UM - SJTU JOINT INSTITUTE
VE401 - Project 2

Model Fitting Analysis:
Calculation of Overlay Error on Photolithography

Group3

Jiajin Wu
Jinyou Kim
Yipeng Lin
Zhongqian Duan

December 1, 2021

Contents

Abstract

This paper is written to examine how a given data-set from the photo-lithography field can be fit into a proper model, by using several computer programs such as MATLAB, Python, etc. In this project, photo-lithography is introduced as the application for the data-fitting practice. The objective of this research is to discover the best model to relate given data and the outcome: Overlay errors. To fulfill the objective, firstly the basic model given is examined; then the optimal model is obtained through attempts from various software.

Keywords: Data analysis, Data Model.

1 Introduction

1.1 Background

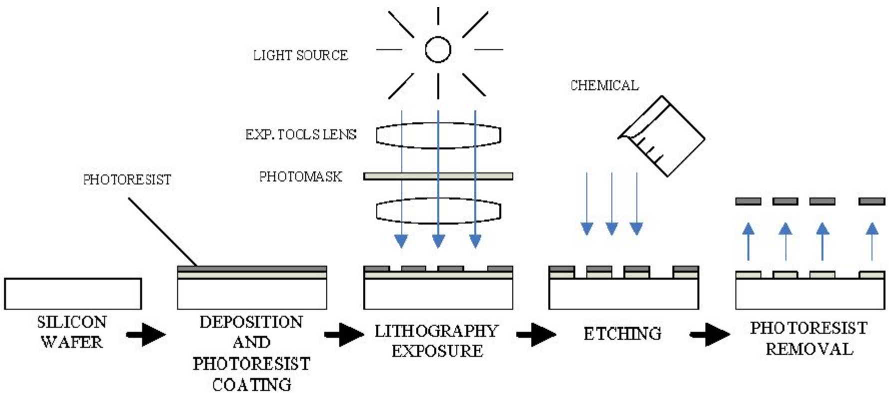


Figure 1: Photolithography process in semiconductor manufacturing.

Photolithography, also called optical lithography, is a semiconductor manufacturing process that uses light to print electric patterns on the integrated circuit [1]. This process can be compared to photography.

The process begins with patterns get defined on a photomask. The photomask act as the aperture of a camera, only allows the designed pattern of light to pass through its hole [3].

Then various lens adjusts the focus to magnify the size of a light pattern before it reaches the wafer. There are two layers on the silicon wafer. The photo-resist layer is located at the top and the deposition layer located beneath. When the wafer is exposed to the light source, the light-sensitive upper coating gets eliminated and leaves the dies of the defined pattern. This process is similar to the way how the film records photos in a camera [3].

Next, the chemicals etch the designed pattern on the deposition layer. The photo-resist material does not react with the etching chemicals and acts as the protector of the deposition layer underneath [3].

Finally, a wafer with the layer of designed deposition layer gets obtained after removing the photo-resist materials. Figure 1 exhibits the process with a flow diagram [3].

1.2 Process Control

Before the application of photolithography technology in the semiconductor manufacturing industry, the process should be acceptable for productivity. In other words, the exposed image on the wafer should be as accurate as possible to match what has been designed on the photomask. However, every process has some subtle variations (misalignment) resulting from the alignment system, projection lens, illumination system, or the inconsistencies in the wafer surface which may lower the quality of production. It's impossible to inspect the entire chip with a micro-probe because the electric circuit in the chip is too complicated for either human or machine to trace. Therefore, there should be some methodologies to avoid the problem and to find out the possible issues in this process, and here come the approaches for control overlay errors in semiconductor manufacturing [3].

There are many approaches to control the overlay error in a photolithography process, including the Statistical Process Control (SPC) which is used for control limit check, and the Advanced Process Control (APC) which is a kind of improved process control compared to the SPC system. The mathematical relationship between process parameters and output measurements (the overlay errors) is provided by a parameter model. There are many kinds of process parameters, including translation, rotation, expansion, trapezoid, distortion, magnification, etc [3],[2]. Figure 2 below illustrates some of the overlay errors generated from process parameters.

The APC module applies the parameter model and measurement samples to reverse engineering and find the possible input process parameters. In the feedback loop, these calculated (correctable) parameters will be compared with the real process parameters to adjust the input parameters in the next process [3].

1.3 Overlay Modeling

The parameter model is provided by the exposure tool vendors, but the task required in this project is to find out the best-fitted parameter model from the overlay error [3]. Overlay modeling: using the overlay error samples to establish a model that determines the contribution of process parameters to the total error, should be executed. The overlay errors can be decomposed into two parts: the systematic errors and non-systematic (random) errors. The systematic errors are correctable (by APC module) while the random errors are related to some unknown variations and cannot be corrected. Since the common overlay error patterns are difficult to distinguish with the additional random errors, the statistical regression can be applied to fit the data of source parameters to the observation of total overlay errors [2]. This process will be introduced in the next chapter.

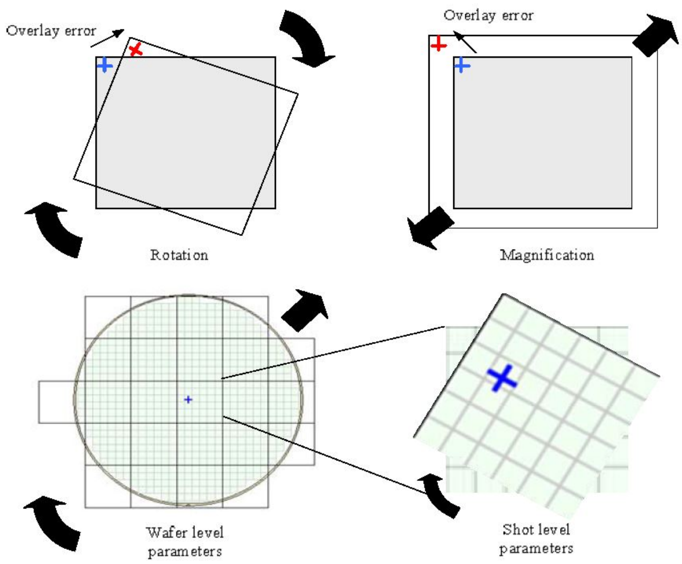


Figure 2: How overlay errors generated from process parameters.

2 Preparation for Fitting

The data set given contains 250 observations, making the data set into a $(250, 6)$ matrix. Respectively, the data presented do not have not enough dimensions to fit a proper model which can predict the overlay error well. Therefore, analyzing data and each variable is necessary for fitting the desired model.

2.1 Analysis on Physical Meaning

Overlay models are usually composed of linear and polynomial-term source parameters. The linear parameters account for translation, rotation, and magnification while the polynomial-term parameters account for trapezoids, distortion, magnification, and bows. Here, discuss the physical meanings of the overlay models with parameters of third-order or higher will not be discussed, as in reality, they can be generalized by the increased error correction ability (the APC module) of exposure systems [2].

The overlay error is measured based on the misalignment of control points attached to the previous layer and the presentation layer of a certain wafer sample [2]. Inside the wafer, overlay control points are attached to the die centers and various locations inside each die. Then, the die coordinate, (X, Y), is defined as the coordinate of the die center in the reference of wafer center. The mark coordinate (x, y) is indicated as the coordinate of the control point in the reference of the die center. Therefore, the control point coordinate from the wafer center can be represented as $(x + X, y + Y)$ as it is shown in Figure 3.

Systematic overlay errors are classified into inter-die errors and inner-die errors. Mentioned above, the (X, Y) can be regarded as the inter-die coordinate while the (x, y) can be regarded as the inner-die coordinate. Therefore, the multiplication of X and Y (or x and y) also represents the physical meaning of the scale inter-die overlay errors. Table 1 lists the source parameters that can explain the two error types – each parameter

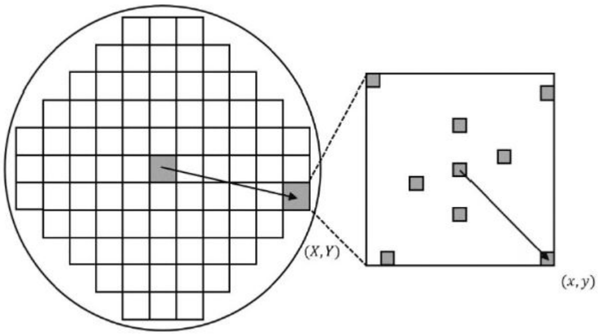


Figure 3: Die and control point coordinates in the reference of wafer.

is associated with a geometric meaning and is expressed by the means of die or control point coordinates [2].

