Efficient Global Optimization of MEMS Based on Surrogate Model Assisted Evolutionary Algorithm

Bo Liu
Department of Computing
Glyndwr University, UK
Email: b.liu@glyndwr.ac.uk

Anna Nikolaeva
Department of Applied Mechanics
Bauman Moscow State Technical University, Russia
Email: Nikolaeva@bmstu.ru

***Abstract*—Optimization plays a key role in MEMS design. However, most MEMS design optimization (exploration) methods either depend on ad-hoc analytical / behavioural models or time consuming numerical simulations. Surrogate modeling techniques have been introduced to integrate generality and efficiency, but the number of design variables which can be handled by most existing efficient MEMS design optimization methods is often less than 5. To address the above challenges, a new method, called Adaptive Gaussian Process-Assisted Differential Evolution for MEMS Design Optimization (AGDEMO) is proposed. The key idea is the proposed ON-LINE adaptive surrogate model assisted optimization framework. In particular, AGDEMO performs global optimization of MEMS using numerical simulation and the differential evolution (DE) algorithm, and a Gaussian process surrogate model is constructed ON-LINE to predict the results of expensive numerical simulations. AGDEMO is tested by two actuators (both with 9 design variables). Comparisons with state-of-the-art methods verify advantages of AGDEMO in terms of efficiency, optimization capacity and scalability.**

1 INTRODUCTION

Design optimization plays an important role in the MEMS design process. Among various optimization methods, evolutionary algorithms (EAs) are attracting much attention because of their global optimization ability, free of a good initial design, generality and robustness [1]. However, numerical simulations (e.g., finite element analysis) are often needed to obtain accurate performance of a MEMS, which are computationally expensive. Considering thousands of simulations may be needed for a standard EA, the MEMS optimization can be very time consuming or even prohibitive [2].

To address the issue of efficiency, available EA-based MEMS design optimization methods can be mainly classified into three categories: (1) analytical model-based methods [2], [3], (2) numerical simulation-based methods assisted by manual design expertise [4], and (3) off-line surrogate model-based methods [5].

Analytical model-based methods adopt a computationally cheap model (analytical equations [3] or SPICE / HDL netlists [6]) to obtain the performance of the targeted MEMS. Although efficient, those methods suffer from the following two drawbacks: (1) Accuracy: It has been reported that analytical models can be used as coarse models but numerical simulations are often unavoidable to obtain accurate analysis [7]. (2) Generality: Most analytical models are specifically developed for a certain kind of MEMS, which are ad-hoc.

Directly embedding numerical simulations into an EA can provide high quality design solutions with accurate analysis. Moreover, this kind of methods is general. To handle the bottleneck of efficiency, some methods integrate design expertise from manual design to narrow down the search range [4] or to first obtain an initial reasonably good design and then use EAs / local optimization methods (with numerical simulation embedded). However, this again invites ad-hoc design process suffering the generality.

To combine efficiency and generality, off-line surrogate model-based methods are introduced to MEMS design optimization [5], which work as follows. An initial random sampling is firstly carried out. Using the sampled design variables as the input and the performance via numerical simulations as the output, a black-box surrogate model is constructed to approximate the performance of the MEMS device. The surrogate model is often constructed by statistical learning techniques. For example, [5] uses artificial neural networks (ANN) to construct a black-box surrogate model for predicting the performances of new candidate designs. The computationally cheap surrogate model is then used to replace the expensive numerical simulation model so as to improve the optimization efficiency.

In off-line surrogate model-based methods, the main computational cost comes from initial random sample generation (training data points generation via simulation) for building the surrogate model. In an electronic system, the MEMS may have strict design specifications to collaborate with other ICs and devises, which leads to a high approximation accuracy requirement of the surrogate model. For instance, the required critical pressure and displacement are often in narrow ranges for an actuator to be integrated in a biomedical system. When the number of design variables is more than a few, obtaining an accurate enough surrogate model may need a tremendous number of initial samples. For example, in [5], the 6 design variables are cut down to 3 (other design variables are kept constant) to obtain an accurate ANN model with a reasonable number of initial samples. Clearly, this suffers the optimality.

