
	248.800	247.755	37	38
	248.800	247.631	3	38
KRIGING	434.800	434.800	13	10
	410.000	410.000	12	15
	391.500	391.500	24	51
	368.300	368.300	10	10
ANN	248.900	249.474	59	67
	248.800	248.120	32	54
	248.800	248.178	9	35

$f(x)$ is true function value, $\tilde{f}(x)$ is surrogate predicted value, $\#f(x)$ is number of true function evaluations, and $\#\tilde{f}(x)$ is number of surrogate function evaluations

Table 2 Latin hypercube sampling (LHS) results, in the same format as Table 1

LHS	$f(x)$	$\tilde{f}(x)$	$\#f(x)$	$\#\tilde{f}(x)$
QSHEP3D	259.300	257.336	35	85
	259.100	259.096	43	66
	259.100	259.100	12	58
LSHEP	248.900	248.900	64	97
	248.900	248.900	10	4
POLY1	248.900	253.449	32	30
	248.900	253.050	10	4
POLY2	259.300	257.395	54	110
	259.100	259.083	33	63
	259.100	259.089	3	77
KRIGING	344.500	332.746	43	136
	279.400	277.377	11	24
	277.800	277.785	11	24
	276.300	276.174	11	21
	274.800	274.791	22	53
	268.300	265.700	22	41
	260.000	260.000	11	11
ANN	259.100	259.100	87	180
	259.100	259.092	37	49
	259.100	259.974	2	48

Table 3 Central composite design (CCD) results, in the same format as Table 1

CCD	$f(x)$	$\tilde{f}(x)$	$\#f(x)$	$\#\tilde{f}(x)$
QSHEP3D	259.200	261.381	82	127
	259.200	255.654	16	32
	259.100	259.100	38	58
	259.100	259.100	2	64
LSHEP	248.900	247.069	61	142
	248.900	248.900	25	29
	248.800	248.800	9	11
POLY1	248.900	249.753	57	41
	248.800	250.211	49	34
	248.800	250.167	9	24
POLY2	248.900	247.371	58	54
	248.900	247.772	14	4
	248.800	248.669	25	34
	248.800	248.669	1	34
KRIGING	434.800	434.800	16	14
	410.000	410.000	14	15
	391.500	391.500	12	10
	368.300	−2960.420	13	26
ANN	248.900	250.378	73	86
	248.900	249.308	10	4

function value $f(x)$, the surrogate predicted value $\tilde{f}(x)$, the number of true function evaluations $\#f(x)$, and the number of surrogate function evaluations $\#\tilde{f}(x)$. Subsequent rows for an approximation method correspond to subsequent runs of the optimization algorithm DDSAO using a cumulatively growing simulation database. The cost of the optimization is measured by the total number of true function evaluations ($\#f(x)$) needed, and the approximation quality of a surrogate is measured by $|f(x) - \tilde{f}(x)|$. The surrogate types and the DOEs used are compared in terms of both the approximation accuracy and the optimization cost.

4.4 Discussion

A generally observed trend is that the very first run of the optimization algorithm DDSAO is more expensive than having DOT directly call the simulation code. However, the results also show that the optimization algorithm DDSAO returned a better point for a large subset of runs. When directly coupled with the simulation code, DOT returned the point (550.413, 2, 6), and the objective function value $f(x) = 259.3$. The optimization algorithm DDSAO returned a point near the boundary of the entire design space with the objective function value of approximately 248 for more than 50% of the runs. This behavior is observed due to the iterative sampling nature of the SAO framework that allows the DDSAO algorithm to explore more of the design space than DOT. Consequently, DDSAO does more work but also finds a better point. Another interesting result is the significant reduction in the number of true function evaluations (simulations) for the subsequent runs with a cumulatively growing simulation database. (Occasionally, the number of function evaluations increases for a subsequent run, since the subsequent run begins with a design centered at the previous run's final point, and this new starting design and approximation can result in more work to converge than for the previous run.) A gradually maturing database increases the probability of finding a nearby data point, thereby reducing the number of expensive simulation runs. Whenever a simulation is executed, the results are stored in the database and all the subsequent runs use the previously stored simulation data. Over time, the database matures and is enriched as more and more optimizations are performed, further reducing the optimization cost.

