KOSIM: A Knowledge-oriented Derivative-free Subspace Method Based on Inexact Model for Inverse Lithography Problems

ABSTRACT

This paper proposes a novel derivative-free optimization (DFO) method for solving the inverse problem of Inverse lithography technology (ILT). The proposed method is a kind of subspace method which utilizes inexact model and some prior information of the objective function. The function evaluation times per iteration of the new method is $O(1)$. Numerical experiments on academic examples and application problems demonstrate the efficiency and robustness of the new method. In particular, a specific ILT problem with up to 10^6 dimensions can be solved to over 80% reduction of L_2 square error within 500 function evaluations.

KEYWORDS

Inverse lithography technology (ILT), Large-scale derivative-free optimization, Subspace method, Knowledge-oriented method

ACM Reference Format:

. 2023. KOSIM: A Knowledge-oriented Derivative-free Subspace Method Based on Inexact Model for Inverse Lithography Problems. In *Proceedings of DAC 2024: 61st IEEE/ACM Design Automation Conference. (DAC'24).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Optical lithography is a crucial step in semiconductor industry. The procedure of optical lithography is to print mask layouts onto the wafer through an optical system. Because of the influence of interference and diffraction at the tiny scale of circuit device size, the image on wafer distorted very much. To resolve the distortion, quite a few approaches are proposed and used extensively in optical lithography [10]. Optical proximity correction (OPC) dominates these approaches. With optical proximity correction, the mask layout is adjusted such that the output pattern approximates the target. A mainstream technique of OPC is to discrete the mask into a matrix and the imaging procedure is characterized by a pixel-wise imaging function, then the OPC process is modeled as an inverse problem. This technique is called inverse lithography technology (ILT) [11]. Usually, the inverse problem is formulated as a non-convex optimization problem with respect to the matrix elements.

In recent years, quite a few methods are developed to solve ILT problems, such as classical attempts [3, 8, 12] and several current works [6, 14, 15, 17]. In general, these methods are based on approximate forward imaging modeling to reduce the computational cost. The imaging procedure is modeled as explicit formulation whose gradients are available. However, the inexactness of the imaging procedure may results in unreliability of the adjusted layouts. In practice, the computation of forward imaging may need numerical simulations and probably be packed as softwares. Therefore, we consider the imaging function to be a blackbox which lacks gradient information and develop general DFO methods in this paper.

Derivative-free optimization plays a critical role in many application scenarios nowadays, especially in chip industry. A general unconstrained DFO problem can be formulated by $\min_{x\in\mathbb{R}^n} f(x)$, where the gradient of the objective function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is unavailable. As DFO does not admit traditional optimality conditions, the best function value f_{best} is used to measure the quality of solution. To evaluate the efficiency of algorithms, another important index is the total number of function value evaluations, abbreviated as NF, since function value evaluation is regarded to be costly in DFO. Model based approaches dominate the development of DFO for decades for their high efficiency (see Implicit Filtering [7], NEWUOA [13], ORBIT [16]). The computational complexity of model based approaches per iteration reaches $O(n^2)$ and $O(n^3)$ some times, which become costly when the size of problem is large. Moreover, any model-based method needs at least $O(n)$ function evaluations for initializing the model, which is too many to be accepted in ILT scenarios. Most of the existing approaches have been reported to solve problems with at most thousands of variables. To meet the growing demand for solving large-scale DFO, subspace techniques (see [9, 18]) are borrowed to this area, for example, OMoRF [5]; RSDFO [2]. However, these approaches only increase the scales to at most ten thousands.