Table 1: Source parameter descriptions

Notation inter-die (intra-die)	Overlay error along the x axis	Notation inter-die (intra-die)	Overlay error along the y axis
constant	Translation	constant	Translation
$X(x)$	Magnification	$Y(y)$	Magnification
$Y(y)$	Rotation	$X(x)$	Rotation
$X^2(x^2)$	2^{nd} magnification	$Y^2(y^2)$	2^{nd} magnification
$XY(xy)$	Trapezoid	$XY(xy)$	Trapezoid
$Y^2(y^2)$	Bow	$X^2(x^2)$	Bow
$X^3(x^3)$	3^{rd} magnification	$Y^3(y^3)$	3^{rd} magnification
$X^2Y(x^2y)$	Accordion	$Y^2X(y^2x)$	Accordion
$XY^2(xy^2)$	C-shape distortion	$YX^2(yx^2)$	C-shape distortion
$Y^3(y^3)$	3^{rd}-order flow	$X^3(x^3)$	3^{rd}-order flow

2.2 Analysis on Data

Another approach is to examine the data directly. Assuming all the data measured follows a normal distribution, Pearson correlation coefficients are calculated with the equation:

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{1}$$

The Pearson correlation coefficients always fall in the region of $[-1, 1]$. The further the coefficients are to 0, the more related the two variables are. When the coefficients are negative, the relation between the two variables is negative. i.e. if one variable increasing, the other variable will probably be decreasing. Positive coefficients mean positive relations between variables.

In this case, it will be unnecessary to judge whether it is positively related or negatively related. The concern is whether the variables has significant relation. Therefore, it will be more comprehensive to use absolute Pearson correlation coefficients: $|\rho|$.

As shown in the last part, the variables up to dimension 3 are interested in the model fitting due to their physical meanings. Now the calculation and its visualization of all terms within the dimension of 3 will be presented in the form of a heat map:

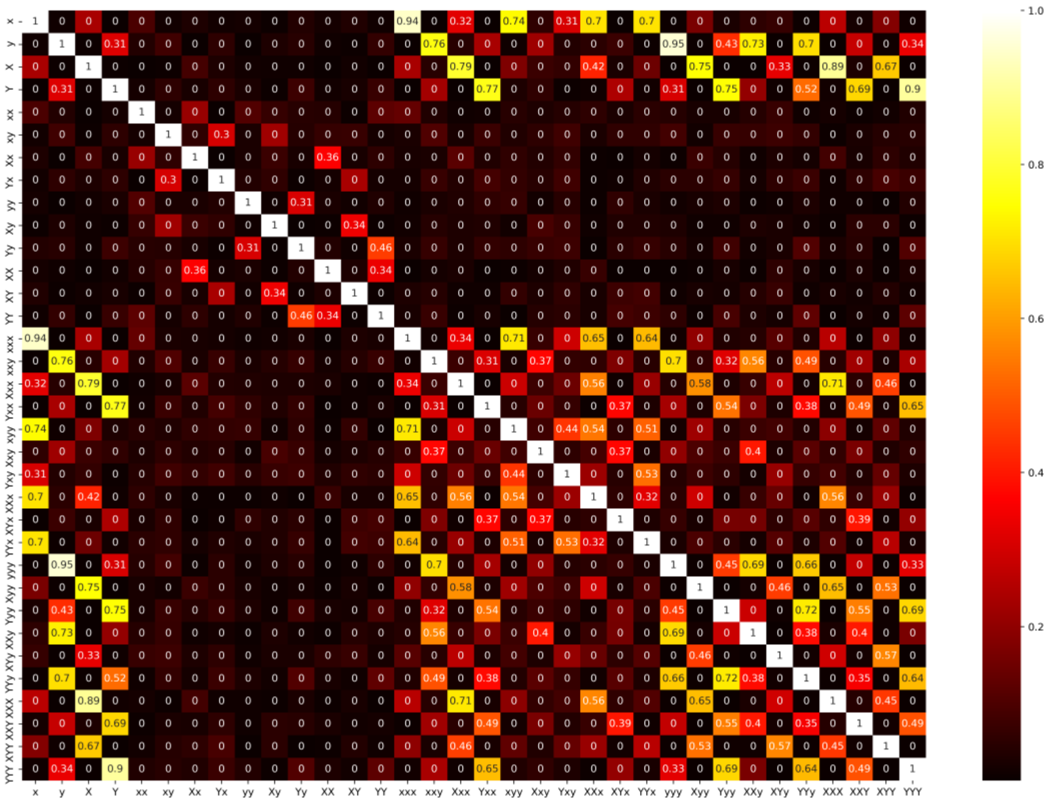


Figure 4: Absolute correlation for dimension 3

Each block in the figure represents the value of $|\rho|$. The brighter the color is, the more related the two variables are.

From the heat map, it can be observed that some terms correlate up to 0.9 and many other terms between 0.3 to 1.0. Thus these variables are related to each other at a certain degree.

According to the guideline provided by Laerd Statistics [4], the absolute correlation coefficients between 0.3 and 0.5 is considered of medium correlation and between 0.5 to 1.0 of high correlation. Therefore, it is necessary to consider only keeping one of those highly related paired variables in the model fitting part.

3 Model Fitting

3.1 The Default Model

3.1.1 Find Coefficients of the Multiple Linear Regression

The general model for Multiple Linear Regression:

$$\boldsymbol{Y} = X\beta + \boldsymbol{E} \tag{2}$$

From the slide, it is known that using Least-Squares Estimation, the coefficients of the multiple linear regression is

$$\boldsymbol{b} = \left(X^T X\right)^{-1} X^T \boldsymbol{Y} \tag{3}$$

Note that $\widehat{\beta} = \boldsymbol{b} := \begin{pmatrix} b_0 \\ \vdots \\ b_p \end{pmatrix}$

In this project, the default model is given by

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_1 X + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_8 xy + \alpha_9 x^2 + \alpha_{10} y^2 \tag{4}$$

$$o_y(X,Y,x,y) = \beta_0 + \beta_1 X + \beta_2 Y + \beta_3 XY + \beta_4 X^2 + \beta_5 Y^2 + \beta_6 x + \beta_7 y + \beta_8 xy + \beta_9 x^2 + \beta_{10} y^2 \tag{5}$$

In this case, the X matrix in Equation is defined as follows:

$$\begin{pmatrix} 1 & X_1 & Y_1 & X_1Y_1 & X_1^2 & Y_1^2 & x_1 & y_1 & x_1y_1 & x_1^2 & y_1^2 \\ 1 & X_2 & Y_2 & X_2Y_2 & X_2^2 & Y_2^2 & x_2 & y_2 & x_2y_2 & x_2^2 & y_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_{250} & Y_{250} & X_{250}Y_{250} & X_{250}^2 & Y_{250}^2 & x_{250} & y_{250} & x_{250}y_{250} & x_{250}^2 & y_{250}^2 \end{pmatrix}$$

The Y matrix in Equation is the horizontal combination of o_x and o_y data matrix.

Using MATLAB, all the coefficients of the multiple linear regression can be easily obtained:

α_0	-0.262423	β_0	0.706481
α_1	0.000652	β_1	0.002644
α_2	0.001277	β_2	0.003584
α_3	-0.000056	β_3	0.000003
α_4	-0.000046	β_4	0.000006
α_5	-0.000013	β_5	-0.000009
α_6	0.066598	β_6	0.016311
α_7	-0.034596	β_7	0.013139
α_8	-0.000353	β_8	0.000788
α_9	0.002939	β_9	-0.003350
α_{10}	0.001052	β_{10}	0.002651

3.1.2 Find Coefficients of Determination

Define $P := \frac{1}{n}\begin{pmatrix} 1 & 1 & \dots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}$ and $H := X\left(X^T X\right)^{-1} X^T$

The Error Sum of Squares:

$$SS_E = \langle \boldsymbol{Y}, (1_n - H)\,\boldsymbol{Y} \rangle \tag{6}$$

The Regression Sum of Squares:

$$SS_R = \langle \boldsymbol{Y}, (H - P)\boldsymbol{Y} \rangle \tag{7}$$

The Total Sum of Squares:

$$\mathrm{SS_T} = \mathrm{SS_R} + \mathrm{SS_E} = \langle \boldsymbol{Y}, (1_n - P)\,\boldsymbol{Y} \rangle = \underbrace{\langle \boldsymbol{Y}, (1_n - H)\,\boldsymbol{Y} \rangle}_{=SS_E} + \underbrace{\langle \boldsymbol{Y}, (H - P)\boldsymbol{Y} \rangle}_{=SS_R} \tag{8}$$

The coefficient of determination is then given by

$$R^2 = \frac{\mathrm{SS_T} - \mathrm{SS_E}}{\mathrm{SS_T}} = \frac{\mathrm{SS_R}}{\mathrm{SS_T}} \tag{9}$$

Using MATLAB, R^2 can be obtained as:

1. R^2 for the model of $o_x(X, Y, x, y)$ is 0.5786
2. R^2 for the model of $o_y(X, Y, x, y)$ is 0.3881

3.1.3 Interval Estimation for Coefficients

The unbiased estimator for σ^2 is given by

$$S^2 := \frac{\mathrm{SS_E}}{n - p - 1} \tag{10}$$

In the default model for this project, $n = 250$ and $p = 11$.
Using MATLAB, S^2 value can be obtained:

1. S^2 for the model of $o_x(X, Y, x, y)$ is 0.41564
2. S^2 for the model of $o_y(X, Y, x, y)$ is 0.38881

Write the matrix $\left(X^T X\right)^{-1}$ as

$$\left(X^T X\right)^{-1} = \begin{pmatrix} \xi_{00} & * & \cdots & \cdots & * \\ * & \xi_{11} & \ddots & & * \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & * \\ * & \cdots & \cdots & * & \xi_{pp} \end{pmatrix} \tag{11}$$

..., then it can obtained: $\mathrm{Var}\,[B_i]$:

$$\mathrm{Var}\,[B_i] = \xi_{ii}\sigma^2, \quad i = 0, \ldots, p \tag{12}$$

The $100(1-\alpha)\%$ confidence intervals for the model parameters is given by

$$\beta_j = b_j \pm t_{\alpha/2,n-p-1} S\sqrt{\xi_{jj}} \tag{13}$$

Set $\alpha = 5\%$. Since $n = 250$ and $p = 11$, $t_{\alpha/2,n-p-1} = 1.97$.

