An Efficient Kriging-based Constrained Multi-objective Evolutionary Algorithm for Analog Circuit Synthesis via Self-adaptive Incremental Learning

Sen Yin, Wenfei Hu, Wenyuan Zhang, Ruitao Wang, Jian Zhang and Yan Wang
School of Integrated Circuits, Tsinghua University, China
Bio

- Sen Yin
- Tsinghua University

2014.9-2018.6 received the B.S. degree in Wuhan University, Wuhan, China

2018.9-present pursuing the Ph.D. degree at the Institute of Microelectronics, Tsinghua University, Beijing, China.

My current research interests include analog/RF circuit automatic synthesis, RF passive component modeling and the optimization of system-level circuits.
Contents

1. Introduction
2. Motivation
3. Proposed SILE algorithm
4. Experimental results
5. Conclusion
1. Introduction
2. Motivation
3. Proposed SILE algorithm
4. Experimental results
5. Conclusion
Introduction

- Manual analog circuit design is the mainstream in industry

- The key problem in analog circuit synthesis: circuit sizing

- We concentrate on analog circuit sizing, multi-objective optimization.
Introduction

• Most of the work concentrates on the synthesis of analog circuits at block-level. However, few results are reported on the optimization of an analog system.

• The optimization of an analog system

  [Image of system concept diagram]

  Top-down method[1]

  Bottom-up method[2]

• **Low efficiency** of multi-objective optimization is the bottleneck to optimize the system.

Introduction

- Multi-objective analog circuit sizing
- 1. model-based methods, 2. simulation-based methods

- Fast, but inaccuracy.
Introduction

- Multi-objective analog circuit sizing
- 1. model-based methods, 2. simulation-based methods

- High accuracy. Require a large number of time-consuming simulations.
Introduction

- Multi-objective analog circuit sizing
- Online model-based methods

- High accuracy with less number of simulations
1. Introduction

2. Motivation

3. Proposed SILE algorithm

4. Experimental results

5. Conclusion
Motivation

• Multi-objective Bayesian optimization (MOBO)\textsuperscript{[3]} is one of the most representative algorithms in online model-based methods.

\begin{itemize}
  \item Initialization
  \item Build models for objectives by Gaussian process (Kriging)
  \item Acquisition function: $[\text{LCB}_1, \text{LCB}_2, \ldots, \text{LCB}_m]$
  \item Internal optimization by modified NSGA-II to obtain the sample point $x$
  \item Simulate and add to the database
  \item Pareto front
\end{itemize}


• Sacrifice \textbf{the time spent on the model} for simulation time.
Motivation

• There are two problems remains to be solved in online model-based methods.
• 1. The time spent on the model is comparable to or even exceed the simulation time in online model-based methods.
• 2. Most online model-based methods can not deal with constrained problems.

• The motivation of this work is to develop an online model-based method which shortens the time spent on the model.
1. Introduction
2. Motivation
3. Proposed SILE algorithm
4. Experimental results
5. Conclusion
Proposed SILE algorithm

- Time complexity of Kriging model
- Training process: the goal is to find the fittest hyperparameters $\hat{\theta}$ for observed points. The time complexity is $O(T_1 \cdot n^3)$. $T_i$ is the number of evaluating (1)
  \[ \hat{\theta} = \text{argmax} \left( -\frac{n}{2} \ln \hat{\sigma}^2 - \ln |R| \right) \]
  \[
  \begin{cases}
    \hat{\mu} = \frac{1^T R^{-1} y}{1^T R^{-1} 1} \\
    \hat{\sigma} = \frac{(y - 1\hat{\mu})R^{-1}(y - 1\hat{\mu})}{n} \\
    R(x, x') = \exp\left(-\sum_{i=1}^{d} \theta_i (x_i - x_i')^2\right)
  \end{cases}
  \]