In this paper, we propose KOSIM, a general subspace method for solving DFO problems in ILT. Firstly, compared with existing subspace DFO methods, KOSIM has a novel way in constructing subspaces, which makes our method appreciably efficient. Specifically, we develop a projection technique for computing an inexact gradient. This step is knowledge-oriented when some prior information of the objective function can be analyzed. A separable quadratic model is maintained in each iterations using an economic way in updating the sampling points. The model does not need to be fully linear [2], so we call it an inexact model. The above two techniques help KOSIM construct good subspaces while it only evaluates $O(1)$ function values in each iteration. Secondly, KOSIM only produces $O(n)$ computational cost in each iteration, so it is suitable for large-scale problems. Finally, we demonstrate the efficiency and robustness of KOSIM through comprehensive numerical experiments. By utilizing priors independent of the imaging function, KOSIM is able to solve ILT problems within constant number of function evaluations not related to the mask size. In particular, KOSIM can solve an ILT problem with up to 10^6 dimensions to over 80% reduction of L_2 square error within 500 function evaluations.

The rest of this paper is organized as follows. section 2 lists some preliminaries. section 3 introduces the framework and details

DAC'24, June 23-27, San Francisco, CA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

of the proposed method. We present comprehensive numerical experiments in section 4. Finally, we conclude our paper in section 5.

2 PRELIMINARIES

In this section, we introduce the background and formulate the DFO problem. The most common objective function of the ILT problem is the misfit between the image on wafer and the target pattern. Assume the mask is discrete into matrix $U \in \mathbb{R}^{N\times N}$, $U_{ij} \in \{0,1\}$ and $\mathcal{I} : \mathbb{R}^{N\times N} \to \mathbb{R}^{N\times N}$ is a black-box imaging function. The target pattern is abbreviated by U_0. Then the optimization problem is formulated by

$$\begin{aligned} \min_{U\in\mathbb{R}^{N\times N}} \ & ||\mathcal{I}(U) - U_0||_F^2, \\ s.t.\ & U_{ij} \in \{0,1\}. \end{aligned} \tag{1}$$

In general, we may not directly accept an arbitrary solution of the above optimization problem for industry production, because of the irregularity of the corrected mask. In this paper, we for the first time use mean filtering to reduce the noise of the corrected mask and formulate a modified objective function. In order to turn an arbitrary real matrix $U \in \mathbb{R}^{N\times N}$ into an acceptable pattern in industrial manufacturing, $N_t \in \mathbb{N}$ times of convolution by core

$$\hat{H} := \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}. \tag{2}$$

is acted on U and then it is truncated by

$$\mathcal{T} := \begin{cases} 1, & x \geq 0.5 \\ 0, & x < 0.5 \end{cases}. \tag{3}$$

Operator $\mathcal{M} : \mathbb{R}^{N\times N} \to \mathbb{R}^{N\times N}$ is defined by

$$\mathcal{M}(U) := \mathcal{T}\left(\left(\hat{H}*\right)^{(N_t)} U\right). \tag{4}$$

Define matrization operator

$$\mathcal{U} : \mathbb{R}^{N^2} \to \mathbb{R}^{N\times N},\ x \mapsto (x_{Ni+j})_{ij}. \tag{5}$$

Then the modified formulation of the optimization problem is

$$\min_{x\in\mathbb{R}^{N^2}} \ ||\mathcal{I}(\mathcal{M}(\mathcal{U}(x))) - U_0||_F^2. \tag{6}$$

For the evaluation of the quality of solutions, we give some definitions.

Definition 2.1 (EPE). The edge placement error (EPE) is defined as a picture

$$EPE := |\mathcal{I}(U) - U_0|. \tag{7}$$

Definition 2.2 (L_2 square error). The L_2 square error can be calculated by

$$\left\|\mathcal{U}^{-1}(EPE)\right\|_2^2, \tag{8}$$

i.e., square of L_2 norm of the vectorized EPE.

Algorithm 1: **K**nowledge-**O**riented **S**ubspace Method based on **I**nexact **M**odel (**KOSIM**)

Input: objective function $f : \mathbb{R}^n \to \mathbb{R}$, prior generator $\mathcal{D} : \mathbb{R}^n \to \mathbb{R}^n$.
Output: $x^{(k)}, f\left(x^{(k)}\right)$.