Using MATLAB, $100(1-\alpha)\%$ confidence intervals for all model parameters can be obtained.

Sample calculation:

$$\alpha_0 = -0.262423 \pm 1.97 \times \sqrt{0.41564} \times \sqrt{0.0352} = -0.262423 \pm 0.238210$$

α_0	-0.262423 ± 0.238210	β_0	0.706481 ± 0.230394
α_1	0.000652 ± 0.001090	β_1	0.002644 ± 0.001055
α_2	0.001277 ± 0.001112	β_2	0.003584 ± 0.001076
α_3	-0.000056 ± 0.000017	β_3	0.000003 ± 0.000016
α_4	-0.000046 ± 0.000017	β_4	0.000006 ± 0.000015
α_5	-0.000013 ± 0.000016	β_5	-0.000009 ± 0.000015
α_6	0.066598 ± 0.012719	β_6	0.016311 ± 0.012302
α_7	-0.034596 ± 0.007377	β_7	0.013139 ± 0.007135
α_8	-0.000353 ± 0.001114	β_8	0.000788 ± 0.001077
α_9	0.002939 ± 0.002497	β_9	-0.003350 ± 0.002415
α_{10}	0.001052 ± 0.000708	β_{10}	0.002651 ± 0.000685

3.2 T-Test for Model Sufficiency

Suppose that a regression model using the parameters $\beta_0, ..., \beta_p$ is fitted to Y

$$H_0 : \beta_j = 0$$

Define

$$T_{n-p-1} = \frac{b_j}{S\sqrt{\xi_{jj}}} \tag{14}$$

Then H_0 is rejected at significance level α if $|T_{n-p-1}| > t_{\alpha/2,n-p-1}$

In this project, n = 250, p = 11, set $\alpha = 5\%$, $t_{\alpha/2,n-p-1} = 1.97$.

3.2.1 T-Test for α_i

For the $o_x(X, Y, x, y)$ model, $S^2 = 0.5786$. It follows that $S = 0.6447$. Then the test statistic for each parameter α_i can be calculated via MATLAB:

	α_0	α_1	α_2	α_3	α_4	α_5
T_{n-p-1}	-2.1702	1.1782	2.2611	-6.5561	-5.8607	-1.6971
	α_6	α_7	α_8	α_9	α_{10}	
T_{n-p-1}	10.3152	-9.2391	-0.6241	2.3189	2.9256	

Sample calculation: suppose the null hypothesis $H_0 : \alpha_0 = 0$, then

$$T_{n-p-1} = T_{238} = \frac{\alpha_0}{S\sqrt{\xi_{00}}} = \frac{-0.262423}{0.6447 \times \sqrt{0.0352}} = -2.1702$$

3.2.2 T-Test for β_i

For the $o_y(X, Y, x, y)$ model, $S^2 = 0.38881$. It follows that $S = 0.6235$. Similarly, the test statistic for each parameter β_i can be calculated via MATLAB:

	β_0	β_1	β_2	β_3	β_4	β_5
T_{n-p-1}	6.0408	4.9381	6.5631	0.3678	0.7235	-1.2695
	β_6	β_7	β_8	β_9	β_{10}	
T_{n-p-1}	2.6120	3.6280	1.4401	-2.7329	7.6253	

Sample calculation: suppose the null hypothesis $H_0 : \beta_0 = 0$, then

$$T_{n-p-1} = T_{238} = \frac{\beta_0}{S\sqrt{\xi_{00}}} = \frac{0.706481}{0.6235 \times \sqrt{0.0352}} = 6.0408$$

3.3 The Simplified Model

The default model is a eleven-term four-variable model. The values of all parameters are given in the above table.

Here, the simplified model will be further examined by implementing the backward elimination procedure. In general, possibility will be judged; whether variable or terms (x, y, X, Y) can be deleted by implementing the backward elimination procedure.

3.3.1 Find the Simplified Model for $o_x(X, Y, x, y)$

The default model $o_x(X, Y, x, y) = \alpha_0 + \alpha_1 X + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_8 xy + \alpha_9 x^2 + \alpha_{10} y^2$ has been fitted. It has eleven terms in total.

If one term in the default model get deleted in order (delete term $X, Y, XY, X^2, Y^2, x, y, xy, x^2, y^2$ in order), then the ten-term models are:

$$o_x(X, Y, x, y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_8 xy + \alpha_9 x^2 + \alpha_{10} y^2 \quad (1.1)$$

$$o_x(X, Y, x, y) = \alpha_0 + \alpha_1 X + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_8 xy + \alpha_9 x^2 + \alpha_{10} y^2 \quad (1.2)$$

....

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_1 X + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_8 xy + \alpha_9 x^2 \quad (1.10)$$

Here, MATLAB code get manipulated and values of R^2 for each model given are obtained.

model	1.1	1.2	1.3	1.4	1.5
R^2	0.5775	0.5728	0.5010	0.5160	0.5740
model	1.6	1.7	1.8	1.9	1.10
R^2	0.3938	0.4179	0.5777	0.5676	0.5683

The model (1.8) has the largest R^2. Thus test $H_0 : \alpha_8 = 0$ now executed; comparing with the default model.

From the section of T-test, for α_8, $|T_{n-p-1}| = 0.6241 < 1.97 = t_{\alpha/2,n-p-1}$. Thus it is unable to reject H_0; term xy should be deleted from the model. Now the new model has form of:

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_1 X + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2$$

Then one term in this new model get deleted in order (delete term $X, Y, XY, X^2, Y^2, x, y, x^2, y^2$ in order), and the nine-term models have form:

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2 \quad (2.1)$$

....

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_1 X + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 \quad (2.9)$$

The values of R^2 for nine-term models are given below

model	2.1	2.2	2.3	2.4	2.5
R^2	0.5766	0.5718	0.4985	0.5158	0.5730
model	2.6	2.7	2.8	2.9	
R^2	0.3933	0.4146	0.5667	0.5676	

The model (2.1) has the largest R^2. Test $H_0 : \alpha_1 = 0$ should be executed. From the section of T-test, for α_1, $|T_{n-p-1}| = 1.1782 < 1.97 = t_{\alpha/2,n-p-1}$. Thus it is unable to reject H_0; term X should be from the model.

Now the new model becomes

$$o_x(X,Y,x,y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_5 Y^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2$$

Then the term-deleting process get repeated in this new model in order (delete term $Y, XY, X^2, Y^2, x, y, x^2, y^2$ in order), and encode these eight-term models as 3.1 - 3.8. The values of R^2 for eight-term models is given below

model	3.1	3.2	3.3	3.4
R^2	0.5706	0.4975	0.5148	0.5719
model	3.5	3.6	3.7	3.8
R^2	0.3888	0.4134	0.5654	0.5663

The model (3.4) has the largest R^2. Test $H_0 : \alpha_5 = 0$ should be executed. From the section of T-test, for α_5, $|T_{n-p-1}| = 1.6971 < 1.97 = t_{\alpha/2,n-p-1}$. It follows that it is unable to reject H_0. Thus Y^2 should be deleted from the model.

Now the new model becomes

$$o_x(X, Y, x, y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2$$

Since the absolute values of T_{n-p-1} for α_i remained in the new model are all larger than $t_{\alpha/2,n-p-1}$, all terms in the new model can not be deleted. Thus it is appropriate to end the backward elimination procedure. The final version of the simplified model for o_x is:

$$o_x(X, Y, x, y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2 \tag{15}$$

Through applying the backward elimination procedure step by step, an interesting preliminary conclusion is found: assume the term k with parameter α has the smallest absolute values of T_{n-p-1}. Then if the term k is deleted from the model, it is very likely to lead to the largest value of R^2. This implies that the complicated backward elimination procedure can be simplified: only need to delete all terms with absolute values of T_{n-p-1} smaller than the critical value $t_{\alpha/2,n-p-1}$.

3.3.2 Find the Simplified Model for $o_y(X, Y, x, y)$

Applying the conclusion in previous subsection, set $\beta_3, \beta_4, \beta_5, \beta_8$ to zero. The simplified model is:

$$o_y(X, Y, x, y) = \beta_0 + \beta_1 X + \beta_2 Y + \beta_6 x + \beta_7 y + \beta_9 x^2 + \beta_{10} y^2 \tag{16}$$

The result is checked by backward elimination procedure. Since the process is unnecessarily long and similar to that in simplifying o_x, the procedure in detail is omitted.