- Prediction process: Given estimated $\hat{\theta}$ and calculated $R^{-1}$, one can predict the performances at any untested point. The time complexity is $O(T_2 \cdot n^2)$.
  \[
  \begin{cases}
    \hat{y}(x) = \hat{\mu} + r^T R^{-1} (y - 1\hat{\mu}) \\
    \hat{s}^2(x) = \hat{\sigma}^2 \left[ 1 - r^T R^{-1} r + \frac{(1 - 1^T R^{-1} r)^2}{1^T R^{-1} 1} \right]
  \end{cases}
  \]
Proposed SILE algorithm

- We propose an efficient Kriging-based constrained multi-objective evolutionary algorithm for analog circuit synthesis via self-adaptive incremental learning (SILE).

\[
\begin{align*}
\text{Total optimization time} & \quad \text{Simulation time} \\
\text{Training time of the model} & \quad \text{Prediction time of the model} \\
\end{align*}
\]

The prescreening strategy

- Self-adaptive strategy: \( O(T_1 \cdot n^3) \rightarrow O(1 \cdot n^3) \rightarrow O(n^2) \)
- The Incremental learning technique: No internal optimization

\[
\begin{align*}
O(T_1 \cdot n^3) & \quad \Rightarrow \quad O(1 \cdot n^3) & \quad \Rightarrow \quad O(n^2) \\
O(T_2 \cdot n^2) & \quad \Rightarrow \quad O(1 \cdot n^2) \\
\end{align*}
\]
Proposed SILE algorithm

• Incremental learning technique

• How to calculate new $\tilde{R}^{-1}$ from $R^{-1}$ of the old model?

\[
\tilde{R} = \begin{bmatrix}
R_{11} & R_{12} & \cdots & R_{1n} \\
R_{21} & R_{22} & \cdots & R_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
R_{n1} & R_{n2} & \cdots & R_{nn}
\end{bmatrix}
\begin{bmatrix}
R_{1(n+1)} \\
R_{2(n+1)} \\
\vdots \\
R_{n(n+1)}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
R & A \\
A^T & B
\end{bmatrix}
\]

\[
\tilde{R}^{-1} = \begin{bmatrix}
R^{-1} + R^{-1}AC^{-1}A^TR^{-1} & -R^{-1}AC^{-1} \\
-C^{-1}A^TR^{-1} & C^{-1}
\end{bmatrix}
\]

\[
C = B - A^TR^{-1}A
\]

• $C$ is a $1 \times 1$ matrix. The time complexity of computing $C^{-1}$ is $O(1)$.

• Now that $R^{-1}$ is known, the time complexity of $\tilde{R}^{-1}$ is $O(n^3) \rightarrow O(n^2)$
Proposed SILE algorithm

- $\tilde{R}$ is a symmetry positive definite matrix $\leftrightarrow \tilde{R} = \tilde{L}\tilde{L}^T$

- How to calculate new $\tilde{L}$ from $L$ of the old model?

\[
\tilde{R} = \tilde{L}\tilde{L}^T + \tilde{L} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}
\]

\[
\begin{align*}
\tilde{R} &= \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \\
&= \begin{bmatrix} L_{11}L_{11}^T & \left(L_{11}L_{21}^T + L_{22}L_{22}^T\right) \\ L_{21}L_{21}^T & L_{22}L_{22}^T \end{bmatrix} \\
&= \begin{bmatrix} R & A \\ A^T & B \end{bmatrix}
\end{align*}
\]

- the time complexity of $L^{-1}A$ is $O(n^2)$ by using back substitution method

- The time complexity of $\text{Chol}(B - A^T L^{-T} L^{-1}A)$ is $O(1)$.