LSHEP from SHEPPACK appears to be closely imitating the true function behavior and is the best choice overall. LSHEP outperforms all the other surrogate types in terms of approximation accuracy and optimization cost. It works equally well with all the DOEs in terms of the approximation accuracy and works best with the FF design in terms of the optimization cost.

Among the other surrogates, QSHEP3D from SHEPPACK appears to be quite competitive when constructed using the FF design. In general, QSHEP3D from SHEPPACK, and POLY1, POLY2, and ANN from SURFPACK work well with some of the DOEs and not so good with others. The response predictions obtained by using QSHEP3D, POLY2, and ANN with the LHS design, and QSHEP3D with the CCD design are as good as the response prediction obtained by having DOT directly call the simulation code.

It is well known that a polynomial surface fit may be a poor choice for modeling data trends over an entire parameter space, unless the true data trends are polynomial. The response predictions using POLY1 and POLY2 with all three experimental designs confirm this conclusion where the surrogates fail to adequately mimic the true function behavior, resulting in a poor approximation for a large subset of runs. Results show that the approximation quality is awful for the first two runs for POLY2 using the FF design. One is reminded of the fact that with an interpolating polynomial, uniform convergence is not even guaranteed for infinitely differentiable functions (recall Runge's classical example of the divergence of interpolating polynomials as the number of data points increases). Also, for a second-order polynomial a two-level FF design may not be an appropriate choice. For a second-order polynomial fit there must be at least $1 + 2k + k(k-1)/2$ distinct design points, where k is the number of design variables. For subsequent runs, POLY2 appears to work well as the simulation database matures providing enough design points for deriving reasonable second-order polynomial coefficients.

Results show that the surrogate predicted responses for KRIGING are unpredictable in general, and are worse when the surrogate is constructed with the FF and CCD designs. However, differences between the predictions and the true function behavior for KRIGING may be predominantly caused by the experimental design that is used. One of the designs more commonly used with KRIGING is Latin hypercube sampling (LHS). The results reported in Table 2 confirm these findings and show that the KRIGING method works better with a space filling experimental design like LHS. However, results from Tables 1, 2, 3 also show that the predicted response using KRIGING always turned out to be a local optimum, irrespective of the sampling approach used. In general, KRIGING (without tuning) does not seem to be competitive at all.

An abnormal behavior is observed for some of the runs where the DDSAO algorithm returns a point away from the previously found optimum (see all but the first run for POLY1 with all three DOEs, POLY2 with the LHS design, and ANN with the LHS and CCD designs). This behavior is not attributed to the surrogate type or the DOE method used, but rather is a peculiarity of the DDSAO algorithm. For each new run (except for the first run), the DDSAO algorithm starts at the previous run's optimal point, and defines an experimental design around that point. This experimental design may or may not be the same as the previous run's last experimental design, and hence, DOT might return a different candidate point. Reducing the trust region radius for the second run onwards might prevent this behavior, but this would negate the more global search nature of DDSAO.

4.5 Convergence histories

Convergence histories for the DDSAO algorithm for six different surrogates using the three DOEs are shown in Figs. 4, 5, and 6, where the true function value $f(x)$ is plotted against the optimization cost $\#f(x)$. Each curve exhibits the progress of a single run of the DDSAO algorithm through a series of SAO iterations till convergence, and the x-y coordinates for each intermediate point (shown as a geometric symbol—diamond, star, box, or circle) indicate the number of simulations and the true function value at the end of the corresponding SAO iteration. Convergence histories reveal some useful information about the performance of the DDSAO algorithm during the optimization process in terms of the number of SAO iterations required and the rate of convergence for a particular approximation method (e.g., POLY1 in Fig. 5 converges rapidly to the true optimum in four SAO iterations).