3.3.3 Linear Regression for the Simplified Models

1. For the simplified model

$$o_x(X, Y, x, y) = \alpha_0 + \alpha_2 Y + \alpha_3 XY + \alpha_4 X^2 + \alpha_6 x + \alpha_7 y + \alpha_9 x^2 + \alpha_{10} y^2,$$

the $100(1-\alpha)\%$ confidence intervals for the parameters is given below. ($\alpha = 5\%$)

α_0	α_2	α_3	α_4
-0.385510 ± 0.193787	0.001231 ± 0.001121	-0.000056 ± 0.000017	-0.000041 ± 0.000015
α_6	α_7	α_9	α_{10}
0.064588 ± 0.012468	-0.034517 ± 0.007427	0.003282 ± 0.002492	0.001092 ± 0.000713

The value of R^2 for the simplified model is 0.5719. (R^2 for default model is 0.5786)

2. For the simplified model

$$o_y(X, Y, x, y) = \beta_0 + \beta_1 X + \beta_2 Y + \beta_6 x + \beta_7 y + \beta_9 x^2 + \beta_{10} y^2,$$

the $100(1-\alpha)\%$ confidence intervals for the parameters is given below. ($\alpha = 5\%$)

β_0	β_1	β_2	β_6
0.670236 ± 0.165076	0.002652 ± 0.001066	0.003520 ± 0.001085	0.016434 ± 0.012423
β_7	β_9	β_{10}	
0.013708 ± 0.007190	-0.003155 ± 0.002411	0.002680 ± 0.000690	

The value of R^2 for the simplified model is 0.3728.(R^2 for default model is 0.3881)

The values of R^2 for both simplified models slightly decrease compared to the two default models. It implies that the simplification of the default models is quite successful.

3.4 Extensive Models

Mentioned in the 'Physical Meaning' subsection earlier, variables up to three-dimension has an actual physical meaning, which is needed to be taken into consideration. Now, the variables with higher dimensions will be discussed.

3.4.1 Full Model of Higher Dimensions

First of all, variables of higher dimensions are generated.

Take dimension 2 for example, formula obtained:

$$\begin{aligned} o_x(x, y, X, Y) =& \alpha_0 + \alpha_1 x + \alpha_2 y, \alpha_3 X + \alpha_4 Y + \alpha_5 xx + \alpha_6 xy + \alpha_7 Xx + \alpha_8 Yx + \alpha_9 yy + \\ & \alpha_{10} Xy + \alpha_{11} Yy + \alpha_{12} XX + \alpha_{13} XY + \alpha_{14} YY \end{aligned}$$

$$\begin{aligned} o_x(x, y, X, Y) =& \beta_0 + \beta_1 x + \beta_2 y, \beta_3 X + \beta_4 Y + \beta_5 xx + \beta_6 xy + \beta_7 Xx + \beta_8 Yx + \beta_9 yy + \\ & \beta_{10} Xy + \beta_{11} Yy + \beta_{12} XX + \beta_{13} XY + \beta_{14} YY \end{aligned}$$

As including higher dimension variables may probably lead to overfitting error, PRESS method is introduced to help preventing this problem.

Here, the equation:

$$PRESS = \sum_{i=1}^{n} (y_i - \hat{y_i})^2$$

is used where $\hat{y}_i$ is the predicted value of x_i using model fitted without i^{th} element.

The value of PRESS indicates how much the data may overfit. The smaller the PRESS is, the less possible the model is overfitting.

Then next is dimension 7, calculations for both PRESS and R^2:

$1,$
$Highest\ dim : Y, Size : 4$
$Full\ model : ox\ R^2 = 0.414, oy\ R^2 = 0.192$
$ox\ PRESS = 0.069, oy\ PRESS = 0.055$
$2,$
$Highest\ dim : YY, Size : 14$
$Full\ model : ox\ R^2 = 0.588, oy\ R^2 = 0.404$
$ox\ PRESS = 0.508, oy\ PRESS = 0.471$
$3,$
$Highest\ dim : YYY, Size : 34$
$Full\ model : ox\ R^2 = 0.660, oy\ R^2 = 0.635$
$ox\ PRESS = 3.007, oy\ PRESS = 2.414$
$4,$
$Highest\ dim : YYYY, Size : 69$
$Full\ model : ox\ R^2 = 0.739, oy\ R^2 = 0.760$
$ox\ PRESS = 11.232, oy\ PRESS = 7.255$
$5,$
$Highest\ dim : YYYYY, Size : 125$
$Full\ model : ox\ R^2 = 0.798, oy\ R^2 = 0.778$
$ox\ PRESS = 57.337, oy\ PRESS = 39.011$
$6,$
$Highest\ dim : YYYYYY, Size : 209$
$Full\ model : ox\ R^2 = 0.839, oy\ R^2 = 0.793$
$ox\ PRESS = 302.958, oy\ PRESS = 193.438$
$7,$
$Highest\ dim : YYYYYYY, Size : 329$
$Full\ model : ox\ R^2 = 0.889, oy\ R^2 = 0.794$
$ox\ PRESS = 1454.042, oy\ PRESS = 1032.572$

Here, ox indicates overlay error x and oy indicates overlay error y.

From the data obtained, it can be easily observed that as the dim increases, R^2 values of both ox and oy increases, meaning the fitted model are more able to feature the trend of the data. However, the PRESS for both ox and oy increases, indicating the model is getting overfitted. Even though the model can predict seen data well, it won't work if it cannot predict accurately unseen data. So, it will be necessary to find a balance between $PRESS$ and R^2.

3.5 Possible Optimal Model

In this case, consider the physical meaning of variables in the first place, so it would be preferred to start from dimension 3 and apply the backward method to find a better model. Also, the full extensive model of dimension 3 has $ox\ R^2 = 0.660,\ oy\ R^2 = 0.635$ and $ox\ PRESS = 3.007,\ oy\ PRESS = 2.414$, can be said balanced when comparing to dimension 2, which has R^2 too small, and to dimension 4, which has $PRESS$ too large.

Similar to the procedure used in 3.3, the full model is firstly fitted and the result of the parameter is found. Remove one parameter and fit the reduced model. Repeat this step until the test hypothesis:

$$H_0 : \alpha_i = 0$$

$$H_0 : \beta_i = 0$$

... cannot be rejected anymore.

3.5.1 Model for Overlay Error x

The Python program is executed to calculate and to test backward methods and desired values are obtained as:

$Highest\ dim : YYY, Size : 34$
$ox :\ Delete : XXx, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66044$
$-----$
$test\ statistic : 0.00877$
$T\ value : 1.97106$
$-----$
$ox :\ Delete : XYy, ox\ R^2 = 0.660$
......
$-----$
$ox :\ Delete : YYx, ox\ R^2 = 0.631$
$cur\ R^2 : 0.63472$
$-----$
$test\ statistic : 1.60625$
$T\ value : 1.97002$
$-----$
$ox :\ Delete : Xy, ox\ R^2 = 0.624$
$cur\ R^2 : 0.63075$
$-----$
$test\ statistic : -2.11774$
$T\ value : 1.96998$
$-----$
(See appendix for detail)

After the deletion of Xy, H_0 is rejected. Thus term Xy should be kept and remaining variables are:

$$\{0, y, xx, yy, Xy, XX, XY, xxx, Yxx, XXY, XYY, YYY\}$$

The model is then:

$$\begin{aligned}\hat{ox} = & -3.73 * 10^{-1} \ - \ 3.47 * 10^{-2} * y \ + \ 2.66 * 10^{-3} * xx + \\ & 1.16 * 10^{-3} * yy \ - \ 1.00 * 10^{-4} * Xy \ - \ 4.16 * 10^{-5} * XX - \\ & 6.32 * 10^{-5} * XY \ + \ 9.68 * 10^{-4*} xxx \ + \ 3.62 * 10^{-5} * Yxx + \\ & 3.40 * 10^{-7} * XXY \ + \ 2.17 * 10^{-7} * XYY \ - \ 1.39 * 10^{-7} * YYY\end{aligned}$$

The model has $R^2 = 0.631$, $PRESS = 0.339$

3.5.2 Model for Overlay Error y

The Python program is run to calculate, test using backward methods and desired values are obtained:

$Highest\ dim : YYY, Size : 34$
$oy :\ Delete : Xx, oy\ R^2 = 0.635$
$cur\ R^2 : 0.63492$
$-----$
$test\ statistic : 0.00900$
$T\ value : 1.97106$
$-----$
$oy :\ Delete : Yxx, oy\ R^2 = 0.635$
......
$-----$
$oy :\ Delete : YYY, oy\ R^2 = 0.617$
$cur\ R^2 : 0.62108$
$-----$
$test\ statistic : -1.67098$
$T\ value : 1.97024$
$-----$
$oy :\ Delete : XYy, oy\ R^2 = 0.610$
$cur\ R^2 : 0.61652$
$-----$
$test\ statistic : -1.98644$
$T\ value : 1.97020$
$-----$
(See appendix for detail)