Choose an initial guess $x^{(0)} \in \mathbb{R}^n$, and set $k := 0$.
Initialize $m^{(0)}$.
while *certain termination criterion is not satisfied* **do**
  Determine random subspace $\mathcal{S}_g^{(k)} \subset \mathbb{R}^n$ satisfying $\mathcal{D}\left(x^{(k)}\right) \in \mathcal{S}_g^{(k)}$ (if $\mathcal{D} \neq$ None).
  Approximately compute $g^{(k)} := \mathcal{P}_{\mathcal{S}_g^{(k)}} \nabla f\left(x^{(k)}\right)$.
  Compute a solution $d^{(k)}$ of $m^{(k)}$.
  Define subspace $\mathcal{S}^{(k)} := \text{span}\left(g^{(k)}, d^{(k)}, s^{(k-1)}\right)$.
  Inexactly solve $\min_{s\in\mathcal{S}^{(k)}} f\left(x^{(k)} + s\right)$ and obtain $s^{(k)}$.
  Update $x^{(k+1)} := x^{(k)} + s^{(k)}$.
  Update $m^{(k)}$ to $m^{(k+1)}$.
  Set $k := k + 1$.

3 ALGORITHM

3.1 Motivation and framework

The framework of KOSIM follows a conventional subspace structure [9, 18]. The most important step in the algorithm is the construction of subspaces, which mainly includes two techniques. Firstly, a projection technique for computing inexact gradients is developed. This step is knowledge-oriented and we will detail this in section 3.2. The second technique, inspired by model-based approaches, is to use sampling points in an economical way while maintaining a separable quadratic model in each iteration. This technique will be detailed in section 3.3. After the inexact gradient $g^{(k)} \in \mathbb{R}^n$ and the solution $d^{(k)} \in \mathbb{R}^n$ of the model are computed, together with the difference between current and the last iterates $s^{(k-1)}$, the subspace can be constructed by $\mathcal{S}^{(k)} := \text{span}\left(g^{(k)}, d^{(k)}, s^{(k-1)}\right)$. Then we solve the subproblem restricted in the subspace inexactly. To achieve this, we build a complete positive-definite quadratic model in the subspace by interpolation and minimize it to obtain the next iterate. The framework is stated as Algorithm 1.

3.2 The knowledge-oriented inexact gradient

To compute an inexact gradient $g^{(k)}$, we determine an m dimensional subspace $\mathcal{S}_g^{(k)}$ and approximately calculate the projection $\mathcal{P}_{\mathcal{S}_g^{(k)}} \nabla f\left(x^{(k)}\right)$. Here $m << n$ and we usually set m to be $O(1)$. The approximate calculation of $\mathcal{P}_{\mathcal{S}_g^{(k)}} \nabla f\left(x^{(k)}\right)$ invokes a simple finite difference with step length $\rho^{(k)}$, which is adaptively chosen. When prior information is available, i.e., a prior generator $\mathcal{D}$ is inputed, we set $\mathcal{D}\left(x^{(k)}\right) \in \mathcal{S}_g^{(k)}$.

3.3 The separable quadratic inexact model

Different from conventional quadratic-model-based approaches, a separable model is constructed in KOSIM for lower computational complexity. Furthermore, the model in our method does not need to be fully-linear. Assume $m^{(k)}(x)$ has formulation

$$m^{(k)}(x) = \sum_{i=1}^{n} m_i^{(k)}(x_i) - (n-1) f\left(x^{(k)}\right), \tag{9}$$

where $m_i^{(k)}(u)$ is a one dimensional quadratic model approximating $f\left(x^{(k)} + \left(u - x_i^{(k)}\right) e_i\right)$, $i = 1, ..., n$. An intuitive idea for the construction of $m_i^{(k)}$ is to use three sampling points on $x^{(k)} + \text{span}(e_i)$ interpolating f. Nevertheless, this results in an $O(n)$ function value evaluation cost. For economy, we consider to use sampling points off the line $x^{(k)} + \text{span}(e_i)$ so that historical sampling points can be utilized. Assume three sampling points for the ith model $m_i^{(k)}$ consist $\mathcal{Y}_i^{(k)} := \left\{y^{(k,i)}, z^{(k,i)}, w^{(k,i)}\right\}$, then the interpolation follows

$$\begin{aligned} m_i^{(k)}(u) := & \sum_{cyc} f\left(y^{(k,i)}\right)\left(u - z_i^{(k,i)}\right)\left(u - w_i^{(k,i)}\right) \\ & \Big/ \left[\left(y_i^{(k,i)} - z_i^{(k,i)}\right)\left(y_i^{(k,i)} - w_i^{(k,i)}\right)\right]. \end{aligned} \tag{10}$$

