INL/CON-14-33954
PREPRINT

A New Hybrid Genetic Algorithm for Optimizing the Single and Multivariate Objective Functions

2015 American Society of Agricultural and Biological Engineers (ASABE) Annual International Meeting

Jaya Shankar Tumuluru and Richard McCulloch

July 2015

2950 Niles Road, St. Joseph, MI 49085-9659, USA
269.429.0300 fax 269.429.3852 hq@asabe.org www.asabe.org

An ASABE Meeting Presentation

DOI:10.13031/aim.20152188606

Paper Number: 152188606

A new hybrid genetic algorithm for optimizing the single and multivariate objective functions

Jaya Shankar Tumuluru and Richard McCulloch

750 University Blvd, Biofuels and Renewable Energy Technologies, Idaho National Laboratory, Idaho Falls, Idaho-3750-83415

E mail: JayaShankar.Tumuluru@inl.gov

Written for presentation at the

2015 ASABE Annual International Meeting

Sponsored by ASABE

New Orleans, Louisiana

July 26 – 29, 2015

Abstract.

A new hybrid genetic algorithm was developed which combines a stochastic evolutionary algorithm with a deterministic adaptive step steepest descent hill climbing algorithm in order to optimize complex multivariate problems. By combining both algorithms computational resources are conserved and the solution converges rapidly as compared to either algorithm alone. In genetic algorithms natural selection is mimicked by random events such as breeding and mutation. In the adaptive step steepest descent algorithm the solution moves toward the lowest surrounding point. Step sizes start big and get progressively smaller, increasing computational efficiency. The genetic algorithm ensures the solution samples the entire global search space, thus a global minimum is found. The steepest descent method fine tunes the solution by moving it to the nearest local minimum. The code was developed, including a graphical user interface, in MATLAB. Additional features such as bounding the input, weighting the objective functions individually, and constraining the output are also built into the interface. The algorithm developed was used to optimize the response surface models which use process variables (feedstock moisture content, die speed, and preheating temperature) to predict pellet properties (pellet moisture content, unit, bulk and tapped density, durability, and specific energy consumption). The solution found by the hybrid algorithm was validated experimentally. Execution times were decreased by approximately 40%, based on 1,0000 trials with each method, using the new hybrid algorithm as compared to using a genetic algorithm alone with the same parameters, both developed at INL. Performance of the hybrid algorithm versus the commercial Matlab genetic algorithm is investigated. Results show that the hybrid genetic algorithm converged to the global maximum for bulk density in one iteration, whereas the commercial genetic algorithm took twenty nine iterations to converge.

Keywords: *Genetic algorithm, gradient search, single and multi-objective functions, optimization*

Introduction

A genetic algorithm is a stochastic optimization method which is based on the phenomenon of natural evolution (Simon, 2013; Goldberg 1989). Pragmatic researchers see evolution's remarkable power as something to be emulated. Natural selection eliminates one of the greatest hurdles in software design, i.e. specifying in advance all the features of a problem and actions a program should take to deal with them. By harnessing the mechanisms of evolution, researchers will be able to "breed" programs that solve problems even when no person can fully understand their structure. GAs makes it possible to explore a far greater number of potential solutions to a problem than do conventional programs (Holland, 1992). GAs are different from the normal optimization techniques and search procedures 1) GAs work with a coding of the parameters set, not the parameters themselves, 2) GAs search from a population of points, not a single point, 3) GAs use payoff (objective function) information, not derivatives or the other auxiliary knowledge. Figure 1 indicates flow diagram for a regular genetic algorithm used in solving the optimization problems.

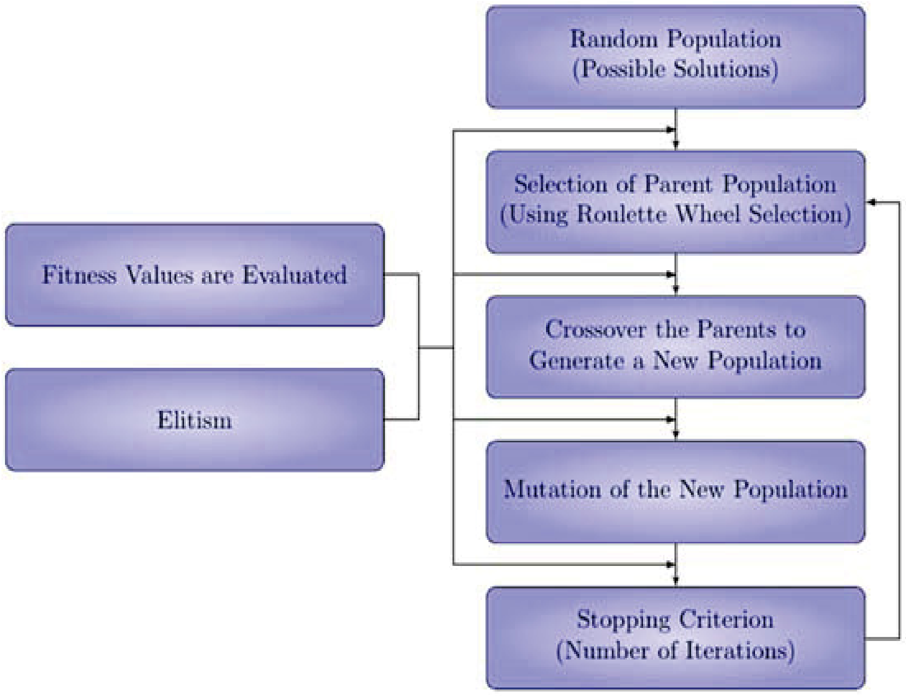

Random Population (Possible Solutions)
Selection of Parent Population (Using Roulette Wheel Selection)
Fitness Values are Evaluated
Crossover the Parents to Generate a New Population
Elitism
Mutation of the New Population
Stopping Criterion (Number of Iterations)

Figure 1: Flowchart of a regular genetic algorithm

Advantages and Disadvantages of Genetic Algorithms

The main advantages of GAs are 1) GA-based approaches are capable of finding a number of optimal solutions rather than a single solution (Kalyanmoy, 2000), 2) GA-based approaches are capable of exploring the search space more thoroughly with a smaller number performances evaluations than those based on local search, such as simulated annealing and tabu search (April et al, 2003) and 3) GA-based approaches are less dependent on the good selection of the starting points, and they don't require neighborhood definition (April et al, 2003), 4) It can solve every optimization problem which can be described with the chromosome encoding 5) It solves problems with multiple solutions 6) Since the genetic algorithm execution technique is not dependent on the error surface, we can solve multi-dimensional, non-differential, non-continuous, and even non-parametrical problems 7) Structural genetic algorithm gives us the possibility to solve the solution structure and solution parameter problems at the same time by means of genetic algorithm 8) Genetic algorithm is a method which is very easy to understand and it practically does not demand the knowledge of mathematics and 9) Genetic algorithms are easily transferred to existing simulations and models. A simple genetic algorithm with operators like reproduction, crossover and mutation yields goods results in practical optimization problems. The GAs utilizes the stochastic operations like crossover and mutation on a population to make a change of generation. Crossover combines substructures of parents to produce new individuals. Crossover is the most characteristic operation of GA which is not used in other global search methods such as simulated annealing. Mutation is another operation which helps the algorithm to move to a global search space which prevents the solution from getting stuck at local space (Simon, 2013). Shankar and Sokhansanj (2010) worked on understanding the effect of crossover and mutation rate on function convergence. Not much work is done on understanding the effect of the genetic algorithm operators like crossover and mutation, elitism on the function convergence.