To address the above challenges, a new method is proposed, called Adaptive Gaussian Process-Assisted Differential Evolution for MEMS Design Optimization (AGDEMO). The method aims to:

- Achieve comparable results with MEMS optimization methods which directly embed numerical simulations to a standard EA (often considered as the best in terms of solution quality);
- Highly improve the efficiency of the above method as well as off-line surrogate model-based methods;
- General enough to handle MEMS optimization with more than a few design variables without any initial solution.

The remainder of the paper is organized as follows. Section 2 introduces the key techniques and the general framework of AGDEMO. Section 3 tests AGDEMO by practical examples. The concluding remarks are presented in Section 4.

2 THE AGDEMO ALGORITHM

2.1 Key Ideas of AGDEMO

A surrogate model is used in AGDEMO. The major drawback of the off-line surrogate model-based method is that it tries to build an accurate enough surrogate model before optimization. However, only a small region of the design space is useful in an EA-based optimization, and many of these expensive simulations to cover the whole design space (most are not optimal region) are wasted.

978-3-9815370-6-2/DATE16/

Motivated by this observation, AGDEMO proposes an ON-LINE surrogate model assisted EA framework. It first constructs a very rough surrogate model by a small number of initial samples, and then improves it online but only in the necessary regions of the design space; in particular, candidate designs with good "potential" will be selected and simulated to serve as new training data points for the update of the surrogate model. The search is supported by the increasingly improved surrogate model until convergence. Consequently, much fewer simulations are needed. Moreover, because all the candidate designs that have good potential to be used as the final result are evaluated by simulations, rather than by the surrogate model predictions, the accuracy requirement is naturally met.

The key issue then becomes identifying designs with good "potential" for simulation in AGDEMO. The answer, however, is not trivial. Unlike off-line surrogate models, the quality of the surrogate model is improving gradually, as more training data points are provided in the optimization process. Hence, the predicted good solutions may be wrong especially at the beginning of the optimization when there are insufficient training data points. Therefore, AGDEMO focuses on two main questions: (1) How to build a high quality surrogate model using a small number of training data points which are restricted by the computing budget (section 2.3)? (2) How to identify promising candidate solutions given uncertainty of predictions (Section 2.4)?

2.2 Basic Techniques

Gaussian Process (GP) machine learning [8] is used for surrogate modeling. A very brief introduction is provided as follows and more details are in [8].

Given a set of observations $x = (x^1, \ldots, x^n)$ and $y = (y^1, \ldots, y^n)$, GP predicts a function value $y(x)$ at some design point x by modeling $y(x)$ as a Gaussian distributed stochastic variable with mean μ and variance σ. By maximizing the likelihood function that $y = y^i$ at $x = x^i (i = 1, \ldots, n)$ and best linear unbiased prediction and predictive distribution, the function value $y(x^*)$ at a new point x^* can be predicted as:

$$\hat{y}(x^*) = \hat{\mu} + r^T R^{-1}(y - I\hat{\mu}) \tag{1}$$

where

$$r = [corr(x^*, x^1), corr(x^*, x^2), \ldots, corr(x^*, x^n)]^T \tag{2}$$

$$R_{i,j} = corr(x^i, x^j), i, j = 1, 2, \ldots, n. \tag{3}$$

$$\hat{\mu} = (I^T R^{-1} y)^{-1} I^T R^{-1} y \tag{4}$$

$$\hat{\sigma^2} = (y - I\hat{\mu})^T R^{-1}(y - I\hat{\mu}) n^{-1} \tag{5}$$

$corr$ is the correlation function and I is a $n \times 1$ vector of ones. The measurement of the uncertainty of the prediction (mean square error) is:

$$\hat{s^2}(x^*) = \hat{\sigma^2}[I - r^T R^{-1} r + (I - r^T R^{-1} r)^2 (I^T R^{-1} I)^{-1}] \tag{6}$$

In this work, we use the ooDACE toolbox [9] to implement the GP surrogate model.