3.4 Algorithm details

Initializing and Updating the sampling points. With the help of the descriptions in section 3.3, we now state the initializing and updating procedure of the sampling points. For initialization, only two function values need to be evaluated. After the calculation of $f\left(x^{(0)} \pm r\mathbf{1}\right)$, we set

$$\begin{aligned} y^{(0,i)} &= z^{(0,i)} = w^{(0,i)} \\ &:= x^{(0)} + (i-2) r\mathbf{1}, \ i = 1, 2, 3. \end{aligned} \tag{11}$$

Here $r > 0$ can be arbitrarily set, and $\mathbf{1} := (1, ..., 1)^\top$. When a new iterate $x^{(k+1)}$ is obtained, one of the sampling points $u^{(k,i)}$ in $\mathcal{Y}_i^{(k)}$ is replaced ($i = 1, ..., n$) and $\mathcal{Y}_i^{(k)}$ is updated to $\mathcal{Y}_i^{(k+1)}$. The sampling points to be replaced are chosen by

$$u^{(k,i)} := \min_{u \in \mathcal{Y}_i^{(k)}} \frac{\left|u_i - x_i^{(k+1)}\right|}{f(u) - f\left(x^{(k+1)}\right)}, \ i = 1, ..., n, \tag{12}$$

The inspiration of this mechanism is that, for sampling points on the same contour of f, the one with larger distance on the ith direction is more likely to have a gradient parallel to e_i, then its function value is more likely to be close to its projection on $x^{(k+1)} + \text{span}(e_i)$. A simple illustration of this procedure is depicted in Figure 1.

Solving the model. After the models are constructed, the model directions are calculated by

$$d^{(k)} := \min_{d \in \Omega^{(k)}} m^{(k)}\left(x^{(k)} + d\right). \tag{13}$$

Here, $\Omega^{(k)}$ is a box region defined by

$$\Omega^{(k)} := \prod_{i=1}^{n} \left[-r_i^{(k)}, r_i^{(k)}\right], \tag{14}$$

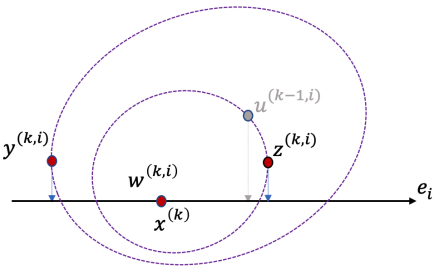

Figure 1: This figure illustrates the updating procedure of sampling points. For example, $u^{(k-1,i)}$ is the point to be replaced by $x^{(k)}$ and $z^{(k,i)}$ is remained, because on the same contour, $\nabla f\left(z^{(k,i)}\right)$ is more likely to be parallel to e_i.

where

$$\begin{aligned} r_i^{(k)} := \max & \left\{ \left|y_i^{(k,i)} - x_i^{(k)}\right|, \left|z_i^{(k,i)} - x_i^{(k)}\right|, \right. \\ & \left. \left|w_i^{(k,i)} - x_i^{(k)}\right| \right\}, \ i = 1, ..., n. \end{aligned} \tag{15}$$