Like other artificial intelligence techniques, the genetic algorithm cannot assure constant optimization response times. Even more, the difference between the shortest and the longest optimization response time is much larger than with conventional gradient methods. This unfortunate genetic algorithm property limits the genetic algorithm's use in real time applications. Genetic algorithm applications in controls which are performed in real time are limited because of random solutions and convergence, in other words this means that the entire population is improving, but this could not be said for an individual within this population. Therefore, it is unreasonable to use genetic algorithms for on-line controls in real systems without testing them first on a simulation model. Certain optimization problems (they are called variant problems) cannot be solved by means of genetic algorithms. This occurs due to poorly known fitness functions which generate bad chromosome blocks in spite of the fact that only good chromosome blocks cross-over. There is no absolute assurance that a genetic algorithm will find a global optimum. It happens very often when the populations have a lot of subjects. Figure 2 indicates the possibility how local and global search algorithms can get struck at complex global search spaces. Hybridization of genetic algorithm with gradient based search methods can help to overcome some of the limitations specific to genetic algorithm while solving complex optimization problems. The hybridization can help to improve the solution search space with every iteration thereby reducing the computation time.

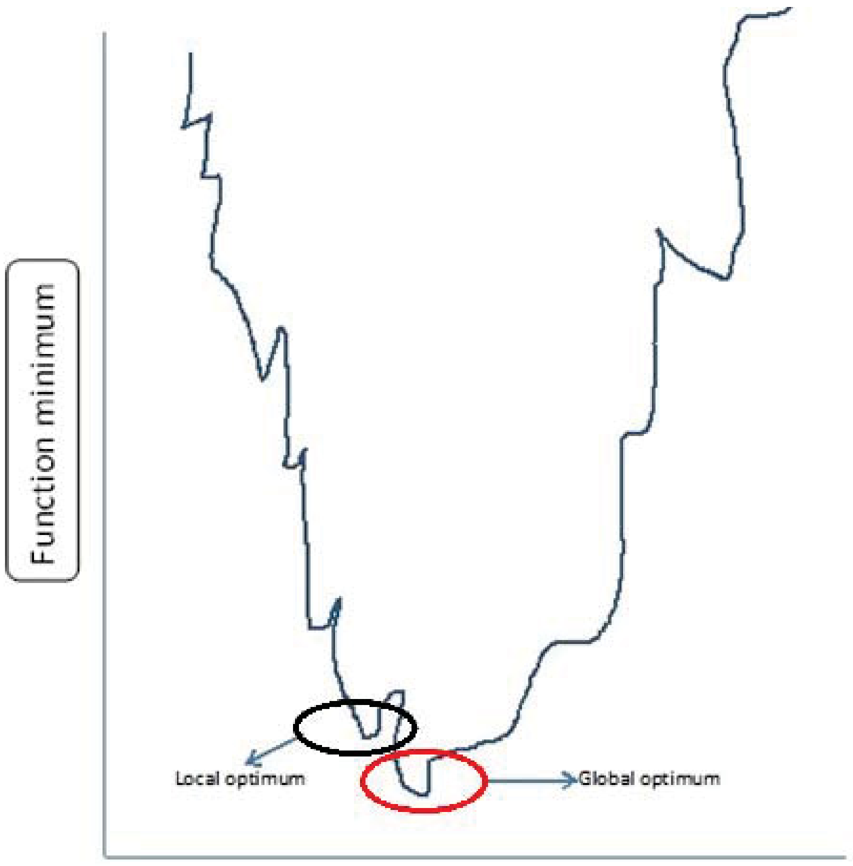


Figure 2: Example of local and global optimum points in function minimization problem

Gradient Search Method

Gradient descent is a first-order optimization algorithm (Chapra, 2015). To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point. If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function; the procedure is then known as gradient ascent. Gradient descent is also known as steepest descent, or the method of steepest descent. When known as the latter, gradient descent should not be confused with the method of steepest descent for approximating integrals (Petrova, 1997). In recent years, sampling-based planning algorithms have met with widespread success due to their ability to rapidly discover the connectivity of high-dimensional configuration spaces. Planners such as the Probabilistic Roadmap (PRM) and Rapidly-Exploring Random Tree (RRT) algorithms, along with their descendants, are now used in a multitude of robotic applications (Kavraki et al., 1996; Kuffiner et al., 2000). Both algorithms are typically deployed as part of a two-phase process: first find a feasible path, and then optimize it to remove redundant or jerky motion.

Objective

Upon developing the regression models for biomass pellet properties the need arose to find the optimum set of input parameters in order to obtain the most desirable output properties for multivariable optimization problems. While many tools and algorithms have been created for this purpose there was not software available that could implement a rigorous stochastic search coupled with deterministic local optimization while still constraining the output and assigning weights to the individual objective functions. In the search for a tool with all of these capabilities and features we found none, thus it became expedient to create such a tool. Upon creation of the program a user had to be a profound numerical modeler in order to use or modify the program input/output, thus a user interface was created to facilitate the use of the tool in the hands of various users who have very little background to single and multi-variable optimization techniques. While the tool has been very useful in this work and has supported other research efforts, there are more improvements to be made, which will be discussed in a future section.

The overall objective of the present study was to develop a new algorithm which uses both evolutionary and gradient search method properties to overcome the limitations of optimum solution getting stuck in a local search space in case of complex optimization problems. The specific objectives of the present work are 1) develop single and multi-variable optimization software using a hybrid genetic algorithm which can handle single and multi-variable optimization problems, 2) develop a graphical user interface using the Matlab platform and 3) provide flexibility to change the algorithm operators like crossover, mutation, iteration, elitism, population size, tolerance persistence, goal, weights, lower and upper constrains for the solution space and three dimensional surface plot for the functions tested.

Materials and Methods

Hybrid Genetic Algorithm (HGA) Tool

To solve the multi-objective optimization problem a hybrid genetic algorithm (HGA) was developed. The working principle of Genetic Algorithms (GAs) is based on Darwin's theory of survival of the fittest (Kalyanmoy, 2000; Davis, 1991 and Holland, 1992). HGA uses both the genetic algorithm and steepest ascent hill climbing methods to reach an optimum solution. In the case of complex optimization problems, the hybrid genetic algorithm will help to prevent the solution from getting stuck at local optimum points. Figure 2 depicts the global and local optimum points in the function minimization problem. The hill climbing subroutine added to the regular genetic algorithm routine helps to move local optimum conditions into the global search space.

Working Principles of the Hybrid Genetic Algorithm (HGA)

In GA analysis a random population is initially generated, which is made up of potential candidate solutions for the objective function. The individuals in the population are represented by a string of symbols called chromosomes. Binary bit strings are used to represent the chromosomes (Davis, 1991; Tumuluru et al., 2013). The length of the chromosome is the length of one complete candidate solution, ie, the number of independent variables defining the objective function. Each variable in the chromosome is typically referred to as an allele. The chromosome selection is based on the goodness of a chromosome in the population and is evaluated over a fitness function, the goodness being the solution with the greatest magnitude in the direction of desired optimum (maximum or minimum). The parent population which is used to breed the new generation is selected based on the Roulette Wheel selection technique. In Roulette Wheel selection, chromosomes with the best fitness values have greater chances of being selected, whereas the worse candidates are more likely to be eliminated (Lipowski, 2012). The crossover operation generates a new offspring population based on the Roulette wheel selection. Single point crossover defines a fixed number of alleles contributed from each parent. The mutation operation, which is a bit inversion process, helps to 1) maintain the diversity within the population, 2) inhibit the premature convergence of the population, and 3) prevent the population from getting stuck at a local optimum. As is the case in nature, mutation occurs by randomly choosing a chromosome and completely changing the alleles, thus creating a new random candidate solution. Elitism is built into the model as well. With elitism the optimum candidate solutions are preserved. This means that mutation will not affect the best chromosome. Elitism also ensures that the optimum solution will carry on to the offspring generation unchanged, thus the solution will not regress, but can only improve with each iteration. After the crossover has taken place, the offspring generation has the possibility of some random mutations occurring, and with the best chromosome preserved, the steepest ascent method is applied to all chromosomes. In the steepest ascent method a Laplacian operator is applied to the chromosome to determine the response to perturbation in each allele value. The allele that shows the most improvement in moving toward the optimum value is incremented and the Laplacian is recalculated from the new chromosome position. This process is repeated until there is no beneficial change in any gradient direction, at which point the step size is reduced and the movement process repeated. This is known as adaptive stepping. In adaptive stepping movement stops when the step size reaches a specified tolerance.