After the deletion of XXy, H_0 is rejected. Thus XXy term should be kept and remaining variables are:

$$\{0, x, xx, yy, Yy, YY, xxx, Xxx, xyy, XYx, YYx, Xyy, Yyy, XXy, XYy, YYy, XXY\}$$

The model is then:

$$\begin{aligned}\hat{oy} = &- 6.13 * 10^{-1} \; - \; 1.06 * 10^{-2} * x \; + \; 2.51 * 10^{-3} * xx \; + \\ &1.17 * 10^{-3} * yy \; + \; 2.05 * 10^{-5} * Yy \; + \; 6.75 * 10^{-7} * YY \; + \\ &1.05 * 10^{-3} * xxx \; + \; 1.36 * 10^{-5} * Xxx \; - \; 4.88 * 10^{-5} * xyy \; - \\ &3.61 * 10^{-6} * XYx \; + \; 2.29 * 10^{-6} * YYx \; - \; 2.78 * 10^{-6} * Xyy \; - \\ &1.07 * 10^{-6} * Yyy \; - \; 2.57 * 10^{-6} * XXy \; - \; 1.25 * 10^{-6} * XYy \; - \\ &2.05 * 10^{-6} * YYy \; + \; 1.41 * 10^{-7} * XXY\end{aligned}$$

The model has $R^2 = 0.617$, $PRESS = 0.600$

3.5.3 Comments on Two Models

The result from the above fitting is listed below: The values of R^2 for the six-term models is given below.

model	R^2	PRESS
Optimized ox	0.631	0.339
Optimized oy	0.617	0.600
Given model ox	0.578	0.280
Given model oy	0.389	0.233
Full model(dim 3) ox	0.660	3.007
Full model(dim 3) oy	0.635	2.414

It can be concluded that our optimized model sacrifices a little bit on PRESS but get a great improvement on R^2 value. Therefore, it can be said that the obtained model is optimized.

4 Discussion & Conclusion

From various attempt to discover the optimal model for given data, it is concluded that the equation below is the model which exhibit the best mathematical relation between the variables and the outcome:

$$\begin{aligned}\hat{ox} = &- 3.73 * 10^{-1} \; - \; 3.47 * 10^{-2} * y \; + \; 2.66 * 10^{-3} * xx \; + \\ &1.16 * 10^{-3} * yy \; - \; 1.00 * 10^{-4} * Xy \; - \; 4.16 * 10^{-5} * XX \; - \\ &6.32 * 10^{-5} * XY \; + \; 9.68 * 10^{-4*} xxx \; + \; 3.62 * 10^{-5} * Yxx \; + \\ &3.40 * 10^{-7} * XXY \; + \; 2.17 * 10^{-7} * XYY \; - \; 1.39 * 10^{-7} * YYY\end{aligned}$$

$$
\begin{aligned}
\hat{oy} = &-6.13 * 10^{-1} \ - \ 1.06 * 10^{-2} * x \ + \ 2.51 * 10^{-3} * xx \ + \\
&1.17 * 10^{-3} * yy \ + \ 2.05 * 10^{-5} * Yy \ + \ 6.75 * 10^{-7} * YY \ + \\
&1.05 * 10^{-3} * xxx \ + \ 1.36 * 10^{-5} * Xxx \ - \ 4.88 * 10^{-5} * xyy \ - \\
&3.61 * 10^{-6} * XYx \ + \ 2.29 * 10^{-6} * YYx \ - \ 2.78 * 10^{-6} * Xyy \ - \\
&1.07 * 10^{-6} * Yyy \ - \ 2.57 * 10^{-6} * XXy \ - \ 1.25 * 10^{-6} * XYy \ - \\
&2.05 * 10^{-6} * YYy \ + \ 1.41 * 10^{-7} * XXY
\end{aligned}
$$

They has the balanced R^2 and $PRESS$ scores to prevent overfitting and lackfitting.

model	R^2	PRESS
Optimized ox	0.631	0.339
Optimized oy	0.617	0.600

Previously mentioned, the highest dimension is limited to 3. The parameters with higher parameters have several physical meaning. However, the R^2 value of the optimized model did not reach a satisfying value: significantly close to 1. This phenomenon might have the characteristic of photolithography technology as the reason. The photolithography technology is the mainstream in the semiconductor manufacturing industry where the process faces extremely short lengths as micrometers or even nanometers. At this length scale, the random variations contributing to the overlay errors are magnified and unpredictable. As errors are large and random, it is a challenge to fit the given data perfectly. Thus, the R^2 value obtained from the optimized model in this report can be considered as an acceptable result.

References

[1] A. U. Haq and D. Djurdjanovic. Robust control of overlay errors in photolithography processes. *IEEE Transactions on Semiconductor Manufacturing*, 32(3):320–333, 2019.

[2] S. J. Kim, H. G. Yoon, K. B. Lee, C. O. Kim, and S. J. Kim. Hybrid overlay modeling for field-by-field error correction in the photolithography process. *IEEE Transactions on Semiconductor Manufacturing*, 33(1):53–61, 2020.

[3] Y. T. Lin, C. C. Hsu, and S. Tseng. A semiconductor photolithography overlay analysis system using image processing approach. In *Ninth IEEE International Symposium on Multimedia Workshops (ISMW 2007)*, pages 63–69, 2007.

[4] Laerd Statistics. Pearson product-moment correlation. https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php, 2020.

5 Appendix

5.1 Procedure for Backward Elimination of Overlay Error x

$Highest\ dim : YYY, Size : 34$
$ox :\ Delete : XXx, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66044$
$-----$
$test\ statistic : 0.00877$
$T\ value : 1.97106$
$-----$
$ox :\ Delete : XYy, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66044$
$-----$
$test\ statistic : -0.17557$
$T\ value : 1.97101$
$-----$
$ox :\ Delete : x, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66039$
$-----$
$test\ statistic : 0.24061$
$T\ value : 1.97096$
$-----$
$ox :\ Delete : Yxy, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66030$
$-----$
$test\ statistic : -0.26338$
$T\ value : 1.97091$

$-----$
$ox :\ Delete : XXX, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66019$
$-----$
$test\ statistic : 0.25027$
$T\ value : 1.97086$
$-----$
$ox :\ Delete : Yyy, ox\ R^2 = 0.660$
$cur\ R^2 : 0.66010$
$-----$
$test\ statistic : 0.28257$
$T\ value : 1.97081$
$-----$
$ox :\ Delete : Yx, ox\ R^2 = 0.660$
$cur\ R^2 : 0.65997$

$-----$
$test\ statistic: -0.31161$
$T\ value: 1.97076$
$-----$
$ox:\ Delete: YYy, ox\ R^2 = 0.660$
$cur\ R^2: 0.65982$
$-----$
$test\ statistic: 0.30461$
$T\ value: 1.97071$
$-----$
$ox:\ Delete: Xyy, ox\ R^2 = 0.659$
$cur\ R^2: 0.65968$
$-----$
$test\ statistic: -0.39078$
$T\ value: 1.97066$
$-----$
$ox:\ Delete: Yy, ox\ R^2 = 0.659$
$cur\ R^2: 0.65945$
$-----$
$test\ statistic: -0.58203$
$T\ value: 1.97061$
$-----$
$ox:\ Delete: xxy, ox\ R^2 = 0.658$
$cur\ R^2: 0.65893$
$-----$
$test\ statistic: 0.67013$
$T\ value: 1.97056$
$-----$
$ox:\ Delete: yyy, ox\ R^2 = 0.657$
$cur\ R^2: 0.65825$
$-----$
$test\ statistic: 0.72128$
$T\ value: 1.97052$
$-----$
$ox:\ Delete: xyy, ox\ R^2 = 0.657$
$cur\ R^2: 0.65747$
$-----$
$test\ statistic: -0.76782$
$T\ value: 1.97047$
$-----$
$ox:\ Delete: Y, ox\ R^2 = 0.655$
$cur\ R^2: 0.65658$
$-----$
$test\ statistic: 0.89048$

$T\ value : 1.97042$
$-----$
$ox:\ Delete : X, ox\ R^2 = 0.654$
$cur\ R^2 : 0.65538$
$-----$
$test\ statistic : -1.04714$
$T\ value : 1.97038$
$-----$
$ox:\ Delete : Xxx, ox\ R^2 = 0.653$
$cur\ R^2 : 0.65373$
$-----$
$test\ statistic : 0.55213$
$T\ value : 1.97033$
$-----$
$ox:\ Delete : XYx, ox\ R^2 = 0.651$
$cur\ R^2 : 0.65327$
$-----$
$test\ statistic : -1.13990$
$T\ value : 1.97029$
$-----$
$ox:\ Delete : Xx, ox\ R^2 = 0.649$
$cur\ R^2 : 0.65132$
$-----$
$test\ statistic : -1.20978$
$T\ value : 1.97024$
$-----$
$ox:\ Delete : xy, ox\ R^2 = 0.646$
$cur\ R^2 : 0.64912$
$-----$
$test\ statistic : -1.48909$
$T\ value : 1.97020$
$-----$
$ox:\ Delete : YY, ox\ R^2 = 0.642$
$cur\ R^2 : 0.64578$
$-----$
$test\ statistic : -1.56906$
$T\ value : 1.97015$
$-----$
$ox:\ Delete : Xxy, ox\ R^2 = 0.638$
$cur\ R^2 : 0.64206$
$-----$
$test\ statistic : 1.69502$
$T\ value : 1.97011$
$-----$