Solving the subproblem. For solving the subproblem in the subspace, we construct a complete three-dimensional quadratic model $q(s)$ approximating $f\left(x^{(k)} + B^{(k)}s\right)$ by interpolation using 10 sampling points $s^{(k,1)}, ..., s^{(k,10)}$ in the subspace. Here, $B^{(k)} \in \mathbb{R}^{n\times 3}$ stands for the basis matrix of $\mathcal{S}^{(k)}$ after a Gram-Schmidt procedure. Then we solve $\min_{s\in\mathbb{R}^3}\left[q(s) + \lambda||s||^2\right]$ and obtain $s_q^{(k)}$, where $\lambda \geq 0$ satisfies $\nabla^2 m\left(x^{(k)}\right) + \lambda I > 0$. Then

$$\begin{aligned} s^{(k)} := \arg\min & \left\{ f\left(x^{(k)}\right), f\left(x^{(k)} + B^{(k)}s^{(k,1)}\right), ..., \right. \\ & \left. f\left(x^{(k)} + B^{(k)}s^{(k,10)}\right), f\left(x^{(k)} + B^{(k)}s_q^{(k)}\right) \right\}. \end{aligned} \tag{16}$$

Updating the finite-difference step length. For updating the finite-difference step length $\rho^{(k)}$, we choose a scaling factor $\sigma^{(k)} > 0$ satisfying $0 < \sigma_l \leq \sigma^{(k)} \leq \sigma_u$ and update

$$\rho^{(k+1)} := \max\left\{\frac{1}{2}\rho^{(k)}, \sqrt{\left[f\left(x^{(k)}\right) - f\left(x^{(k+1)}\right)\right] \Big/ \sigma^{(k)}}\right\}. \tag{17}$$

Remark 1. *In practice, we find*

$$\sigma^{(k)} := \max\left\{10^{-6}, \min\left\{10^{6}, \left\|\nabla^2 m^{(k)}\right\|_2\right\}\right\} \tag{18}$$

a good choice and it is set by default.

Table 1 shows the computational complexity and the number of function evaluations in each iteration of Algorithm 1. We can conclude that our proposed method has only $O(n)$ computational cost and $O(1)$ function evaluation cost per iteration.

Step	Complexity	NF
Initializing the model	$O(n)$	2
Updating the model	$O(n)$	0
Computing the inexact gradient	$O(n)$	m
Solving the subproblem	$O(n)$	10

Table 1: Computational complexity and NF of each step in Algorithm 1

n	$\|\|Ax-b\|\|/\|\|b\|\|$	iteration number		walltime		speed-up ratio
		CG	KOSIM	CG	KOSIM	
100	9.67e-04	13307	464	2.11s	0.14s	14.45×
200	5.36e-02	40000	765	7.36s	0.43s	16.97×
400	7.50e-04	66537	1349	17.30s	1.64s	10.51×
800	1.38e-03	103658	2093	114.96s	7.38s	15.55×

Table 2: Comparison of KOSIM with gradient information and CG

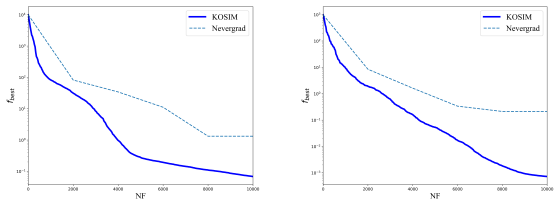

Figure 2: Comparing KOSIM and Nevergrad on problem constructed by CHROSEN (left) and SINQUAD (right) ($n = 10000$)

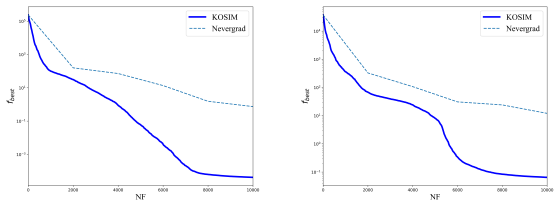

Figure 3: Comparing KOSIM and Nevergrad on problem constructed by CHROSEN (left) and SINQUAD (right) ($n = 50000$)

4 EXPERIMENTAL RESULTS

This section presents numerical experiments on some benchmark functions as well as a few examples from ILT to demonstrate the performance of KOSIM. First, we compare KOSIM with some state-of-the-art DFO algorithms on several benchmark functions without any prior to show the good efficiency of KOSIM as a DFO method. Second, we test KOSIM on these benchmark function with derivative information comparing with traditional gradient-based methods to show the acceleration of KOSIM as a gradient-based method. Finally, we test KOSIM on application problems in ILT with and without prior information, comparing with other state-of-the-art DFO algorithms. Our code is implemented in C, and packed as a Python interface. All the tests of KOSIM were performed on a Lenovo ST8810 cluster.