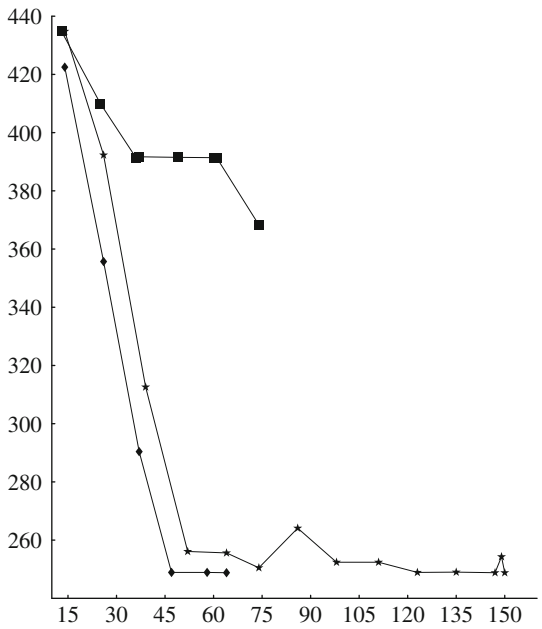

Fig. 4 Convergence profiles (objective function value $f(x)$ vs. cost $\#f(x)$) for LSHEP (*diamond*), POLY2 (*star*), and KRIGING (*box*) using FF design. The profiles for QSHEP3D, POLY1, and ANN (not shown) are similar to that for LSHEP

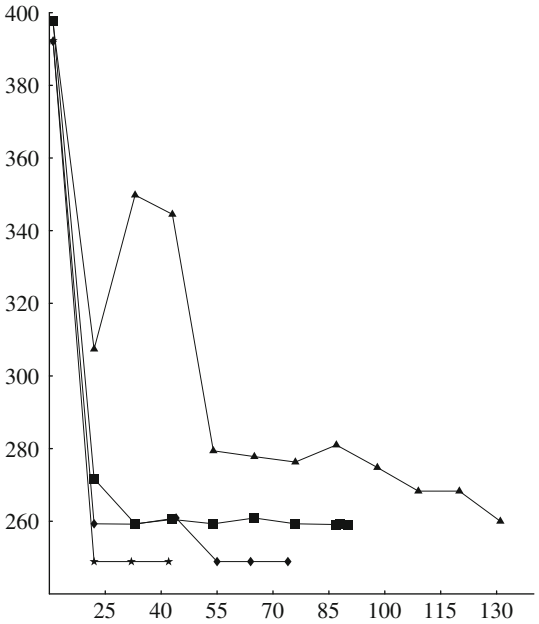

Fig. 5 Convergence profiles (objective function value $f(x)$ vs. cost $\#f(x)$) for LSHEP (*diamond*), POLY1 (*star*), POLY2 (*Box*), and KRIGING (*triangle*) using LHS design. The profiles for QSHEP3D, and ANN (not shown) are similar to that for POLY2

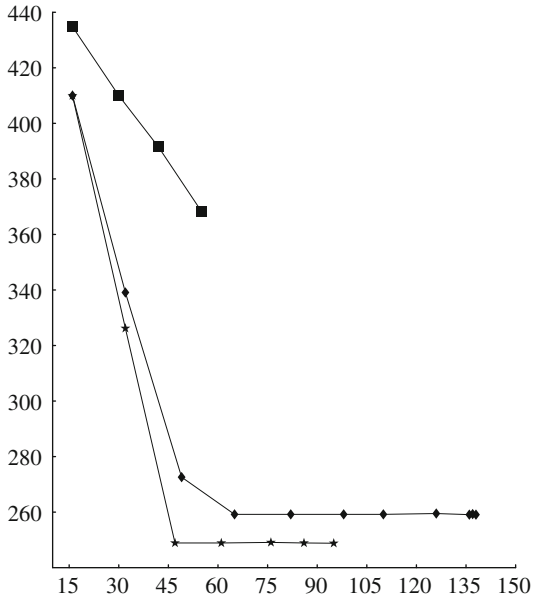

Fig. 6 Convergence profiles (objective function value $f(x)$ vs. cost $\#f(x)$) for LSHEP (*star*), QSHEP3D (*diamond*), and KRIGING (*box*) using CCD. The profiles for POLY1, POLY2, and ANN (not shown) are similar to that for LSHEP

It can be observed from all three plots that KRIGING is worse than all other surrogate construction methods, and among the three DOEs it works better with LHS. The convergence profiles for QSHEP3D, POLY1, and ANN in Fig. 4, and POLY1, POLY2, and ANN in Fig. 6 are similar to those for LSHEP in the respective plots, however, a closer examination of the Tables 1 and 3 shows that the approximation accuracy for some of these methods is considerably worse than that for LSHEP.