The DE algorithm [10] is selected as the global search engine. The following DE operators are used:

DE mutation: For each target individual i, a mutant vector is generated:

$$v^i = x^i + F \cdot (x^{best} - x^i) + F \cdot (x^{r_1} - x^{r_2}) \tag{7}$$

where x^{best} is the best individual in the population and x^{r_1} and x^{r_2} are two different solutions randomly selected and are also different from x^{best}. v^i is the i^{th} mutant vector in the population after mutation. $F \in (0, 2]$ is the scaling factor.

DE crossover: The following crossover operator is applied to produce the child population u:

1 Randomly select a variable index $j_{rand} \in \{1, \ldots, d\}$ (d is the number of variables),

2 For each $j = 1$ to d, generate a uniformly distributed random number $rand$ from $(0, 1)$ and set:

$$u_j^i = \begin{cases} v_j^i, & \text{if } (rand \leq CR) | j = j_{rand} \\ x_j^i, & \text{otherwise} \end{cases} \tag{8}$$

where $CR \in [0, 1]$ is a constant called the crossover rate.

2.3 The General Framework of AGDEMO

The AGDEMO algorithm consists of the following steps.

Step 1: Sample α (often small) solutions from the design space using Latin Hypercube sampling method [11], perform numerical simulations of all these solutions and let them form the initial database.

Step 2: If a preset stopping criterion (e.g., computing budget) is met, output the best solution from the database; otherwise go to Step 3.

Step 3: Select the λ best solutions from the database based on the fitness values to form a population P.

Step 4: Apply the DE operators (eqn. (7) and (8)) on P to generate λ child solutions.

Step 5: Calculate the median of the λ child solutions. Take the τ nearest solutions to the median in the database (based on Euclidean distance) and their function values (performances) as the training data points to construct a GP surrogate model.

Step 6: Prescreen the λ child solutions generated in Step 4 using the method in Section 2.4.

Step 7: Simulate the estimated best child solution from Step 6. Add this solution and its performance (via simulation) to the database. Go back to Step 2.

The goal of the above surrogate model assisted EA (SAEA) framework is to improve the locations of training data points. In literature, the number of training data points is often considered as the main factor affecting the quality of a surrogate model. But with the same number of training data points, it is intuitive that using training data points located near to the points waiting to be predicted (child population in Step 4) can obtain better surrogate model and prediction results. Hence, in each iteration, the λ current best candidate solutions form the parent population (it is reasonable to assume that the search focuses on the promising region) and the best candidate based on prescreening in the child population is selected to replace the worst one in the parent population. Hence, only at most one candidate is changed in the parent population in each iteration, so the best candidate in the child solutions in several consecutive iterations may be quite near to each other (they will then be simulated and are used as training data points). Therefore, the training data points describing the current promising region can be much denser compared to those generated by a standard EA population updating mechanism.

On the other hand, this SAEA framework emphasizes exploitation to help surrogate modeling, and the population diversity decreases compared to a standard EA. However, there are research works stating that standard EAs may have excessive diversity [12]. To maintain the exploration ability, we use DE/current-to-best/1 mutation (eqn. (7)). There are various kinds of DE mutations leading to different population diversity. Our pilot experiments on more than

ω_m =median(OM)
FOR i=1 to λ
  ω_i=*Norm* (ω_m ,0.5)
  IF ω_i<0, ω_i=0
  IF ω_i>3, ω_i=3
END
OM=OM ∪ ω_b