4.1 Comparison on academic examples

First, we compare KOSIM with the conjugate gradient algorithm implemented by `minimize` in Scipy. We construct a general linear least-square problem for this test. Thus, the objective function is

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\,||Ax-b||^2, \tag{19}$$

where A is a random $n\times n$ dense matrix and b is a random n dimensional vector. The dimension of $\mathcal{S}_g^{(k)}$ in this test is set as $m := 1$. The initial points are set to be 0. The test problem is solved by `minimize` and obtain f_{best}, then f_{best} is set as the target function value for KOSIM for termination. Total iteration numbers and wall-clock time of both solvers are recorded and reported in Table 2. From the results in Table 2 we can observe that the quadratic model in KOSIM obviously accelerates the algorithm.

Next, we compare KOSIM with Nevergrad platform [1], which collects a large number of state-of-the-art DFO methods, including stochastic or heuristic methods. Nevergrad can adaptively choose the most suitable algorithm for the problem by using solver option NGOpt. We choose two representative benchmark functions, CHROSEN and SINQUAD, from CUTEr [4] for experiments. The objective functions are formulated by $f(x) := f_{ben}(Ax)$, where f_{ben} is either of the benchmark functions, and A is a random $t\times n$ matrix. The dimension of the problem and $\mathcal{S}_g^{(k)}$ are set as $n := 10000, 50000$, $m := 20$ respectively and the initial points are set randomly in $[-10,10]^n$. In this test, integer t is chosen to be 100. From Figure 2 and Figure 3 we can clearly observe that KOSIM has better efficiency than Nevergrad on both problems.

4.2 Choosing prior generator for ILT problems

Before we test KOSIM on ILT problems, we propose a practicable prior generator for problem (6). An intuitive observation is that, for mask pattern $U \in \mathbb{R}^{N\times N}$, when $\mathcal{I}(U)_{ij} < U_{0ij}$, we tend to increase the value at U_{ij}; otherwise we tend to reduce it. Thus, a prior generator $\mathcal{D}$ can be formulated by

$$\mathcal{D}(x) := \mathcal{U}^{-1}\left(U_0 - \mathcal{I}\left(\mathcal{M}\left(\mathcal{U}\left(x\right)\right)\right)\right). \tag{20}$$

In addition, no internal information of the imaging function is used in this prior. It is only related to the geometry structure of the mask layout.

4.3 Comparison on application examples in ILT

Finally, we test KOSIM on ILT problems described in section 2. We implement an imaging function according to the models in [10], and the number of convolution times is set as $N_t := 50$. We run KOSIM with and without this prior together with Nevergrad on several ILT problems which has different mask size and mask patterns and report the results in Figure 4, Figure 5 and Figure 6. A clear observation from Figure 4 is that, KOSIM has better efficiency than Nevergrad in these two problems, and the prior information accelerate KOSIM to a large extent. Furthermore, according to the correction results reported in Figure 5 and Figure 6, KOSIM with prior gives a correction of high accuracy of mask within 500 function

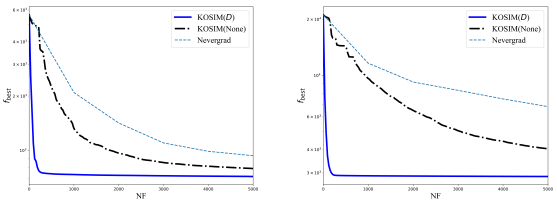

Figure 4: This figure shows the comparison of KOSIM with and without prior and Nevergrad on two ILT problems with mask size 200×200 and 400×400. Best function values within 5000 function evaluations are reported.