4.6 Other issues

The computer-based simulation models are deterministic in nature where the response values are not random variables but are determined by the underlying mathematical models. The RDS model under consideration here is one such deterministic model. An important issue in the analysis of data from a deterministic computer experiment as discussed in [18] is that many of the usual statistical techniques cannot be directly applied because of the lack of a random error component. It is observed that the full factorial (FF) design is a more appropriate choice of DOE for LSHEP, QSHEP3D, POLY1, and ANN; the Latin hypercube sampling (LHS) for KRIGING, and the central composite design (CCD) for POLY2. Even though these DOEs appear to be compatible with the approximation methodologies discussed, there are many classes of experimental designs (orthogonal arrays, Box-Behnken design, Koshal design, small composite design, etc.) in the literature that are worth trying. A more sophisticated choice of the experimental design may provide more insight. Orthogonal arrays are widely used for data sampling in many large multidisciplinary design optimization (MDO) problems and might be useful for the implementation of the proposed approach for more complex models in WBCSim. Another sampling approach known as optimization-based sampling has proved to be more efficient in driving the optimization in a SAO framework [24].

Numerous sophisticated techniques are available to build RSAs, such as radial basis functions (RBF), smoothing splines, etc., that are worth trying. The RDS model that has been used as a testbed is one of the simplest models in WBCSim, and although the proposed methodology appears to be competitive for the chosen model, one must extend the current study to different MDO problems having large dimensionality and complexity, e.g., the hot pressing model in WBCSim, in order to conclude much more.

There is a limitation to the optimization algorithm DDSAO. In general, a true optimum value cannot be guaranteed when the algorithm converges. Methods like defining a final trust region around the last candidate point to verify the true optimality do not work without mathematical assumptions about the objective function and its gradient.

5 Conclusions

A data driven, surrogate-based optimization algorithm DDSAO was applied to the simulation code of the RDS model in WBCSim. Although a RDS simulation is relatively cheap (545 ms), the DDSAO methodology extends to very expensive simulation models (e.g., the two hour HC simulation in WBCSim), where exploiting an existing database of previous analyses can be imperative. Six different approximation algorithms from the two packages SURFPACK and SHEPPACK were used to build a surrogate using three DOEs: full factorial (FF), Latin hypercube sampling (LHS), and central composite design (CCD). Results show that the RSAs constructed using design of experiments can be effectively managed by a SAO framework based on a trust region strategy. A generally observed trend is that the very first run of the optimization algorithm DDSAO is more expensive than having DOT directly call the simulation code. However, results also show that the optimization algorithm DDSAO returned a better point for a large subset of runs. This behavior derives from the iterative sampling nature of the SAO framework that allows the DDSAO algorithm to explore more of the design space than DOT. Consequently, DDSAO does more work but also finds a better point. Another interesting result is the significant reduction in the number of simulations (exact function evaluations) for the subsequent runs with a cumulatively growing simulation database. Whenever a simulation is executed, the results are stored in the database and all the subsequent runs use the previously stored simulation data. Over time, the database matures and is enriched as more and more optimizations are performed, further reducing the optimization cost.

While using the optimization algorithm DDSAO, all the constants initialized in the algorithm are set to their default values except for the initial trust region radius Δ, which was changed to 10% of the diameter of the entire design space for the FF and CCD designs. Comparing 10 versus 20%, 20% was better for LHS and 10% was better for FF and CCD. The initial trust region radius thus depends on the problem and the experimental designs used. However, additional computational experiments should be performed in order to explore the potential effect of variations in the initial trust region radius on the results.

Of the six approximation types used to build a surrogate, LSHEP from SHEPPACK appears to be the best choice in terms of approximation accuracy and optimization cost. It is observed that the full factorial (FF) design is a more appropriate choice of DOE for LSHEP, QSHEP3D, POLY1, and ANN; the Latin hypercube sampling (LHS) for KRIGING, and the central composite design (CCD) for POLY2. Although the proposed methodology appears to be competitive for the chosen RDS model, one must extend the current study to different MDO problems having large dimensionality and complexity, e.g., the hot pressing model in WBCSim, in order to conclude much more.

Acknowledgments This work was supported in part by Department of Energy Grant DE-FG02-06ER25720, Air Force Research Laboratory Grant FA8650-09-2-3938, National Science Foundation Grant CCF-0726763, and Air Force Office of Scientific Research Grant FA9550-09-1-0153.