The flow diagram for the hybrid genetic algorithm (HGA) developed is given in Figure 3. The hybrid GA program was developed on the Matlab platform with an accompanying graphical user interface (GUI). A user manual was written for the software developed to explain in detail regarding how to work with the various operators like elitism, crossover, mutation, tolerance, persistence, bounding the optimization problem within the constraints, and providing weights to the functions to be optimized. The manual also has some typical examples of the functions solved using the new optimization software.

While hybrid genetic algorithms have gained some attention in recent years (Hart, 1994; El-Mihoub, 2006) there still does not exist available software for multi-objective optimization that can constrain the output and apply weights to individual objective functions. What's more, this work has not been applied to biomass densification as shown in this work, in which the material properties or objective functions are highly correlated to the independent variables and to each other. These differences make the work unique and necessary.

To illustrate the advantage of the hybrid GA consider the curve shown in Figure 2. Imagine placing thumb tacks on the curve at random spots. Now use the lowest tack to determine where to re-pin the highest tacks. Eventually you will successfully place a tack on the lowest part of the curve. Now imagine randomly placing marbles instead of thumb tacks. Again, the lowest marble is used to determine where the higher marbles are placed again on the curve, but with each placement the marble sinks to the lowest spot available. In essence for a hybrid GA the placement is governed by natural selection, where the best is more likely to determine the placement of new candidates, and the rolling of each marble is analogous to the local search routine that takes each candidate and places it at the nearest extreme using local gradient information.

Algorithm Parameters for Single and Multivariable Optimization Problems

Population: Population defines the number of candidate solutions to consider for each generation.

Elitism: Defines the top percentage of parent solutions to transfer to the child generation.

Crossover: Defines the percentage of the child population to generate from breeding from the parent generation. The remaining child population is copied directly from the parent generation. Parents to be copied are selected using the Roulette Wheel probability method. Parents that result in fitness values closer to the goal (maximum or minimum) are more likely to be copied or used as parents.

Mutation: Defines the percentage of the child population to mutate. The alleles of the chosen children are completely randomized. The most elite or fit solution is not a candidate for mutation.

Iterations: Is the number of generations to create.

Lower and Upper Constraints: The lower and upper constraints apply a bound to the results. Constraints are imposed during the roulette wheel selection subroutine. A given constraint is assigned a weight value, which is used to weight the deviation from the bounds of the results. Using this approach a user can decide which objective functions are more important to obtain within the given bounds. Note: the weight values for the constraints are not the same weight values for the objective functions.

Tolerance: The tolerance determines which solutions are returned as possible answers. If a candidate has a fitness that is within a certain distance from the optimum solution it is included in the solution set.

Bounds: Lower bound and upper bound are vectors that define the limit for the independent variables. In the single objective example these define the limits for x as

Lower bound $<$ x $<$ Upper bound

For the multi-objective optimization example the limits are defined using bounding vectors as

$$LB_1 \leq x_1 \leq UB_1$$

$$LB_2 \leq x_2 \leq UB_2$$

$$LB_3 \leq x_3 \leq UB_3$$

Where the lower bound = [LB_1 LB_2 LB_3] and the upper bound = [UB_1 UB_2 UB_3].

Goal: The goal defines whether to maximize or minimize the fitness function(s).

Weights: Weights can be assigned to the output of the functions to define importance values for each function in reference to the others.

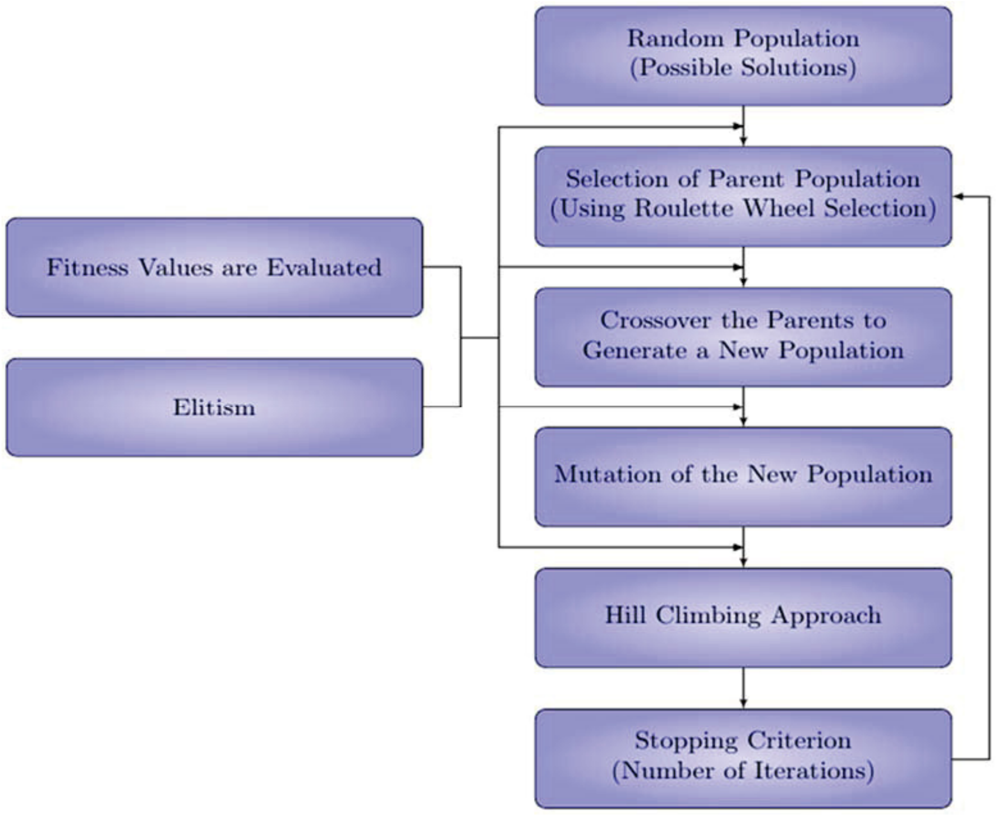

Random Population
(Possible Solutions)
Selection of Parent Population
(Using Roulette Wheel Selection)
Fitness Values are Evaluated
Crossover the Parents to
Generate a New Population
Elitism
Mutation of the New Population
Hill Climbing Approach
Stopping Criterion
(Number of Iterations)

Figure 3: Flow diagram of the hybrid genetic algorithm (HGA) developed at Idaho National Laboratory

Architecture for the Simulation

For all of the simulations in this work simulations were performed on an Asus X551M Laptop using a 64 bit Windows 8 operating system with 4 Gb installed physical RAM and an Intel Celeron CPU with a 2.16 GHz clock speed.