Figure 1. The pseudocode of the adaptive prescreening method

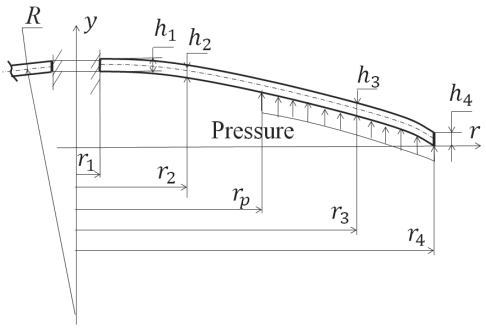

Figure 2. The shell-type dome actuator

10 mathematical benchmark problems show successful results. We also found that when using DE mutations with higher population diversity (e.g., DE/rand/1 [10]), the surrogate model quality may be suffered. The above evidences imply that the AGDEMO framework balances exploration and exploitation appropriately.

2.4 The Adaptive Prescreening Method

Due to prediction uncertainty, some truly good candidate designs may not have good prediction values because of large prediction uncertainty. Thus, they may not be selected to simulate and to serve as new training data points, which leads to a risk that the surrogate model quality in that promising region may not be improved in consecutive iterations. Considering the prediction uncertainty when selecting the best candidate in the child population (Steps 6-7) is called prescreening in the computational intelligence field [13]. Most prescreening methods reward candidates with large prediction uncertainty (eqn. (6)) when assessing the quality of a candidate design.

Most popular prescreening methods have a fixed reward scheme, but for different problems and for different search phases of the same problem, the optimal extent of reward is different. Therefore, in AGDEMO, an adaptive lower confidence bound (LCB) method is developed. Considering minimization of $y(x)$, LCB works as follows:

$$y_{lcb}(x) = \hat{y}(x) - \omega\hat{s}(x), \omega \in [0, 3] \quad (9)$$

where ω controls the extent of reward. In AGDEMO, we use $y_{lcb}(x)$ instead of $\hat{y}(x)$ for the objective function, where ω is adaptively generated. For constraints, ω is set to 0. The reasons are: (1) In MEMS design, the constraint is often tight (e.g., the displacement is around $150\mu m$ with a tolerance of $5\mu m$). We do not expect that many expensive simulations are cost for candidates which are nearly (but not) feasible. (2) The reward on the objective function already helps to promote exploration.

The adaptive LCB method for the objective function works according to the steps in Fig. 1.

ω is generated from $norm(\omega_m, 0.5)$ (the function $norm$ generates a Gaussian distributed random number with a mean of ω_m and a preset standard deviation) for each candidate design in the child population. In the first L iterations, the initial ω_m is used. Then, the procedure in Fig. 1 is used. ω_b refers to the ω value used for generating the estimated best candidate design in the current iteration (Step 7), which is then added to the memory of successful ω values, OM. It is reasonable to assume that most of the ω values used for generating the estimated best candidate design are appropriate for a certain MEMS optimization problem in a certain search phase.

Table I
RANGES OF DESIGN VARIABLES OF EXAMPLE 1 (IN μm)

$Variables$	R	r_1	r_2	r_3	r_p	h_1	h_2	h_3	h_4
Lower bound	3.2e+4	200	200	200	200	10	10	10	10
Upper bound	5.2e+4	400	2.8e+3	2.8e+3	2.8e+3	40	40	40	40

2.5 Parameter Settings

For DE parameters, we set $F = 0.8$, $CR = 0.8$, $\lambda = 50$, which is a common setting of the DE algorithm [10]. For surrogate modeling parameters, we set $\alpha = 5 \times d$ and $\tau = 10 \times d$. This is based on the empirical rule of [13], [14] for on-line surrogate modeling and pilot experiments show success. For adaptive prescreening, L is set to 50 and the initial ω_m is set to 1.5. It is clear that L is not sensitive. Setting initial ω_m to 1.5 shows the balance of exploration and exploitation in early phase. In section 3, the above same parameters are used for different examples.

3 EXPERIMENTAL RESULTS AND COMPARISONS

In this section, the AGDEMO algorithm is demonstrated by two MEMS actuators, both with 9 design variables. The examples are run on a PC with 1.90GHz Xeon CPU and 24GB RAM. ANSYS is used as the simulation tool. 10 runs with independent random numbers are performed for AGDEMO. Two reference methods are: (1) standard DE with ANSYS simulation embedded with a budget of 2500 simulations, (2) the off-line surrogate model-based method [5].