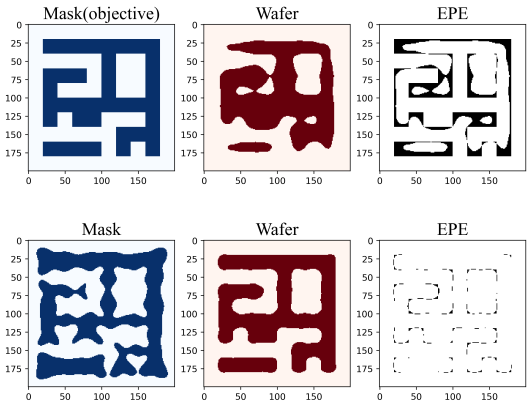

Figure 5: This figure shows the correction result of KOSIM with prior on the same ILT problem with mask size 200×200 in Figure 4 within 500 function evaluations. The 6 subfigures respectively show the mask patterns, images and EPEs.

evaluations. This is a considerable result since the the dimensions of the problems are much larger than 500.

We run KOSIM on ILT problems with mask size 400×400, 600×600, 800×800, and 1000×1000 within 500 function evaluations and report the correction results in Table 3 and Figure 7. We can clearly observe that no matter how large the mask size is, KOSIM can solve the ILT problem to over 80% L_2 square error reduction within 500 function evaluations.

5 CONCLUSION

In this paper, we propose a general subspace DFO method, KOSIM, for solving DFO problems in ILT. Several techniques are developed for KOSIM in computing knowledge-oriented inexact gradients, maintaining inexact models, constructing subspaces, and solving the subproblems. KOSIM evaluates only $O(1)$ function values and has only $O(n)$ computational complexity per iteration. Hence, KOSIM has a good efficiency and is able to solve large-scale DFO problems. Numerical results clearly show the advantages of

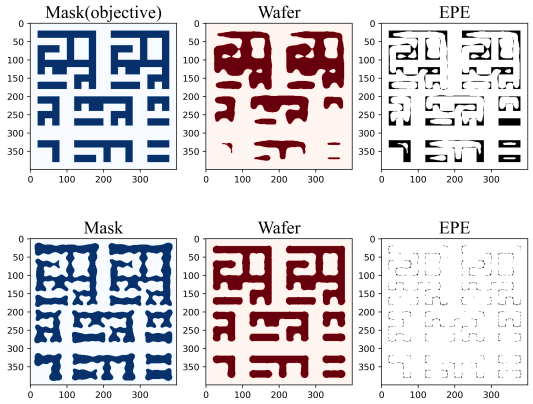

Figure 6: This figure shows the correction result of KOSIM with prior on the same ILT problem with mask size 400×400 in Figure 4 within 500 function evaluations. The 6 subfigures respectively show the mask patterns, images and EPEs.

N	n	L_2 square error reduction	NF
200	40000	89.8%	500
400	160000	87.0%	500
600	360000	85.6%	500
800	640000	84.1%	500
1000	1000000	84.8%	500

Table 3: Correction result of KOSIM with prior on ILT problems with different mask size within 500 function evaluations

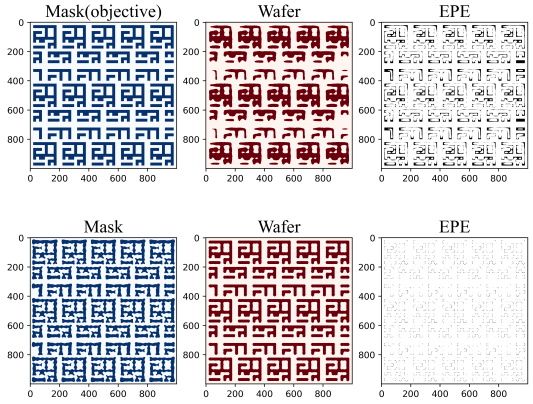

Figure 7: This figure shows the correction result of KOSIM with prior on an ILT problem with mask size 1000×1000 within 500 function evaluations. The 6 subfigures respectively shows the mask patterns, images and EPEs.