Pelleting Process

Biomass Feedstock

Corn stover harvested from farms in Iowa, Boone, in autumn of 2011 was further baled for transportation. At the Idaho National Laboratory the baled corn stover was further size reduced using a 50.8 mm screen size in a Vermeer HG200 grinder (Vermeer Corporation-Agriculture, Pella, IA, USA). These bigger size corn stover particles were further size reduced to smaller grinds using a hammer mill (Bliss Eliminator Hammer mill, model E-4424-TF, manufactured by Bliss Industries, Ponca City, OK, USA) fitted with 4.8 mm screen size (Tumuluru, 2014). The ground corn stover was stored in air-tight containers and was measured for properties such as moisture, bulk, and tapped densities. Further, the moisture content of the corn stover was adjusted to the desired moisture levels based on the experimental design.

High Moisture Pelleting Process (HMPP)

About 3 kg of corn stover, with calculated amounts of water, were mixed in a ribbon blender (Model: RB 500, Colorado Mill Equipment, Canon City, CO, USA), to adjust the moisture content to desired levels based on the experimental design. The moisture-adjusted corn stover was stored overnight in a cold storage unit set at approximately 4 °C (Tumuluru, 2014 & 2015). The moisture-adjusted corn stover was then loaded into the feeder hopper of the pellet mill, where it was preheated for about 4-5 min at different temperatures based on experimental design. The preheated biomass was fed continuously using the feeder, and care was taken that there were no flow irregularities inside the pellet mill (Tumuluru, 2014 & 2015). The pellets produced were then cooled using a horizontal cooler to reduce the pellet moisture content. The moisture content of the pellets after cooling was measured to determine how much moisture is lost due to preheating, pelleting, and cooling. As the moisture in the pellets after the cooling step was found to be high, pellets were further dried in a mechanical oven at 70 °C for about 2–4 hours to reduce the moisture content to about 7–9 % (w.b.) for safe storage without any microbial degradation. These dried pellets were immediately further analyzed for physical properties such as unit, bulk, and tapped density, and durability (Tumuluru, 2014 & 2015).

Experimental Conditions and Design

All of the pelleting experiments were conducted at one particle size (i.e., biomass ground using Bliss Eliminator Hammer mill, fitted with a 4.8 mm screen) using an 8 mm flat-die pellet mill. The process variables selected were feedstock moisture content (28-38%), die speed (40-60 Hz) and preheating temperature (30-110°C). Table 1 indicates the coded and actual levels of the process variables used in the experimental design. An extended Box and Behnken design was used to generate the experimental data for the physical properties of the pellets produced and specific energy consumption of the high moisture pelleting process. Tumuluru (2014) has discussed in detail the process for making pellets from high moisture corn stover. The experimental data obtained was used to develop the response surface models.

Table 1: Experimental conditions used for high moisture pelleting studies

Process Conditions	Coded Levels		
	-1	0	+1
Feedstock Moisture Content (%, w.b.)	28	33	38
Die Speed (Hz)	40	50	60
Preheating Temperature (°C)	30	70	110

Pellet Properties Measurement

Pellet Moisture, Unit and Bulk Density

The moisture content of raw and pelleted corn stover was measured by drying 50 g samples in a heated convection oven set at 105 °C for 24 hours following ASABE Standards, (ASABE Standards, 2007). The bulk density of pellets produced was determined based on ASABE Standard S269.4 (ASABE Standards, 2007).

Durability

Durability of pellets was determined by tumbling the test sample at 50 rpm for 10 min, in a dust-tight enclosure.

$$\text{Durability} = \frac{\text{Mass of pellets after tumbling}}{\text{Mass of pellets before tumbling}} \times 100 \qquad (1)$$

Specific Energy Consumption (kWhr/ton)

The power demand in kilowatts is calculated using (2) provided by the variable frequency drive (VFD) vendor for the pellet mill. This equation calculates kW based on the % motor power consumed. Vendor provided calculation for power (kW):

$$\text{Power (kW)} = \frac{\left(\frac{\%\text{ motor power}}{100}\right) \times (\text{Nominal size of the motor horse power}) \times 746\ \text{watts/HP} + 114}{1000} \qquad (2)$$

- % motor power = Data recorded using labview software during running of the pellet mill
- Nominal size of the pellet mill motor = 10HP;
- Approximate power losses of the VFD=114 W

$$\text{Specific energy consumption} = \frac{\left(\text{Full load power (kW)} - \text{No load power (kW)}\right) * \text{time (hr)}}{\text{weight of pelleted material (kg)}} = \frac{\text{kWhr}}{\text{kg}} \qquad (3)$$

Note: The pellet mill was run at 40, 50 and 60 Hz speeds and with no load to record the power (kW).

Objective Functions

The objective functions used were the response surface models developed for bulk density (BD), durability (D) and specific energy consumption (SEC), pellet moisture content (%, w.b.), (Table 2). The hybrid genetic algorithm (HGA) was used to create a multi-objective GA solver which optimizes different functions simultaneously. The program developed can simultaneously optimize any number of objective functions for predicting the desirable process conditions for either maximization or minimization of objective functions.

Table 2: Response surface models developed for pellet properties and energy consumption

Physical Property	Model	R^2
Bulk Density (kg/m^3)	$-708.139 + 130.481x_1 - 34.082x_2 + 3.375x_3 - 2.651x_1^2 + 0.1117x_2^2 - 0.01199x_3^2 + 0.7265x_1x_2 - 0.0681x_1x_3 + 0.00819x_2x_3$	0.85
Durability (%)	$26.26884 + 6.994x_1 - 1.16322x_2 - 0.11594x_3 - 0.13075x_1^2 + 0.00878x_2^2 - 0.00063x_3^2 + 0.00920x_1x_2 + 0.00672x_1x_3 - 0.00015x_2x_3$	0.83
Specific Energy Consumption (kWhr/ton)	$1744.6165 - 69.2999x_1 - 10.9941x_2 - 2.3827x_3 + 1.22397x_1^2 + 0.147777x_2^2 - 0.00664x_3^2 - 0.31698x_1x_2 + 0.0391444x_1x_3 + 0.0416159x_2x_3$	0.86
Pellet Moisture Content (%, w.b.)	$-12.9754 + 0.5385x_1 + 1.0205x_2 - 0.0397x_3 + 0.0059x_1^2 - 0.0036x_2^2 - 0.0009x_3^2 - 0.176x_1x_2 + 0.0054x_1x_3 - 0.0010x_2x_3$	0.93
Unit Density (kg/m^3)	$-3374.408 + 285.427x_1 - 9.16708x_2 + 5.55107x_3 - 5.1623x_1^2 - 0.2563x_2^2 - 0.02371x_3^2 + 1.0288x_1x_2 - 0.20793x_1x_3 + 0.07566x_2x_3$	0.86
Tapped Density (kg/m^3)	$-282.3796 + 112.821x_1 - 37.447x_2 + 4.65223x_3 - 2.4547x_1^2 + 0.10838x_2^2 - 0.01468x_3^2 + 0.82445x_1x_2 - 0.0965x_1x_3 + 0.00847x_2x_3$	0.85

Note: x_1: Feedstock moisture content (%, w.b.); x_2: Die speed (Hz); x_3: Preheating temperature (°C) and R^2: Coefficient of determination

Results & Discussion

Individual Optimum Process Conditions

The objective functions developed for pelleting process conditions for the quality and energy consumption are used as the objective functions for the testing the new hybrid genetic algorithm developed. Table 2 gives the equations used for the testing.