- Now that $L$ is known, the time complexity of $\tilde{L}^{-1}$ is $O(n^3) \rightarrow O(n^2)$
Proposed SILE algorithm

- Self-adaptive strategy
- 1. In most cases, we build models with incremental Kriging model without updating hyperparameters. We only learn hyperparameters with Kriging model under a specific number of simulations.
- 2. In the early stage of optimization, we need to update hyperparameters more frequently.
- 3. In the later stage of optimization, we lower the frequency to update hyperparameters.
Proposed SILE algorithm

- Self-adaptive strategy
- As a result, hyperparameters are updated every $H$ generations. $H$ is adaptively adjusted based on the number of simulations, $N$

\[
H = \left\lfloor \frac{N - \lambda}{N_{\text{max}} - \lambda} (H_{\text{max}} - H_{\text{min}}) \right\rfloor + H_{\text{min}}
\]

- $\lambda$: the initial sample number
- $N_{\text{max}}$: the maximum number of simulations
- $H_{\text{max}}$: upper bound for $H$
- $H_{\text{min}}$: lower bound for $H$
- For $\lambda = 100$, $N_{\text{max}} = 200$, $H_{\text{min}} = 10$, $H_{\text{max}} = 19$, hyperparameters only update when $N = 100, 110, 121, 133, 146, 160, 175, 192$. 
Proposed SILE algorithm

- Self-adaptive strategy

The area under a curve is the total training time in the process of optimization.

- At small spikes, we use Kriging model to optimize hyperparameters.

- In the rest number of simulations, incremental Kriging model is used.
Proposed SILE algorithm

The framework of prescreening

Reduce the training time of the model

Reduce the prediction time of the model

SILE algorithm

Initialization

Stopping criterion

Self-adaptive strategy

Incremental Kriging model

Select the $l$ best design. Generate $l$ offspring

Prescreen the most promising one in offspring

Simulate and add it to the database

MOBO algorithm

Initialization

Output Pareto front in the feasible region

Stopping criterion

Acquisition function (LCB)

Internal optimization by modified NSGA-II

Locate the most promising one in the model

Simulate and add it to the database
Proposed SILE algorithm

- More details
- \( x \) is said to constraint-dominate \( y \) if the following condition holds:
  1) if \( CV(x) = 0 \) and \( CV(y) = 0 \), \( \forall i \in \{1, 2, \ldots, m\} \) such that \( f_i(x) \leq f_i(y) \) and \( \exists j \in \{1, 2, \ldots, m\} \) such that \( f_i(x) < f_i(y) \)
  2) otherwise, \( CV(x) < CV(y) \)

\[
\text{minimize} \quad f_1(x), f_2(x), \ldots, f_m(x) \\
\text{s.t.} \quad g_i(x) < 0 \quad \forall i \in 1,2,\ldots,p
\]

\[
CV(x) = \sum_{i=1}^{p} \max(g_i(x), 0)
\]
1. Introduction
2. Motivation
3. Proposed SILE algorithm
4. Experimental results
5. Conclusion
Experimental results

• A two-stage amplifier, 11 design variables, 180nm
• 15 corners: $-40^\circ C, 27^\circ C, 85^\circ C$ and tt, ss, ff, fs, sf

\[
\text{minimize} \ (-\text{Gain}, -\text{UGBW}, -\text{PM})
\]
\[
s.t. \ \text{PM} > 60^\circ
\]
**Experimental results**

- A two-stage amplifier, 11 design variables, 180nm

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>SILE</th>
<th>MOBO</th>
<th>NSGA-II</th>
<th>MOEA/D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Gain(dB)</td>
<td>81.58</td>
<td>81.33</td>
<td>80.31</td>
<td>81.17</td>
</tr>
<tr>
<td>Max UGBW(MHz)</td>
<td>19.64</td>
<td>18.86</td>
<td>16.95</td>
<td>17.68</td>
</tr>
<tr>
<td>Max PM(°)</td>
<td>93.10</td>
<td>92.71</td>
<td>92.84</td>
<td>85.99</td>
</tr>
<tr>
<td>Mean HV</td>
<td>14821</td>
<td>13951</td>
<td>13880</td>
<td>13709</td>
</tr>
<tr>
<td>Median HV</td>
<td>14726</td>
<td>14038</td>
<td>14231</td>
<td>13582</td>
</tr>
<tr>
<td>Max HV</td>
<td>16268</td>
<td>14624</td>
<td>15209</td>
<td>16190</td>
</tr>
<tr>
<td>Min HV</td>
<td>13484</td>
<td>12526</td>
<td>10597</td>
<td>12224</td>
</tr>
<tr>
<td>$N_{max}$</td>
<td>400</td>
<td>400</td>
<td>4000</td>
<td>4000</td>
</tr>
<tr>
<td>Training time/s</td>
<td>56.54</td>
<td>1183.76</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Prediction time/s</td>
<td>2.53</td>
<td>798.26</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Simulation time/s</td>
<td>1423.37</td>
<td>1492.73</td>
<td>15051.81</td>
<td>15170.12</td>
</tr>
<tr>
<td>Total time/s</td>
<td>1490.35</td>
<td>3476.15</td>
<td>15052.09</td>
<td>15172.64</td>
</tr>
</tbody>
</table>