KOSIM in efficiency and robustness. With specific imaging function and readily available prior, large-scale ILT problems with up to 10^6 dimensions can be efficiently solved to high accuracy.

REFERENCES

[1] P. Bennet, C. Doerr, A. Moreau, J. Rapin, F. Teytaud, and O. Teytaud. 2021. Nevergrad. *ACM SIGEVOlution* (2021).

[2] C. Cartis and L. Roberts. 2021. Scalable Subspace Methods for Derivative-Free Nonlinear Least-Squares Optimization. *Mathematical Programming* (2021).

[3] J.-R. Gao, X. Xu, B. Yu, and D. Z. Pan. 2014. Mosaic: Mask optimizing solution with process window aware inverse correction. *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)* (2014), 1–6.

[4] N. Gould, D. Orban, and P. L. Toint. 2003. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Software* (2003).

[5] J. C. Gross and G. T. Parks. 2021. Optimization by moving ridge functions: derivative-free optimization for computationally intensive functions. *Engineering Optimization* 1 (2021), 1–23.

[6] B. Jiang, L. Liu, Y. Ma, B. Yu, , and E. F. Young. 2021. Neural-ilt 2.0: Migrating ilt to domain-specific and multi-task-enabled neural network. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2021).

[7] C. T. Kelley. 2011. *Implicit Filtering*. SIAM, Philadelphia. https://doi.org/10.1137/1.9781611971903

[8] F. Liu and X. Shi. 2011. An efficient mask optimization method based on homotopy continuation technique. *2011 Design, Automation & Test in Europe* (2011), 1–6.

[9] X. Liu, Z. Wen, and Y. Yuan. 2021. Subspace Methods for Nonlinear Optimization. *CSIAM Transactions on Applied Mathematics* 2 (2021), 585–651.

[10] X. Ma and R. Arce. 2010. *Computational Lithography*. John Wiley & Sons, Inc.

[11] L. Pang, Y. Liu, and D. Abrams. 2006. *Inverse lithography technology (ILT): What is the impact to the photomask industry?* Vol. 6283. International Society for Optics and Photonics.

[12] A. Poonawala and P. Milanfar. 2007. Mask design for optical microlithography—an inverse imaging problem. *IEEE Transactions on Image Processing* (2007), 774–788.

[13] M. J. D. Powell. 2006. The NEWUOA software for unconstrained optimization without derivatives. In *Large-Scale Nonlinear Optimization*, G Di Pillo and M Roma (Eds.). Nonconvex Optimization and Its Applications, Vol. 83. Springer, Boston, 255–297. https://doi.org/10.1007/0-387-30065-1_16

[14] I. Torunoglu, A. Karakas, E. Elsen, C. Andrus, B. Bremen, B. Dimitrov, and J. Ungar. 2010. A gpu-based full-chip inverse lithography solution for random patterns. *Design for Manufacturability through Design-Process Integration IV* (2010).

[15] Qijing Wang, Bentian Jiang, Martin D. F. Wong, and Evangeline F. Y. Young. 2022. A2-ILT: GPU Accelerated ILT with Spatial Attention Mechanism. In *Proceedings of the 59th ACM/IEEE Design Automation Conference* (San Francisco, California) *(DAC '22)*. Association for Computing Machinery, New York, NY, USA, 967–972.

[16] S. M. Wild and C. Shoemaker. 2013. Global convergence of radial basis function trust-region algorithms for derivative-free optimization. *SIAM Rev.* (2013).

[17] H. Yang, S. Li, Y. Ma Z. Deng, B. Yu, and E. F. Young. 2019. Gan-opc: Mask optimization with lithography-guided generative adversarial nets. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2019), 2822–2834.

[18] Y. Yuan. 2014. A review on subspace methods for nonlinear optimization. *Proceedings of International Congress of Mathematicians* Soeul, Korea (2014), 807–827.