Common Optimum Process Conditions using the Hybrid Genetic Algorithm (HGA)

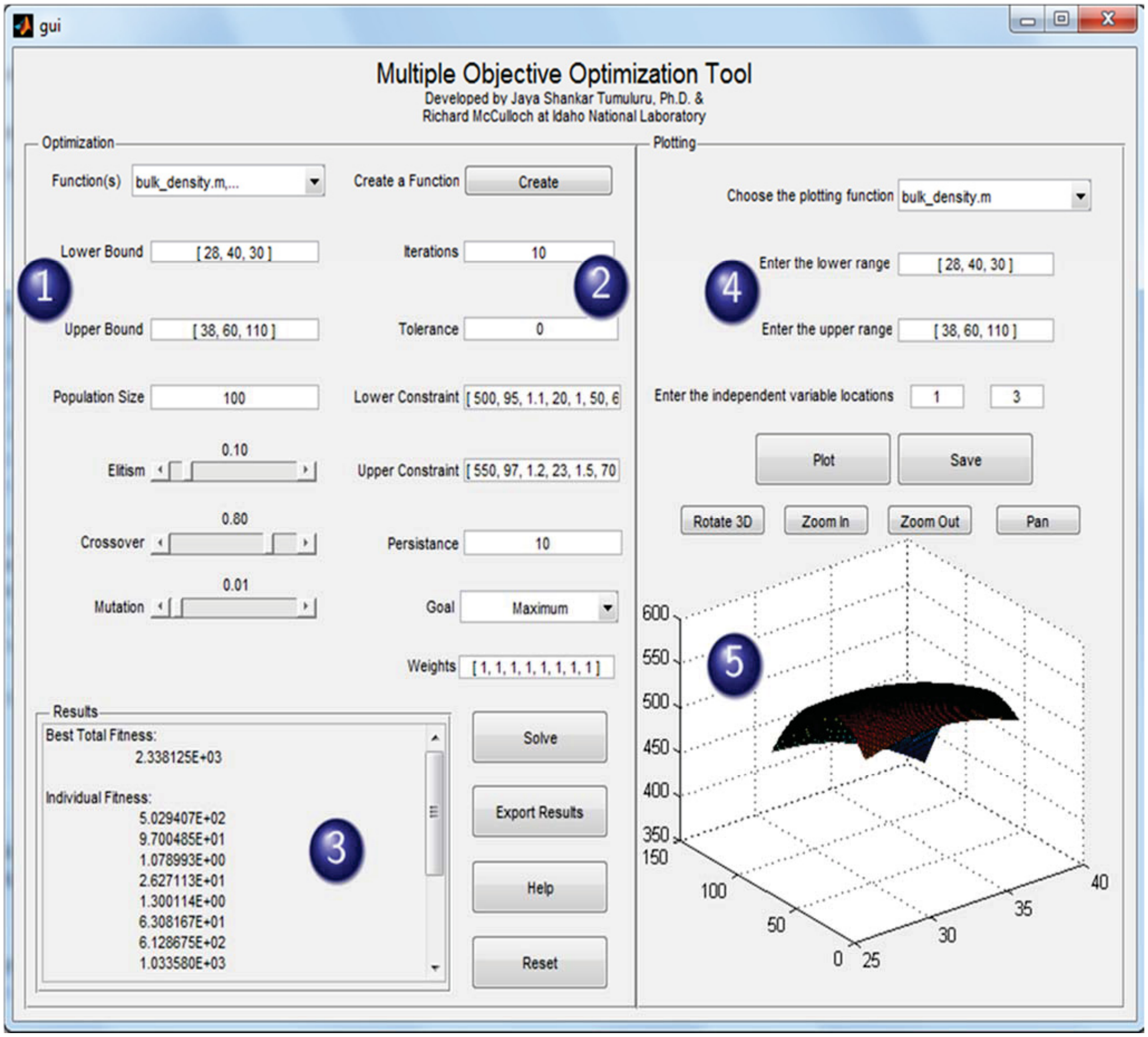
The objective functions used to find the common optimum process condition are the response surface models (second-order polynomial regression equations) obtained for BD, D, and SEC (Table 2). A common equation ((4)) was used to find the common optimum process condition. When this equation ((4)) is subjected to maximization using the hybrid genetic algorithm, the resulting solution will help to maximize density and durability and minimize specific energy consumption.

$$\text{BD+D-SEC-PMC+UD+TD=maximize} \tag{4}$$

A random population of 100 chromosomes and crossover and mutation probabilities of 0.80 and 0.99, respectively, were used based on the earlier studies conducted by Shankar and Bandyopadhyay (2004); Shankar et al. (2010); Shankar and Sokhansanj (2010) and Tumuluru et al. (2013) on optimization of extruded biomaterial properties using a simple genetic algorithm. The search for the optimum was carried up to 100 iterations.

Single and Multi-Function Optimization

Figure 4 shows the Multiple Objective Optimization Tool developed in this work. Area 1 is where inputs for single objective optimization are provided by the user. This type of input is common in all genetic algorithm optimization programs and tools. Area 2 is where multi-objective inputs are defined. The main benefit of this tool is the ability to constrain the output as shown and to assign weights to the objective functions as the comparison demands. Another feature that was missing from most other software programs was the ability to choose whether to maximize or to minimize the objective function. While this may seem like a small feature, keeping consistent approaches reduces complexity and thus reduces the possibility of human error in both obtaining and interpreting the results. Most genetic algorithm tools, such as that of Matlab for example, in order to find the maximum of an objective function the user must negate the objective function and then negate the result. Switching between minimizing and maximizing in this manner can be troublesome. Area 3 is where immediate results are displayed to the user. A more comprehensive set of results, including the entire final population, intermediate population and fitness information, final fitness and corresponding inputs, etc. can be generated by using the “Export Results” button. Areas 4 and 5 give the user a method of visualizing the functions being optimized. In area 4 the user supplies visualization inputs such as the function to be plotted and the appropriate ranges for the input variables. The resulting plot is shown in area 5.



gui

Multiple Objective Optimization Tool

Developed by Jaya Shankar Tumuluru, Ph.D. &
Richard McCulloch at Idaho National Laboratory

Optimization

Function(s): bulk_density.m,...

Lower Bound: [28, 40, 30]

Upper Bound: [38, 60, 110]

Population Size: 100

Elitism: 0.10

Crossover: 0.80

Mutation: 0.01

Iterations: 10

Tolerance: 0

Lower Constraint: [500, 95, 1.1, 20, 1, 50, 6

Upper Constraint: [550, 97, 1.2, 23, 1.5, 70

Persistance: 10

Goal: Maximum

Weights: [1, 1, 1, 1, 1, 1, 1, 1]

Results

Best Total Fitness:
2.338125E+03

Individual Fitness:
5.029407E+02
9.700485E+01
1.078993E+00
2.627113E+01
1.300114E+00
6.308167E+01
6.128675E+02
1.033580E+03

Plotting

Choose the plotting function: bulk_density.m

Enter the lower range: [28, 40, 30]

Enter the upper range: [38, 60, 110]

Enter the independent variable locations: 1 3

Figure 4: The user front-end of the Multi-Objective Optimization Tool

Singular Optimization Problem

The specific energy equation (Table 2) was used to test single optimization problem. Figure 5 shows the optimum and mean fitness values for each generation, both with and without the steepest descent algorithm included in the simulation. The bounds for the input variables are given as

$$28 \text{ \%w.b.} \leq \text{feedstock moisture} \leq 38 \text{ \%w.b.}$$

$$40 \text{ Hz} \leq \text{die rotational frequency} \leq 60 \text{ Hz}$$

$$30\ ^{o}\text{C} \leq \text{feedstock temperature} \leq 110\ ^{o}\text{C}$$

For each simulation 100 candidates comprised each population, elitism preserved the top 10% of the population, crossover was applied to 80% of the population, 1% of the population was randomly mutated, and the simulation terminated when 100 generations were made or when the absolute relative error between the population mean and the population optimum was below 1E-12. This allowed for a timed execution. Normally a genetic algorithm is run for a set number of iterations. If the comparison were done with fixed iterations both codes would run for the same amount of time and use roughly the same computational resources. By setting a relative convergence criterion the code was stopped when it was within a tolerance, 1E-12. The results below show the performance until the convergence criterion was met. The optimum conditions were those that minimized specific energy consumption and the other quality attributes (minimum of pellet moisture content, maximum of unit, bulk and tapped density and durability) obtained using hybrid genetic algorithm, are given in Table 3.