3.1 Example 1

The first example is a shell-type dome actuator [15] made of silicon (Fig. 2). The design variables and the ranges of them are in Table 1. The optimization goal is the area consumption and the design constraints are the critical pressure $P_c \in [85, 95]pa$ and the displacement $D_f(@P_c) \in [145, 155]\mu m$.

In all the 10 runs for AGDEMO, the constraints are satisfied and the average $\sqrt{area}$ is $2.61mm$ and the best $\sqrt{area}$ is $2.52mm$. In all the 10 runs, the convergence happens before using 200 ANSYS simulations. The average time cost is 3.1 hours (wall clock time). To verify the design quality, a standard DE with the same common parameters is carried out. After 1350 simulations, the algorithm converges. The two constraints are satisfied with a $\sqrt{area}$ of $2.53mm$. To obtain the average result of AGDEMO, the standard DE needs 1200 simulations. Hence, AGDEMO obtains a comparable result with standard DE but decreases the computational cost by about 8 times.

Off-line surrogate model-based methods are then compared using 500 LHS samples. A feedforward ANN model is built but the best

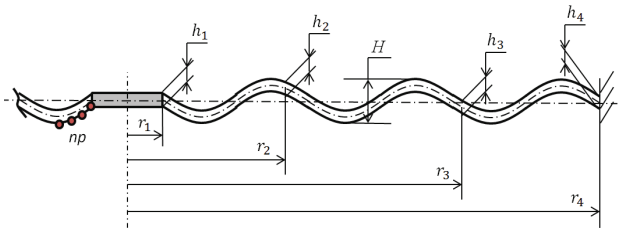

Figure 3. The corrugated membrane actuator

Table II
RANGES OF DESIGN VARIABLES OF EXAMPLE 2 (IN μm EXCEPT np)

$Variables$	H	r_1	r_2	r_3	h_1	h_2	h_3	h_4	np
Lower bound	75	30	150	400	6	6	6	6	3
Upper bound	80	70	390	500	7.5	7.5	7.5	7.5	5

error of the validation set in 10 training is 14.7%, which is impossible to be used to address the narrow ranges of the design specifications (i.e., constraints). A GP model is then built. Because GP modeling is based on interpolation (there is no validation set), the model is used for optimization with the standard DE algorithm for 10 runs. The optimized design is then verified by simulations. In all the 10 runs, none of them can satisfy the constraints and some of them are quite far from the specifications. This verifies that when the number of design variables is more than a few, such as 9 in this example, building a reliable off-line surrogate model needs a large number of samples, which cancels out the advantage on efficiency of this kind of methods. In contrast, AGDEMO provides an effective solution.

3.2 Example 2

The second example is a corrugated membrane actuator [16] made of silicon (Fig. 3). The design variables and the ranges of them are in Table 2. The minimization goal is the area consumption and the design specifications are $P_c \in [847.5, 852.5]kpa$ and $D_f(@Pc) \in [170, 175]\mu m$. It can be seen that the constraints are even stricter than example 1 since the tolerance is even smaller.

In all the 10 runs of AGDEMO, the constraints are satisfied. The average $\sqrt{area}$ is $0.550mm$. In 10 runs, the convergence happens before using 170 simulations, except 411 simulations is cost in one run to converge. The average time cost is 3.6 hours (wall clock time). The standard DE algorithm with the same common parameters converges after 2250 simulations, obtaining a $\sqrt{area}$ of $0.553mm$, costing about 40 hours (wall clock time). Both constraints are satisfied. Hence, more than an order of magnitude (i.e., 13 times) speed enhancement is achieved by AGDEMO compared to standard DE.