References

1. Allen NA, Shaffer CA, Vass MT, Ramakrishnan N, Watson LT (2003) Improving the development process for eukaryotic cell cycle models with a modeling support environment. Simulation 79:674–688
2. Ames AL, Nadeau DR, Moreland JL (1996) VRML 2.0 Sourcebook, 2nd edn. Wiley, New York, pp 241–295
3. Burnett T, Chaput C, Arrighi H, Norris J, Suson DJ (2000) Simulating the Glast satellite with Gismo. IEEE Comput Sci Eng 2:9–18
4. Bramley R, Gannon D, Stuckey T, Villacis J, Akman E, Balasubramanian J, Breg F, Diwan S, Govindaraju M (1998) The linear system analyzer, technical Report TR-511. Department of Computer Science, Indiana University, Bloomington, IN
5. Boisvert RF, Rice JR (1985) Solving elliptic problems using ELLPACK. Springer, New York
6. Chen JX, Fu X (1999) Integrating physics-based computing and visualization: modeling dust behavior. IEEE Comput Sci Eng 1:12–16
7. Dymond R, Lohani V, Kibler D, Bosch D, Rubin EJ, Dietz R, Chanat J, Speir C, Shaffer CA, Ramakrishnan N, Watson LT (2003) From landscapes to waterscapes: a PSE for landuse change analysis. Eng Comput 19:9–25
8. Eldred MS, Hart WE (1998) Design and implementation of multilevel parallel optimization on the intel teraflops. Multidiscip Anal Optim 98:44–54
9. Gallopoulos E, Houstis E, Rice JR (1994) Computer as thinker/doer: problem solving environments for computational science. IEEE Comput Sci Eng 1:11–23
10. Goel A, Phanouriou C, Kamke FA, Ribbens CJ, Shaffer CA, Watson LT (1999) WBCSim: a prototype problem solving environment for wood-based composites simulations. Eng Comput 15:198–210
11. Guisset P, Tzannetakis N (1997) Numerical methods for modeling and optimization of noise emission applications. In: Proceedings of the ASME international mechanical engineering congress and exposition, ASME FAIRFIELD, NJ, (USA), vol 24, pp 315–322
12. Goel A, Baker CA, Shaffer CA, Grossman B, Mason WH, Watson LT, Haftka RT (2001) VizCraft: a problem solving environment for aircraft configuration design. IEEE Comput Sci Eng 3:56–66
13. Giunta AA, Richards MD, Cyr EC, Swiler LP, Brown SL, Eldred MS (2006) Surfpack version 1.0 user's manual. Sandia National Laboratories, Albuquerque
14. Giunta AA, Swiler LP, Brown SL, Eldred MS, Richards MD, Cyr EC (2006) The Surfpack software library for surrogate modeling of sparse irregularly spaced multidimensional data. In: 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, AIAA 1708-1736, Portsmouth, VA
15. Houstis E, Gallopoulos E, Bramley R, Rice JR (1997) Problem solving environments for computational science. IEEE Comput Sci Eng 4:18–21