Table 3: Optimum process conditions for the minimization of each individual objective function

	Property	**Goal**	**Value**	**Conditions**		
				Moisture (%w.b.)	**Die Speed (Hz)**	**Temperature (°C)**
Single Objective Optimization	Bulk Density (kg/m^3)	Max	566	29.0	59.7	67.0
	Tapped Density (kg/m^3)	Max	649	29.7	59.8	80.3
	Unit Density (kg/m^3)	Max	1,085	29.9	59.7	78.4
	Durability	Max	98	29.5	59.6	73.4
	Moisture Content (%w.b.)	Min	14	28.2	40.2	110.0
	Specific Energy Consumption (kWhr/ton)	Min	43	28.1	60.0	30.1

Both the hybrid and non-hybrid codes used to optimize this problem were developed at Idaho National Laboratory. The only computational difference between the codes is the implementation of a local search routine during the optimization loop. Both codes were written in Matlab and have not yet been optimized for performance. Because most, if not all, commercial GA codes are optimized to some degree, the comparison for performance considerations is best made using codes that are exactly similar with the only difference being the improvement of interest, in this case the addition of a local gradient search routine.

It can be seen in Figure 5 that the case of not using the steepest descent converges more slowly than the hybrid case. The mean values of the hybrid algorithm drop dramatically compared to the non-hybrid case. This is attributed to the fact that each candidate solution is optimized and therefore the entire population is at a local minimum or maximum, whereas in the non-hybrid case candidates are extremely likely to be resting on a gradient instead of an extrema.

The goal of this algorithm is to ultimately save time and resources for computation and optimization. Therefore a comparison needs to be made regarding computational time for the two methods. For this reason constraints are not considered in these simulations. Figure 6 shows the mean program execution times for the program to converge on a solution. Because the algorithm is stochastic in nature a direct comparison cannot be made. Rather a statistical comparison must be used to bound the true time required for convergence within a confidence interval at a specified confidence value. Table 4 gives the number of iterations and the execution time required for methods as well as upper and lower bounds at 95% confidence levels for the optimization of specific energy consumption. The results show that even with the added computation of the steepest descent method the solution converges much faster and thus time is conserved by not having to perform more iterations.

Table 4: Iterations and execution times required for convergence between the fitness mean and fitness minimum/maximum for specific energy consumption. 1,000 trials were used in each case

	Non-Hybrid	**Hybrid**

Mean Iterations	20.195	14.027
95% Interval	19.911 <i< 20.479	13.886 <i< 14.168
Mean Times (s)	4.362	3.170
95% Interval (s)	4.301 < t < 4.423	3.139 < t < 3.200

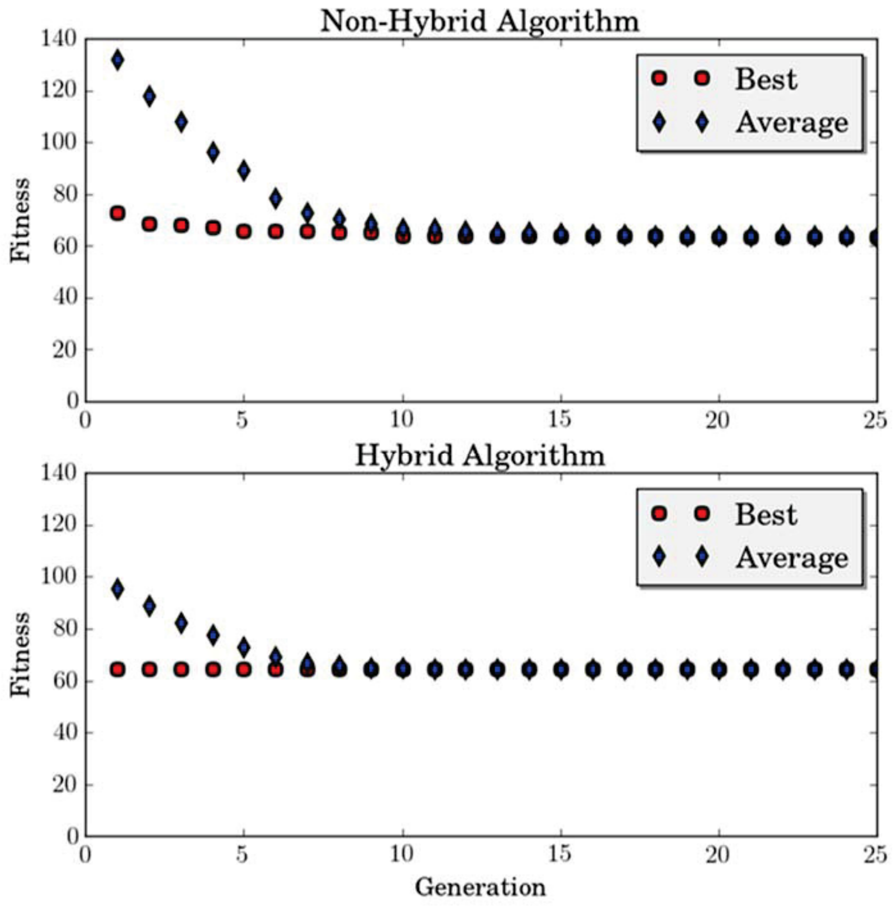


Figure 5: Best and average fitness values for 25 generations both without (top) and with (bottom) the steepest descent algorithm being implemented in the optimization of specific energy consumption as given in Table 2

Multi-Function Optimization

(4) was used as the objective function to test the multi-objective optimization capability of the new hybrid genetic algorithm developed. A similar analysis as the last example was done to test the performance improvement by using the regular and hybrid genetic algorithms developed at INL. The bounds for feedstock moisture, die rotational frequency and feedstock temperature are the same as given above: 28-38 %w.b., 40-60 Hz, 30-110 ^{o}C, respectively. Figure 7 shows the optimum and mean fitness values for each generation, both with and without the steepest descent algorithm included in the simulation. Clearly there is a faster rate of convergence with the hybrid simulation. Figure 8 shows the mean execution times for 1,000 trial simulations. Statistical comparisons and confidence intervals are given in Table 6 for the multi-objective optimization. In the same manner as the last optimization example, the solution was considered to have converged when the difference between the mean population fitness and the best population fitness was less than 1E-12. The optimum process conditions obtained using hybrid genetic algorithm using(4)are given in

Table 5.