$ox: \ Delete: XXy, ox \ R^2 = 0.635$
$cur \ R^2: 0.63768$
$-----$
$test \ statistic: 1.38760$
$T \ value: 1.97007$
$-----$
$ox: \ Delete: YYx, ox \ R^2 = 0.631$
$cur \ R^2: 0.63472$
$-----$
$test \ statistic: 1.60625$
$T \ value: 1.97002$
$-----$
$ox: \ Delete: Xy, ox \ R^2 = 0.624$
$cur \ R^2: 0.63075$
$-----$
$test \ statistic: -2.11774$
$T \ value: 1.96998$
$-----$

5.2 Procedure for Backward Elimination of Overlay Error y

$Highest \ dim: YYY, Size: 34$
$oy: \ Delete: Xx, oy \ R^2 = 0.635$
$cur \ R^2: 0.63492$
$-----$
$test \ statistic: 0.00900$
$T \ value: 1.97106$
$-----$
$oy: \ Delete: Yxx, oy \ R^2 = 0.635$
$cur \ R^2: 0.63492$
$-----$
$test \ statistic: -0.02050$
$T \ value: 1.97101$
$-----$
$oy: \ Delete: XX, oy \ R^2 = 0.635$
$cur \ R^2: 0.63492$
$-----$
$test \ statistic: 0.14228$
$T \ value: 1.97096$
$-----$
$oy: \ Delete: xxy, oy \ R^2 = 0.635$
$cur \ R^2: 0.63489$
$-----$

$test\ statistic : -0.14966$
$T\ value : 1.97091$
$-----$
$oy :\ Delete : XY, oy\ R^2 = 0.635$
$cur\ R^2 : 0.63485$
$-----$
$test\ statistic : 0.22386$
$T\ value : 1.97086$
$-----$
$oy :\ Delete : XYY, oy\ R^2 = 0.635$
$cur\ R^2 : 0.63477$
$-----$
$test\ statistic : -0.26001$
$T\ value : 1.97081$
$-----$
$oy :\ Delete : XXX, oy\ R^2 = 0.635$
$cur\ R^2 : 0.63465$
$-----$
$test\ statistic : -0.28535$
$T\ value : 1.97076$
$-----$
$oy :\ Delete : Xxy, oy\ R^2 = 0.634$
$cur\ R^2 : 0.63452$
$-----$
$test\ statistic : -0.30637$
$T\ value : 1.97071$
$-----$
$oy :\ Delete : Xy, oy\ R^2 = 0.634$
$cur\ R^2 : 0.63437$
$-----$
$test\ statistic : 0.29914$
$T\ value : 1.97066$
$-----$
$oy :\ Delete : Yxy, oy\ R^2 = 0.634$
$cur\ R^2 : 0.63422$
$-----$
$test\ statistic : 0.61599$
$T\ value : 1.97061$
$-----$
$oy :\ Delete : XXx, oy\ R^2 = 0.632$
$cur\ R^2 : 0.63360$
$-----$
$test\ statistic : 0.83770$
$T\ value : 1.97056$

$- - - - -$
$oy:\ Delete: Yx, oy\ R^2 = 0.631$
$cur\ R^2: 0.63246$
$- - - - -$
$test\ statistic: 0.86999$
$T\ value: 1.97052$
$- - - - -$
$oy:\ Delete: yyy, oy\ R^2 = 0.630$
$cur\ R^2: 0.63123$
$- - - - -$
$test\ statistic: -0.99971$
$T\ value: 1.97047$
$- - - - -$
$oy:\ Delete: y, oy\ R^2 = 0.629$
$cur\ R^2: 0.62960$
$- - - - -$
$test\ statistic: 0.59588$
$T\ value: 1.97042$
$- - - - -$
$oy:\ Delete: X, oy\ R^2 = 0.627$
$cur\ R^2: 0.62903$
$- - - - -$
$test\ statistic: 1.12917$
$T\ value: 1.97038$
$- - - - -$
$oy:\ Delete: Y, oy\ R^2 = 0.624$
$cur\ R^2: 0.62696$
$- - - - -$
$test\ statistic: 1.30151$
$T\ value: 1.97033$
$- - - - -$
$oy:\ Delete: xy, oy\ R^2 = 0.621$
$cur\ R^2: 0.62421$
$- - - - -$
$test\ statistic: 1.38742$
$T\ value: 1.97029$
$- - - - -$
$oy:\ Delete: YYY, oy\ R^2 = 0.617$
$cur\ R^2: 0.62108$
$- - - - -$
$test\ statistic: -1.67098$
$T\ value: 1.97024$
$- - - - -$
$oy:\ Delete: XYy, oy\ R^2 = 0.610$

$cur\ R^2 : 0.61652$
$-----$
$test\ statistic : -1.98644$
$T\ value : 1.97020$
$-----$

5.3 Code for Default and Simplified Models

```
clc;
data = readmatrix('term_project_overlay_data.csv');
x_err = data(:,1);
y_err = data(:,2);
x = data(:,3);
X = data(:,4);
y = data(:,5);
Y = data(:,6);
XY = X.*Y;
XX = X.*X;
YY = Y.*Y;
xy = x.*y;
xx = x.*x;
yy = y.*y;

o = ones(length(x),1);
A = [o X Y XY XX YY x y xy xx yy];
B = [x_err y_err];
b = inv(A'*A)*A'*B;

H = A*inv(A'*A)*A';
I = eye(length(x));
P = ones(length(x))/length(x);
SSR = B'*((H-P)*B);
SSE = B'*((I-H)*B);
SST = B'*((I-P)*B);
R_square = SSR/SST;
S_square = SSE/(250-11-1);
S = sqrt(S_square);

INV = inv(A'*A);
var = zeros(2, 11);
for i = 1:11
    var(1,i) = 1.97*S(1,1)*sqrt(INV(i,i));
    var(2,i) = 1.97*S(2,2)*sqrt(INV(i,i));
end


test = zeros(2, 11);
for i = 1:11
    test(1,i) = b(i,1)/(S(1,1)*sqrt(INV(i,i)));
    test(2,i) = b(i,2)/(S(2,2)*sqrt(INV(i,i)));
end
```