16. Kamke FA, Wilson JB (1985) Computer simulation of a rotary dryer: retention time. Am Inst Chem Eng J 32:263–268
17. Kamke FA, Wilson JB (1985) Computer simulation of a rotary dryer: heat and mass transfer. Am Inst Chem Eng J 32:269–275
18. Myers RH, Montgomery DC (1995) Response surface methodology, process and product optimization using designed experiments. Wiley, New York
19. Mishra D, Shaffer CA, Ramakrishnan N, Watson LT, Bae KK, He J, Verstak A, Tranter WH (2007) S^4W: a problem solving environment for wireless system design. Softw Pract Exp 37:1539–1558
20. Pérez VM, Renaud JE, Watson LT (2001) Adaptive experimental design for construction of response surface approximation. AIAA 40:2495–2503
21. Pérez VM, Renaud JE, Watson LT (2008) Reduced sampling for construction of quadratic response surface approximations using adaptive experimental design. Eng Comput 25:764–782
22. Pérez VM, Renaud JE, Gano SE (2000) Constructing variable fidelity response surface approximations in the usable feasible region. In: Proceedings of the 8th AIAA/NASA/USAF multidisciplinary analysis & optimization symposium, AIAA 2000-4888, Long Beach, CA
23. Pérez VM, Renaud JE (2000) Decoupling the design sampling region from the trust region in approximate optimization. In: Proceedings of the international mechanical engineering congress and exposition, American Society of Mechanical Engineers, vol 63, pp 205–214
24. Rodríguez JF, Perez VM, Padmanabhan D, Renaud JE (2001) Sequential approximate optimization using variable fidelity response surface approximations. Struct Multidiscip Optim 22:24–34
25. Rodríguez JF, Renaud JE, Watson LT (1998) Trust region augmented Lagrangian methods for sequential response surface approximation and optimization. J Mech Design 120:58–66
26. Resnik J, Kamke FA (1998) Modeling the cure of adhesive-wood bonds using high frequency energy. Final Report, U.S.-Slovene Joint Board on Scientific and Technological Cooperation, Project 95-AES10. University of Ljubljana, Ljubljana
27. Rodríguez JF, Renaud JE, Watson LT (1998) Convergence of trust region augmented Lagrangian methods using variable fidelity approximation data. Struct Optim 15:141–156
28. Renaud JE, Gabriele GA (1994) Approximation in nonhierarchic system optimization. AIAA J 32:198–205
29. Shu J, Watson LT, Ramakrishnan N, Kamke FA, Zombori BG (2004) An experiment management component for the WBCSim problem solving environment. Adv Eng Softw 35:115–123
30. Shu J, Watson LT, Ramakrishnan N, Kamke FA, North C (2008) Unification of problem solving environment implementation layers with XML. Adv Eng Softw 39:189–201
31. Shu J, Watson LT, Zombori BG, Kamke FA (2006) WBCSim: an environment for modeling wood-based composites manufacture. Eng Comput 21:259–271
32. Sioson A, Watkinson JI, Vasquez-Robinet C, Ellis M, Shukla M, Kumar D, Ramakrishnan N, Heath LS, Grene, R, Chevone BI, Kafadar K, Watson LT (2003) Expresso and chips: creating a next generation microarray experiment management system. In: Proceedings of the next generation software workshop, Los Alamitos, California, USA
33. Skidmore R, Verstak A, Ramakrishnan N, Rappaport TS, Watson LT, He J, Varadarajan S, Shaffer CA, Chen J, Bae KK, Jiang J, Tranter WH (2004) Towards integrated PSEs for wireless communications: experiences with the S^4W and SitePlanner projects. ACM SIGMOBILE Mobile Comput Commun Rev 8:20–34
34. Tong SS, Powell D, Goel S (1992) Integration of artificial intelligence and numerical optimization techniques for the design of complex aerospace systems. In: Proceedings of the aerospace design conference, AIAA 92-1189, Irvine, CA
35. Thacker WI, Zhang J, Watson LT, Birch JB, Iyer MA, Barry MW (2009) Algorithm XXX: SHEPPACK: modified Shepard algorithm for interpolation of scattered multivariate data. Technical Report TR 09-13, Department of Computer Science, Virginia Polytechnic Institute & State University, Blacksburg, VA 24061
36. Vanderplaats Research and Development, Inc. (1985) DOT users manual. Version 4.20, Colorado Springs, CO
37. Vass M, Allen N, Shaffer CA, Ramakrishnan N, Watson LT, Tyson JJ (2004) The JigCell model builder and run manager. Bioinformatics 20:3680–3681
38. Vass M, Shaffer CA, Ramakrishnan N, Watson LT, Tyson JJ (2006) The JigCell model builder: a spreadsheet interface for creating biochemical reaction network models. IEEE/ACM Trans Comput Biol Bioinforma 3:155–164
39. Watson LT, Lohani VK, Kibler DF, Dymond RL, Ramakrishnan, N, Shaffer CA (2002) Integrated computing environments for watershed management. J Comput Civil Eng 16:259–268
40. Wolfram S (1996) The mathematica book, 3rd edn. Wolfram Media/Cambridge University Press, Champaign
41. Wujek BA, Renaud JE (1998) New adaptive move-limit management strategy for approximate optimization, part 1. AIAA J 36:1911–1921
42. Wujek BA, Renaud JE (1998) New adaptive move-limit management strategy for approximate optimization, part 2. AIAA J 36:1922–1937
43. Zombori BG, Kamke FA, Watson LT (2001) Simulation of the mat formation process. Wood Fiber Sci 33:564–579
44. Zombori BG, Kamke FA, Watson LT (2003) Simulation of internal conditions during the hot-pressing process. Wood Fiber Sci 35:2–23
45. Zombori BG, Kamke FA, Watson LT (2004) Sensitivity analysis of internal mat environment during hot-pressing. Wood Fiber Sci 36:195–209