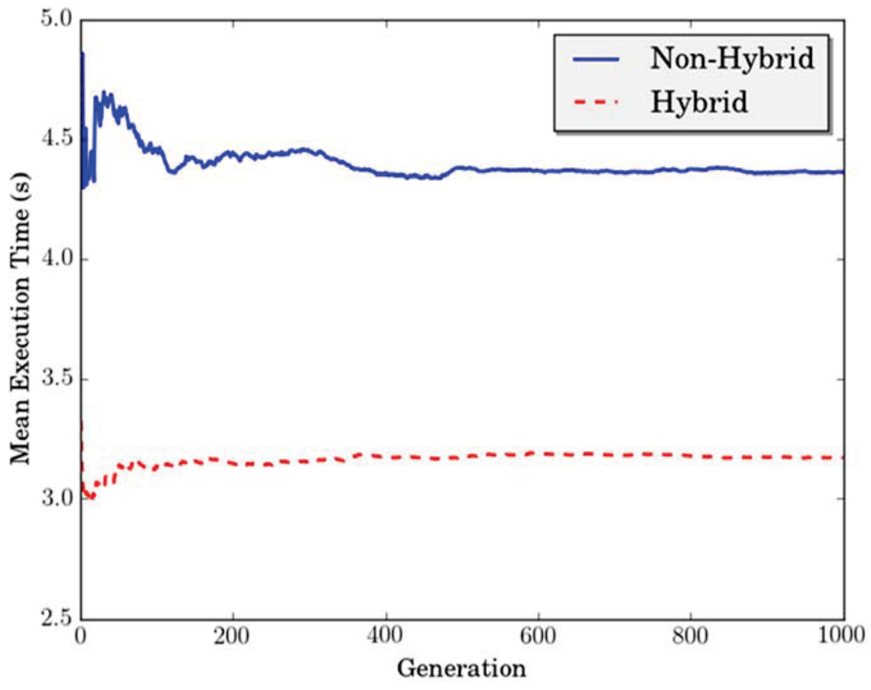


Figure 6: Mean execution times after each trial for 1,000 trials for the single objective optimization of specific energy consumption given in Table 2

Table 5: Optimum process conditions for each objective function individually as well as combined for multi-objective optimization

	Property	Goal	Value	Process Conditions		
				Moisture (%w.b.)	Die Speed (Hz)	Preheating temperature (oC)
Multi-Objective Optimization	Bulk Density (kg/m3)	Max	561	30.0	59.6	75.6
	Tapped Density (kg/m3)	Max	648			
	Unit Density (kg/m3)	Max	1,085			
	Durability	Max	98			
	Moisture Content (%w.b.)	Min	24			
	Specific Energy Consumption (kWhr/ton)	Min	56			

Table 6: Iterations and execution times required for convergence between the fitness mean and fitness minimum/maximum for all of the pellet properties given in Table 2. 1,000 trials were used in each case

	Non-Hybrid	Hybrid
Mean Iterations	20.088	13.750
95% Interval	19.819 < i < 20.357	13.638 < i < 13.862
Mean Times (s)	4.456	3.249
95% Interval (s)	4.391 < t < 4.520	3.207 < t < 3.292

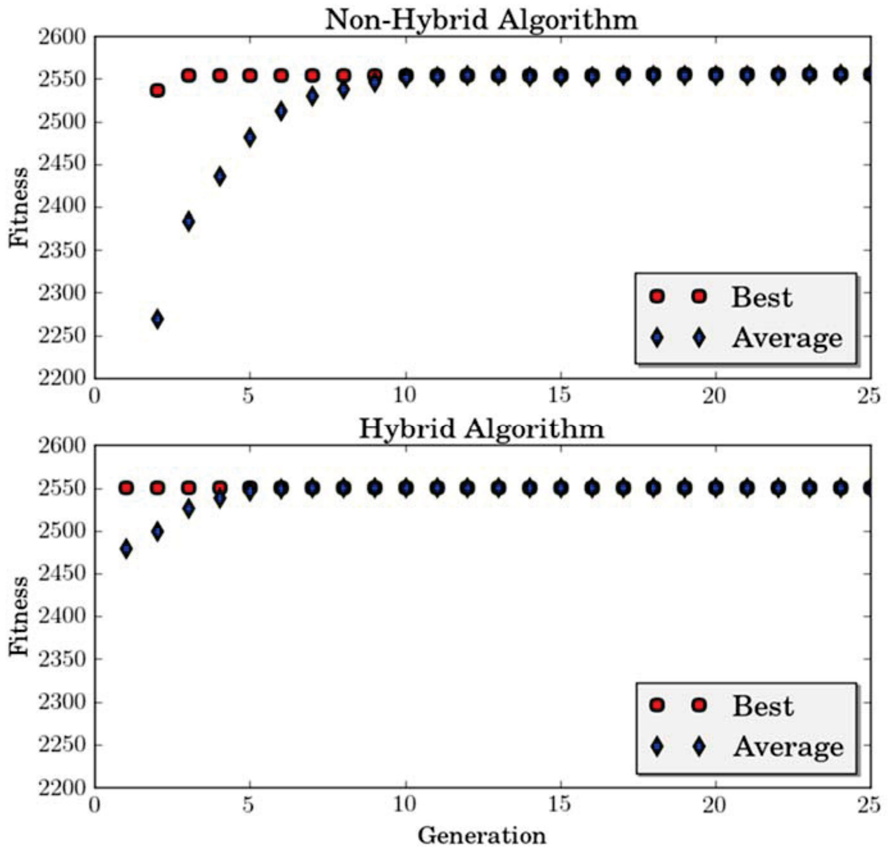


Figure 7: Best and average fitness values for first 25 generations of the multi-objective optimization of all the pellet properties in Table 2

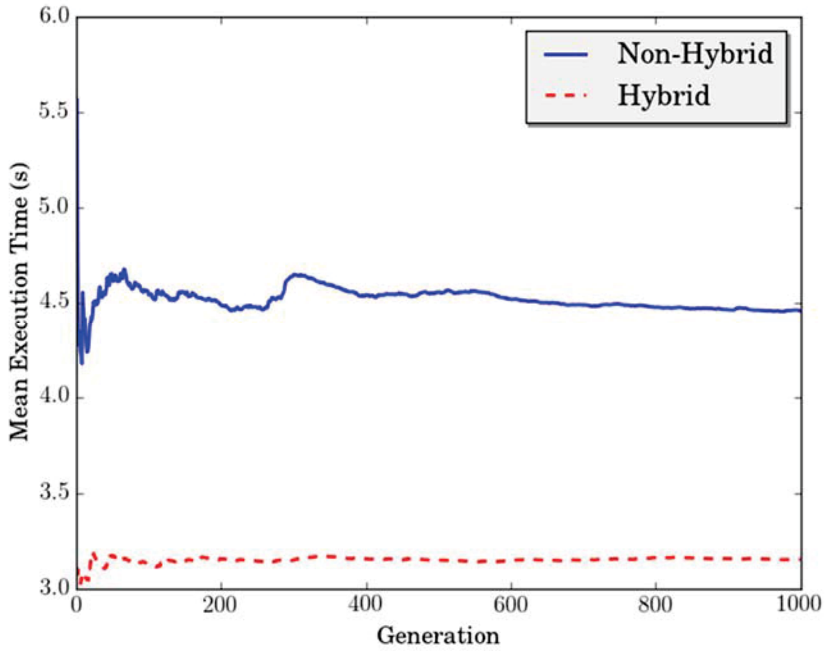


Figure 8: Mean execution times for 1,000 trials of the multi-objective optimization of the pellet properties given in Table 2

Comparison of Hybrid GA and the Commercial Matlab GA

The performance improvement in reducing the iterations and time required for convergence has been studied in the previous examples. An observation needs to be made to show the convergence of the hybrid GA as compared to readily available commercial software. For this example the GA from the commercial code Matlab is used to optimize a single objective function: bulk density from Table 2. Matlab does not give the user the ability to easily constrain the output, thus constraints are also not considered in this example.