5.4 Code for Heat Map and Extensive and Optimized Models

```
In [2]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        import scipy.stats as stats
```

```
In [7]: def fit(df, order, omitted_id=-1, r2=False, print_f=False):
            num_rows=np.shape(df)[0]
            if (omitted_id!=-1):
                x_train=df.drop(index=[omitted_id]).iloc[:, :-2].to_numpy()
                ox_train=df["overlay_error_x"].drop(index=[omitted_id]).to_
        numpy()
                oy_train=df["overlay_error_y"].drop(index=[omitted_id]).to_
        numpy()
                #print(df.drop(index=[omitted_id]).iloc[:, :-2])
            else:
                x_train=df.iloc[:, :-2].to_numpy()
                ox_train=df["overlay_error_x"].to_numpy()
                oy_train=df["overlay_error_y"].to_numpy()
                #print(df.iloc[:, :-2])
            H=np.linalg.pinv(x_train.transpose().dot(x_train)).dot(x_train.
        transpose())
            alpha_x=H.dot(ox_train)
            alpha_y=H.dot(oy_train)
            ox_hat=x_train.dot(alpha_x.transpose())
            oy_hat=x_train.dot(alpha_y.transpose())
            p=1/num_rows*np.ones((num_rows, num_rows))
            id_mat=np.identity(num_rows)
            SSe_ox=sum((ox_train-ox_hat)**2)
            SSr_ox=sum((ox_hat-np.mean(ox_train))**2)
            SSt_ox=SSe_ox+SSr_ox
            ox_R2=SSr_ox/SSt_ox
            SSe_oy=sum(np.abs((oy_train-oy_hat)**2))
            SSr_oy=sum(np.abs((oy_hat-np.mean(oy_train))**2))
            SSt_oy=SSe_oy+SSr_oy
            oy_R2=SSr_oy/SSt_oy
            #if (omitted_id!=-1): print("Omitted {}, ox_R2={:.3f}, oy_R2={:
        .3f}".format(omitted_id, ox_R2, oy_R2))
            if (omitted_id==-1 and print_f): print("Full model: ox R2={:.3f
        }, oy R2={:.3f}".format(ox_R2, oy_R2))
            if (r2): return [ox_hat, oy_hat, alpha_x, alpha_y, ox_R2, oy_R2
        ]
            else: return [ox_hat, oy_hat, alpha_x, alpha_y]

        def get_press(df, order, print_f=False):
            [ox_hat_full, oy_hat_full, t_alpha_x, _]=fit(df, order, print_f
        =print_f)
            ox_press=0
            oy_press=0
            for i in range(np.shape(df)[0]):
                x_train_i=df.iloc[i, :-2].to_numpy()
                [ox_hat, oy_hat, alpha_x, alpha_y]=fit(df, order, i, print_
        f=True)
                ox_press+=(ox_hat_full[i]-x_train_i.dot(alpha_x.transpose()
```

```
))**2
        oy_press+=(oy_hat_full[i]-x_train_i.dot(alpha_y.transpose()
))**2
    print("ox_press={:.3f}, oy_press={:.3f}".format(ox_press, oy_pr
ess))
    return

def draw_heatmap(df, name=""):
    plt.cla()
    corr=np.abs(df.corr())
    corr_t=corr.copy()
    corr_t[corr_t<0.3]=0
    plt.figure(figsize=(20,14))
    sns.heatmap(corr, cmap="hot", annot=corr_t)
    if (name!=""): plt.savefig(name, dpi=400)
    plt.show()


def get_order(df, dim):
    ori_order=["x", "y", "X", "Y"]
    order=["x", "y", "X", "Y"]
    for depth in range(1, dim):
        temp_order=[]
        for i in range(len(ori_order)):
            for j in range(len(order)):
                temp_item="".join(sorted(ori_order[i]+order[j]))
                if (len(temp_item)<depth or (temp_item in temp_orde
r) or (temp_item in order)): continue
                df[temp_item]=df[ori_order[i]]*df[order[j]]
                temp_order.append(temp_item)
        order+=temp_order
    return order

def get_df_of_dim(dim, with_response=True):
    ori_df=pd.read_csv("term_project_overlay_data.csv")
    ori_num_rows=np.shape(ori_df)[0]
    ori_df["0"]=pd.Series(np.ones(ori_num_rows))
    ori_num_cols=np.shape(ori_df)[1]
    df=ori_df
    order=get_order(df, dim)
    print("Highest dim:{}, Size: {}".format(order[-1], len(order)))
    if (with_response): order=["0"]+order+["overlay_error_x", "over
lay_error_y"]
    else: order=["0"]+order
    df=df[order]
    return (df, order)

def run_search(dim):
    ori_df=pd.read_csv("term_project_overlay_data.csv")
    ori_num_rows=np.shape(ori_df)[0]
    ori_df["0"]=pd.Series(np.ones(ori_num_rows))
    ori_num_cols=np.shape(ori_df)[1]
    for i in range(1, dim):
        df=ori_df
        order=get_order(df, i)
        print("Highest dim:{}, Size: {}".format(order[-1], len(orde
r)))
```

```
        order=["0"]+order+["overlay_error_x", "overlay_error_y"]
        df=df[order]
        get_press(df, order, print_f=True)

def param_conf(x_df, y_df, y_hat, alpha):
    n=x_df.shape[0]
    p=x_df.shape[1]-1
    out_list=[]
    t_val=stats.t.ppf(1-alpha/2, df=(n-p-1))
    var_est=var_estimator(y_df.to_numpy(), pd.DataFrame(y_hat).to_n
umpy(), n, p)
    for param_id in range(0, p+1):
        out_list.append(t_val*np.sqrt(var_est*kesai(x_df)[param_id]
[param_id]))
    return out_list

def var_estimator(y_train, y_hat, n, p):
    return sse(y_train, y_hat)/(n-p-1)

def sse(y_train, y_hat):
    return sum((y_train-y_hat)**2)

def ssr(y_train, y_hat):
    return sum((y_hat-np.mean(y_train))**2)

def sst(y_train, y_hat):
    #return sum((y_train-np.mean(y_train))**2)
    return sse(y_train, y_hat)+ssr(y_train, y_hat)

def kesai(x_df):
    x_train=x_df.to_numpy()
    H=np.linalg.inv(x_train.transpose().dot(x_train))
    return H

def smallest_r2(df, order, ox):
    [ox_hat_full, oy_hat_full, t_alpha_x, _]=fit(df, order)
    ox_press=0
    oy_press=0
    ox_df=pd.DataFrame(df.iloc[:,-2])
    oy_df=pd.DataFrame(df.iloc[:,-1])
    ox_R2_list=[]
    oy_R2_list=[]
    for i in range(1, len(order)-2):
        temp_order=[]
        [ox_hat, oy_hat, alpha_x, alpha_y, ox_R2, oy_R2]=fit(df.dro
p(df.columns[i],axis=1), temp_order, r2=True)
        #ox_R2=ssr(ox_df.to_numpy(), pd.DataFrame(ox_hat).to_numpy(
))/sst(ox_df.to_numpy(), pd.DataFrame(ox_hat).to_numpy())
        ox_R2_list.append(ox_R2)
        #oy_R2=ssr(oy_df.to_numpy(), pd.DataFrame(oy_hat).to_numpy(
))/sst(oy_df.to_numpy(), pd.DataFrame(oy_hat).to_numpy())
        oy_R2_list.append(oy_R2)
    if (ox):
        print("ox: Delete: {}, ox_R2={:.3f}".format(order[np.argmax
(ox_R2_list)+1], np.max(ox_R2_list)))
        return np.argmax(ox_R2_list)+1
    else:
```

```
        print("oy: Delete: {}, oy_R2={:.3f}".format(order[np.argmax(oy_R2_list)+1], np.max(oy_R2_list)))
        return np.argmax(oy_R2_list)+1

def param_test(x_df, y_df, y_train, param, i, alpha=0.05):
    n=x_df.shape[0]
    p=x_df.shape[1]-1
    t_val=stats.t.ppf(1-alpha/2, df=(n-p-1))
    #print(param[i])
    #print(kesai(x_df)[i][i])
    test_s=param[i]/np.sqrt(var_estimator(y_df.to_numpy(), pd.DataFrame(y_train).to_numpy(), n, p)*kesai(x_df)[i][i])
    print("-----")
    print("test statistic: {:.5f}".format(test_s[0]))
    print("T value: {:.5f}".format(t_val))
    print("-----")
    if abs(test_s[0])>t_val: return True
    else: return False

def backward(dim, ox, order=[]):
    if (ox): pos=-2
    else: pos=-1
    (df, order_t)=get_df_of_dim(dim)
    if (order==[]): order=order_t
    df=df[order]
    [_, _, _, _, ox_r2_prev, oy_r2_prev]=fit(df, order, r2=True)
    while (1):
        #[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, order)
        i_t=smallest_r2(df, order, ox)
        [ox_hat, oy_hat, alpha_x, alpha_y, ox_r2, oy_r2]=fit(df, order, r2=True)
        flag=False
        if (ox):
            hat=ox_hat
            param=alpha_x
            print("cur r2: {:.5f}".format(ox_r2))
            #if (ox_r2<ox_r2_prev): flag=True
            #else: flag=False
        else:
            hat=oy_hat
            param=alpha_y
            print("cur r2: {:.5f}".format(oy_r2))
            #if (oy_r2<oy_r2_prev): flag=True
            #else: flag=False
        #print(np.shape(param))
        if (param_test(df.iloc[:, :-2],pd.DataFrame(df.iloc[:, pos]), hat, param, i_t, 0.05)):
            break
        df.drop(df.columns[i_t],axis=1, inplace=True)
        order.pop(i_t)
    return df.columns
```