- Compared with MOBO, SILE reduce the training time by **95%** and the prediction time by **99.7%**. SILE shows a speedup of **10X** in terms of the total time while achieving better results.
Experimental results

- A fully differential operational amplifier, 21 design variables, 65nm

minimize \((-Gain, -GBW)\)

s.t. \(PM_{dm} > 60^\circ\)

\(PM_{cm} > 50^\circ\)

\(GBW_{cm} > 1.2GBW_{dm}\)

\(SR > 100V/\mu s\)

\(overshoot_c < 15\%\)

\(tset_c < 50ns\)

\(overshoot_r < 15\%\)

\(tset_r < 50ns\)
Experimental results

- A fully differential operational amplifier, 21 design variables, 65nm.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>SILE</th>
<th>MOBO</th>
<th>NSGA-II</th>
<th>MOEA/D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Gain(dB)</td>
<td>70.89</td>
<td>70.37</td>
<td>69.92</td>
<td>68.10</td>
</tr>
<tr>
<td>Max GBW(MHz)</td>
<td>770</td>
<td>639</td>
<td>685</td>
<td>738</td>
</tr>
<tr>
<td>Mean HV</td>
<td>10684</td>
<td>7678</td>
<td>9709</td>
<td>9368</td>
</tr>
<tr>
<td>Median HV</td>
<td>10892</td>
<td>7715</td>
<td>9565</td>
<td>9344</td>
</tr>
<tr>
<td>Max HV</td>
<td>11676</td>
<td>9235</td>
<td>10664</td>
<td>10160</td>
</tr>
<tr>
<td>Min HV</td>
<td>9239</td>
<td>5799</td>
<td>9102</td>
<td>8724</td>
</tr>
<tr>
<td>$N_{max}$</td>
<td>400</td>
<td>400</td>
<td>4000</td>
<td>4000</td>
</tr>
<tr>
<td>Training time/s</td>
<td>382.44</td>
<td>8084.91</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Prediction time/s</td>
<td>6.29</td>
<td>3014.34</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Simulation time/s</td>
<td>804.43</td>
<td>837.67</td>
<td>7420.74</td>
<td>7234.19</td>
</tr>
<tr>
<td>Total time/s</td>
<td>1200.90</td>
<td>11941.38</td>
<td>7420.97</td>
<td>7235.87</td>
</tr>
</tbody>
</table>

- SILE reduces the training time by **95%**, the prediction time by **99.8%** while achieving much better PF. There is a **6X** speedup over NSGA-II and MOEA/D regarding the total time.
Contents

- 1. Introduction
- 2. Motivation
- 3. Proposed SILE algorithm
- 4. Experimental results
- 5. Conclusion
Conclusion

- We propose an efficient Kriging-based constrained multi-objective evolutionary algorithm for analog circuit synthesis via self-adaptive incremental learning.

- Experimental results on two real-world circuits demonstrate that compared with MOBO, our method can reduce the training time of Kriging model by 95% and the prediction time by 99.7%. Compared with NSGA-II and MOEA/D, the proposed method can achieve up to 10X speed up.
Thanks for your attention!