The bounds for feedstock moisture, die rotational frequency and feedstock temperature are the same as given above: 28-38 %w.b., 40-60 Hz, 30-110^o C, respectively. Unlike the previous examples, execution times are not considered for this example, thus the convergence criteria is a fixed number of 100 iterations (generations) for each case. Figure 9 shows the fitness of the best candidate from each generation for both the hybrid GA from INL (red square) and the commercial Matlab GA (blue diamonds). It can be seen that the hybrid GA reaches the optimum in a single iteration. In this example the bulk density function is uni-modal in the region of interest. Thus on the first iteration the local search routine optimizes the best candidate locally by placing that candidate on the local peak, which coincidentally is also the global peak for this function in this range. Similar results are obtained using the hybrid GA for the remaining functions in Table 2 using the same inputs given here.

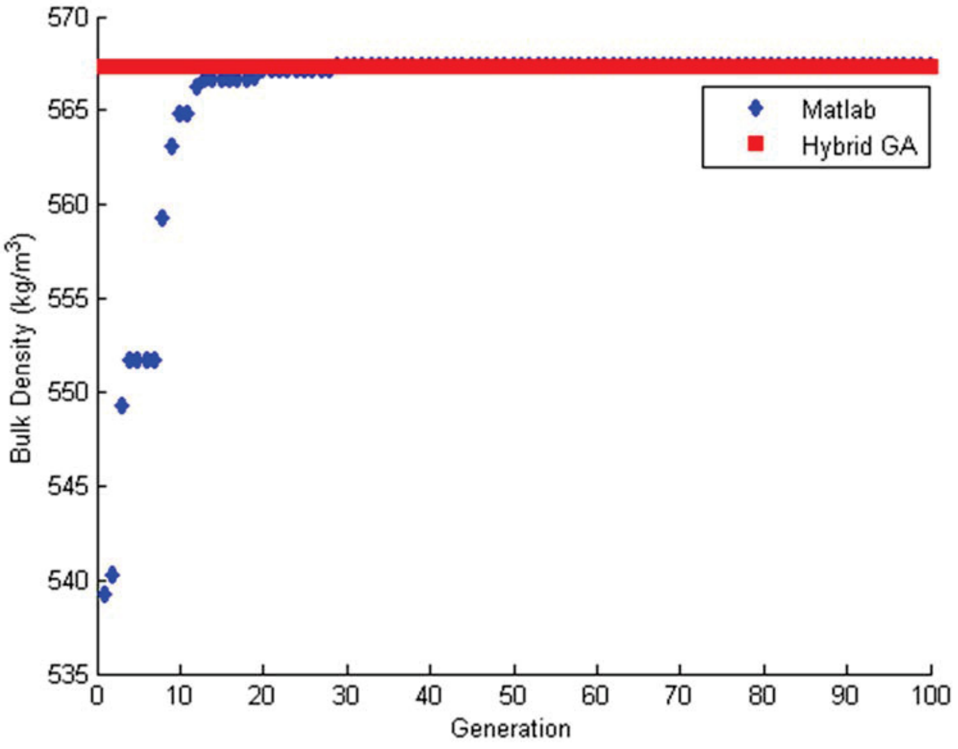


Figure 9: Fitness of the best candidate at each generation for both the commercial

Conclusions and Future Work

While the program works sufficiently well it has room for improvement. The area likely to produce the greatest improvement is in the deterministic subroutine implementation. Currently an adaptive step steepest descent method optimizes each candidate before they are used for breeding. While effective this approach can be less efficient than other methods. An optimum deterministic method needs to be identified and applied in place of the steepest descent method in order to maximize the program's performance. Another are of interest is in the determination of a complete Pareto front, or Pareto set of solutions. Currently the program outputs any solution within a specified tolerance of the best solution. While this is convenient for identifying potential input configurations, this approach does not have a mathematical basis and should be replaced in future revisions.

The results demonstrate the real-time improvement in execution time and iterations required for convergence on the optimum set of input parameters. The deterministic optimization algorithm implemented in parallel with the stochastic genetic algorithm is simple, yet effective. By including the steepest descent algorithm each candidate rests on a local extrema before being used to generate the subsequent population. This brings the population fitness much closer to the goal and enhances the ability of the genetic algorithm to correctly isolate the global optimum.

Author Disclosure Statement

No competing financial interests exist. This information was prepared as an account of work sponsored by an agency of the U.S. government. Neither the U.S. government nor any agency thereof, nor any of their employees,

References

April, J., Glover, F., Kelly, J., & Laguna, M. (2003). Practical introduction to simulation optimization. *Proceedings of the 2003 Winter Simulation Conference*. ed. S. Chick, T. Sanchez, D. Ferrinand D. Morrice (pp. 71-78). Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

ASABE Standards. (2007). S269.4. Cubes, pellets and crumbles-definition and methods for determining density, durability and moisture content. ASABE, St. Joseph, MO.

Chapra, S. & Canale, R. (2015). *Numerical Methods for Engineers*, 7th Ed. New York, NY: McGraw-Hill Education.

Davis, L. (1991). *Handbook of Genetic Algorithms*. New York, NY: Van Nostrand Reinhold.

El-Mihoub, T.A.,Hopgood, A.A.,Nolle, L., & Battersby, A. (2006). Hybrid genetic algorithms: A Review. *Engineering Letters*, 13:2, EL_13_2_11.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Boston, Mass.: Addison-Wesley.

Hart, W.E. (1994). Adaptive global optimization with local search. Ph.d. Dissertation. San Diego, CA.: University of California, Department of Computer Science & Engineering.

Holland JH. (1992). Genetic algorithms. *Sci. Am.*, *267*(1), 66-72.

Kalyanmoy, D. (2000). *Optimization for Engineering Design: Algorithms and Examples.* New Delhi: Prentice-Hall of India Private Ltd.

Kavraki, L., Svestka, P., Latombe, J. C., & Overmars, M. (1996). Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Trans. Robot. Autom, 12* (4): 566–580.

Kuffner, J. & LaValle, S. (2000). RRT-Connect: An efficient approach to single-query path planning. *IEEE International Conference on Robotics and Automation*, (pp. 995–1001). San Francisco, CA.

Lipowski, A. & Lipowski, D. (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6): 2193-2196.

Petrova, S. & Solov'ev, A. (1997). The origin of the method of steepest descent. *Historia Mathematica*; 24(4): 361-375.

Shankar, T. J. & Bandyopadhyay, S. (2004). Optimization of extrusion process variables using a genetic algorithm. *Food and Bioprod. Process*, 82 (2), 143—150.

Shankar, T. J., Sokhansanj, S., Bandyopadhyay, S., & Bawa, A.S. (2010). A case study on optimization of biomass flow during single-screw extrusion cooking using genetic algorithm (GA) and response surface method (RSM). *Food and Bioprocess Tech., 3*(4), 498-510.

Shankar, T.J. & Sokhansanj, S. (2010). A case study on investigating the effect of genetic algorithm operators on predicting the global minimum hardness value of biomaterial extrudate. *Int. J. Optim., 2*(2):109–123.

Simon, D. (2013*). Evolutionary Optimization Algorithms Biologically-Inspired and Population-based Approaches to Computer Intelligence*. Hoboken, New Jersey: John Wiley & Sons.

Tumuluru, J.S. (2014). Effect of process variables on the density and durability of the pellets made from high moisture corn stover. *Biosyst. Eng., 119*: 44-57.

Tumuluru, J.S. (2014). High moisture corn stover pelleting in a flat die pellet mill fitted with a 6 mm die: physical properties and specific energy consumption. *Energy Sci. Eng.*, 3(4): 327-341.

Tumuluru, J.S., Sokhansanj, S., Bandyopadhyay, S., & Bawa, A.S. (2013). Changes in moisture, protein and fat content of fish and rice flour coextrudates during single screw extrusion cooking. *Food Bioprocess Tech.*; *6*(2): 403–415.