4 CONCLUSIONS

In this paper, the AGDEMO algorithm has been proposed for the surrogate model assisted optimization of MEMS. AGDEMO can provide results that are comparable to standard DE, which is expected to provide very high solution quality, but at far lower computational cost (up to 13 times speed improvement). Compared with other state-of-the-art methods, AGDEMO showed clear advantages in accuracy, efficiency and scalability, as demonstrated by the presented two 9 parameter MEMS examples. These results are achieved by the core ideas of the new online SAEA framework and the adaptive pre-screening method. Future work will focus on developing AGDEMO-embedded tools and introducing robust design to the AGDEMO method.

REFERENCES

[1] P. D. Barba and S. Wiak, "Evolutionary computing and optimal design of mems," *IEEE/ASME Trans. On Mechatronics*, vol. 20, no. 4, pp. 1160–1167, 2014.

[2] J. K. Coultate, C. H. Fox, S. McWilliam, and A. R. Malvern, "Application of optimal and robust design methods to a mems accelerometer," *Sensors and Actuators A: Physical*, vol. 142, no. 1, pp. 88–96, 2008.

[3] Z. Fan, J. Liu, T. Sørensen, and P. Wang, "Improved differential evolution based on stochastic ranking for robust layout synthesis of mems components," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 4, pp. 937–948, 2009.

[4] J. J. Lake, A. E. Duwel, and R. N. Candler, "Particle swarm optimization for design of slotted mems resonators with low thermoelastic dissipation," *Journal of Microelectromechanical Systems*, vol. 23, no. 2, pp. 364–371, 2014.

[5] Y. Lee, Y. Park, F. Niu, and D. Filipovic, "Design and optimisation of one-port rf mems resonators and related integrated circuits using ann-based macromodelling approach," *IEE Proceedings-Circuits, Devices and Systems*, vol. 153, no. 5, pp. 480–488, 2006.

[6] P. V. Nikitin, V. Jandhyala, D. White, N. Champagne, J. D. Rockway, C.-J. R. Shi, C. Yang, Y. Wang, G. Ouyang, R. Sharpe *et al.*, "Modeling and simulation of circuit-electromagnetic effects in electronic design flow," in *Proceedings of 5th International Symposium on Quality Electronic Design*. IEEE, 2004, pp. 244–249.

[7] T. D. Tan, S. Roy, N. P. Thuy, and H. T. Huynh, "Streamlining the design of mems devices: An acceleration sensor," *IEEE Circuits and Systems Magazine*, vol. 8, no. 1, pp. 18–27, 2008.

[8] T. J. Santner, B. J. Williams, and W. I. Notz, *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

[9] I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, and T. Dhaene, "Blind kriging: Implementation and performance analysis," *Advances in Engineering Software*, vol. 49, pp. 1–13, 2012.

[10] K. Price, R. Storn, and J. Lampinen, *Differential evolution: a practical approach to global optimization*. Springer-Verlag New York Inc, 2005.

[11] M. Stein, "Large sample properties of simulations using latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987.

[12] K. Rasheed and H. Hirsh, "Informed operators: Speeding up genetic-algorithm-based design optimization using reduced models." in *GECCO*, 2000, pp. 628–635.

[13] D. Jones, M. Schonlau, and W. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[14] B. Liu, D. Zhao, P. Reynaert, and G. G. Gielen, "Synthesis of integrated passive components for high-frequency rf ics based on evolutionary computation and machine learning techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 10, pp. 1458–1468, 2011.

[15] M. Zalalutdinov, K. L. Aubin, R. B. Reichenbach, A. T. Zehnder, B. Houston, J. M. Parpia, and H. G. Craighead, "Shell-type micromechanical actuator and resonator," *Applied Physics Letters*, vol. 83, no. 18, pp. 3815–3817, 2003.

[16] J. Han and D. P. Neikirk, "Deflection behavior of fabry-perot pressure sensors having planar and corrugated membrane," in *Micromachining and Microfabrication'96*. International Society for Optics and Photonics, 1996, pp. 79–90.