In [207]:
```python
#Calculate and test for dim3, overlay error y.
dim=3
(df, order)=get_df_of_dim(dim)
order_f=backward(dim, False, order)
print(order_f)
df=df[order_f]
[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, or
der_f, print_f=True)
get_press(df, order_f)
print(oy_alpha_full)
```

```
Highest dim:YYY, Size: 34
Highest dim:YYY, Size: 34
oy: Delete: Xx, oy_R2=0.635
cur r2: 0.63492
-----
test statistic: 0.00900
T value: 1.97106
-----
oy: Delete: Yxx, oy_R2=0.635
cur r2: 0.63492
-----
test statistic: -0.02050
T value: 1.97101
-----
oy: Delete: XX, oy_R2=0.635
cur r2: 0.63492
-----
test statistic: 0.14228
T value: 1.97096
-----
oy: Delete: xxy, oy_R2=0.635
cur r2: 0.63489
-----
test statistic: -0.14966
T value: 1.97091
-----
oy: Delete: XY, oy_R2=0.635
cur r2: 0.63485
-----
test statistic: 0.22386
T value: 1.97086
-----
oy: Delete: XYY, oy_R2=0.635
cur r2: 0.63477
-----
test statistic: -0.26001
T value: 1.97081
-----
oy: Delete: XXX, oy_R2=0.635
cur r2: 0.63465
-----
test statistic: -0.28535
T value: 1.97076
-----
oy: Delete: Xxy, oy_R2=0.634
```

```
cur r2: 0.63452
-----
test statistic: -0.30637
T value: 1.97071
-----
oy: Delete: Xy, oy_R2=0.634
cur r2: 0.63437
-----
test statistic: 0.29914
T value: 1.97066
-----
oy: Delete: Yxy, oy_R2=0.634
cur r2: 0.63422
-----
test statistic: 0.61599
T value: 1.97061
-----
oy: Delete: XXx, oy_R2=0.632
cur r2: 0.63360
-----
test statistic: 0.83770
T value: 1.97056
-----
oy: Delete: Yx, oy_R2=0.631
cur r2: 0.63246
-----
test statistic: 0.86999
T value: 1.97052
-----
oy: Delete: yyy, oy_R2=0.630
cur r2: 0.63123
-----
test statistic: -0.99971
T value: 1.97047
-----
oy: Delete: y, oy_R2=0.629
cur r2: 0.62960
-----
test statistic: 0.59588
T value: 1.97042
-----
oy: Delete: X, oy_R2=0.627
cur r2: 0.62903
-----
test statistic: 1.12917
T value: 1.97038
-----
oy: Delete: Y, oy_R2=0.624
cur r2: 0.62696
-----
test statistic: 1.30151
T value: 1.97033
-----
oy: Delete: xy, oy_R2=0.621
cur r2: 0.62421
-----
test statistic: 1.38742
```

```
T value: 1.97029
-----
oy: Delete: YYY, oy_R2=0.617
cur r2: 0.62108
-----
test statistic: -1.67098
T value: 1.97024
-----
oy: Delete: XYy, oy_R2=0.610
cur r2: 0.61652
-----
test statistic: -1.98644
T value: 1.97020
-----
Index(['0', 'x', 'xx', 'yy', 'Yy', 'YY', 'xxx', 'Xxx', 'xyy', 'XYx
', 'YYx',
       'Xyy', 'Yyy', 'XXy', 'XYy', 'YYy', 'XXY', 'overlay_error_x'
,
       'overlay_error_y'],
      dtype='object')
Full model: ox R2=0.434, oy R2=0.617
ox_press=1.334, oy_press=0.600
[ 7.88295618e-01  8.29923140e-02 -3.26887705e-03  2.31059502e-03
 -1.17924424e-04 -1.36673601e-05 -9.35851957e-04  2.72879483e-05
  1.05181838e-04  3.24892265e-06 -2.69799481e-06  8.36999407e-06
  1.51802010e-05  1.43663597e-06 -1.28239367e-06  2.53707527e-06
  8.57879515e-07]
```

In [206]:

```
#Calculate and test for dim3, overlay error x.
dim=3
(df, order)=get_df_of_dim(dim)
order_f=backward(dim, True, order)
print(order_f)
df=df[order_f]
[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, or
der_f, print_f=True)
get_press(df, order_f)
print(ox_alpha_full)
```

```
Highest dim:YYY, Size: 34
Highest dim:YYY, Size: 34
ox: Delete: XXx, ox_R2=0.660
cur r2: 0.66044
-----
test statistic: 0.00877
T value: 1.97106
-----
ox: Delete: XYy, ox_R2=0.660
cur r2: 0.66044
-----
test statistic: -0.17557
T value: 1.97101
-----
ox: Delete: x, ox_R2=0.660
cur r2: 0.66039
-----
test statistic: 0.24061
```

```
T value: 1.97096
-----
ox: Delete: Yxy, ox_R2=0.660
cur r2: 0.66030
-----
test statistic: -0.26338
T value: 1.97091
-----
ox: Delete: XXX, ox_R2=0.660
cur r2: 0.66019
-----
test statistic: 0.25027
T value: 1.97086
-----
ox: Delete: Yyy, ox_R2=0.660
cur r2: 0.66010
-----
test statistic: 0.28257
T value: 1.97081
-----
ox: Delete: Yx, ox_R2=0.660
cur r2: 0.65997
-----
test statistic: -0.31161
T value: 1.97076
-----
ox: Delete: YYy, ox_R2=0.660
cur r2: 0.65982
-----
test statistic: 0.30461
T value: 1.97071
-----
ox: Delete: Xyy, ox_R2=0.659
cur r2: 0.65968
-----
test statistic: -0.39078
T value: 1.97066
-----
ox: Delete: Yy, ox_R2=0.659
cur r2: 0.65945
-----
test statistic: -0.58203
T value: 1.97061
-----
ox: Delete: xxy, ox_R2=0.658
cur r2: 0.65893
-----
test statistic: 0.67013
T value: 1.97056
-----
ox: Delete: yyy, ox_R2=0.657
cur r2: 0.65825
-----
test statistic: 0.72128
T value: 1.97052
-----
ox: Delete: xyy, ox_R2=0.657
```

```
cur r2: 0.65747
-----
test statistic: -0.76782
T value: 1.97047
-----
ox: Delete: Y, ox_R2=0.655
cur r2: 0.65658
-----
test statistic: 0.89048
T value: 1.97042
-----
ox: Delete: X, ox_R2=0.654
cur r2: 0.65538
-----
test statistic: -1.04714
T value: 1.97038
-----
ox: Delete: Xxx, ox_R2=0.653
cur r2: 0.65373
-----
test statistic: 0.55213
T value: 1.97033
-----
ox: Delete: XYx, ox_R2=0.651
cur r2: 0.65327
-----
test statistic: -1.13990
T value: 1.97029
-----
ox: Delete: Xx, ox_R2=0.649
cur r2: 0.65132
-----
test statistic: -1.20978
T value: 1.97024
-----
ox: Delete: xy, ox_R2=0.646
cur r2: 0.64912
-----
test statistic: -1.48909
T value: 1.97020
-----
ox: Delete: YY, ox_R2=0.642
cur r2: 0.64578
-----
test statistic: -1.56906
T value: 1.97015
-----
ox: Delete: Xxy, ox_R2=0.638
cur r2: 0.64206
-----
test statistic: 1.69502
T value: 1.97011
-----
ox: Delete: XXy, ox_R2=0.635
cur r2: 0.63768
-----
test statistic: 1.38760
```

```
T value: 1.97007
-----
ox: Delete: YYx, ox_R2=0.631
cur r2: 0.63472
-----
test statistic: 1.60625
T value: 1.97002
-----
ox: Delete: Xy, ox_R2=0.624
cur r2: 0.63075
-----
test statistic: -2.11774
T value: 1.96998
-----
Index(['0', 'y', 'xx', 'yy', 'Xy', 'XX', 'XY', 'xxx', 'Yxx', 'XXY'
, 'XYY',
       'YYY', 'overlay_error_x', 'overlay_error_y'],
      dtype='object')
Full model: ox R2=0.631, oy R2=0.455
ox_press=0.339, oy_press=0.311
[-3.72716175e-01 -3.47164631e-02  2.65520999e-03  1.15593558e-03
 -1.00186880e-04 -4.16163474e-05 -6.31891951e-05  9.67866135e-04
  3.62387526e-05  3.39990434e-07  2.17051468e-07 -1.39170662e-07]
```

In [7]:

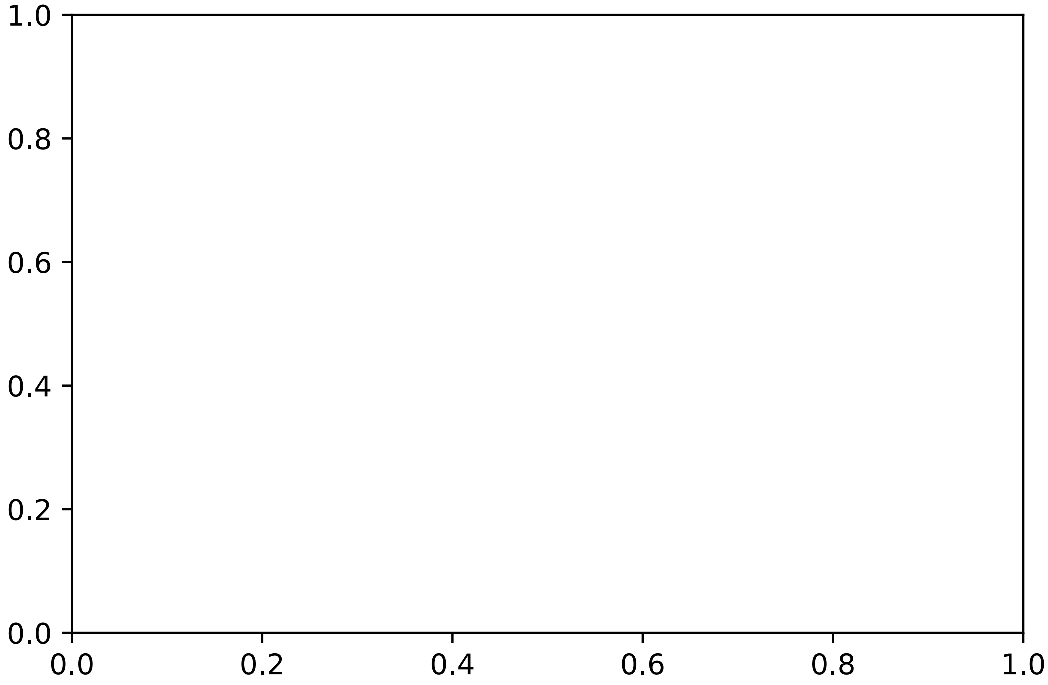
```
#Calculate PRESS and R2 up to dim 7
run_search(8)
```

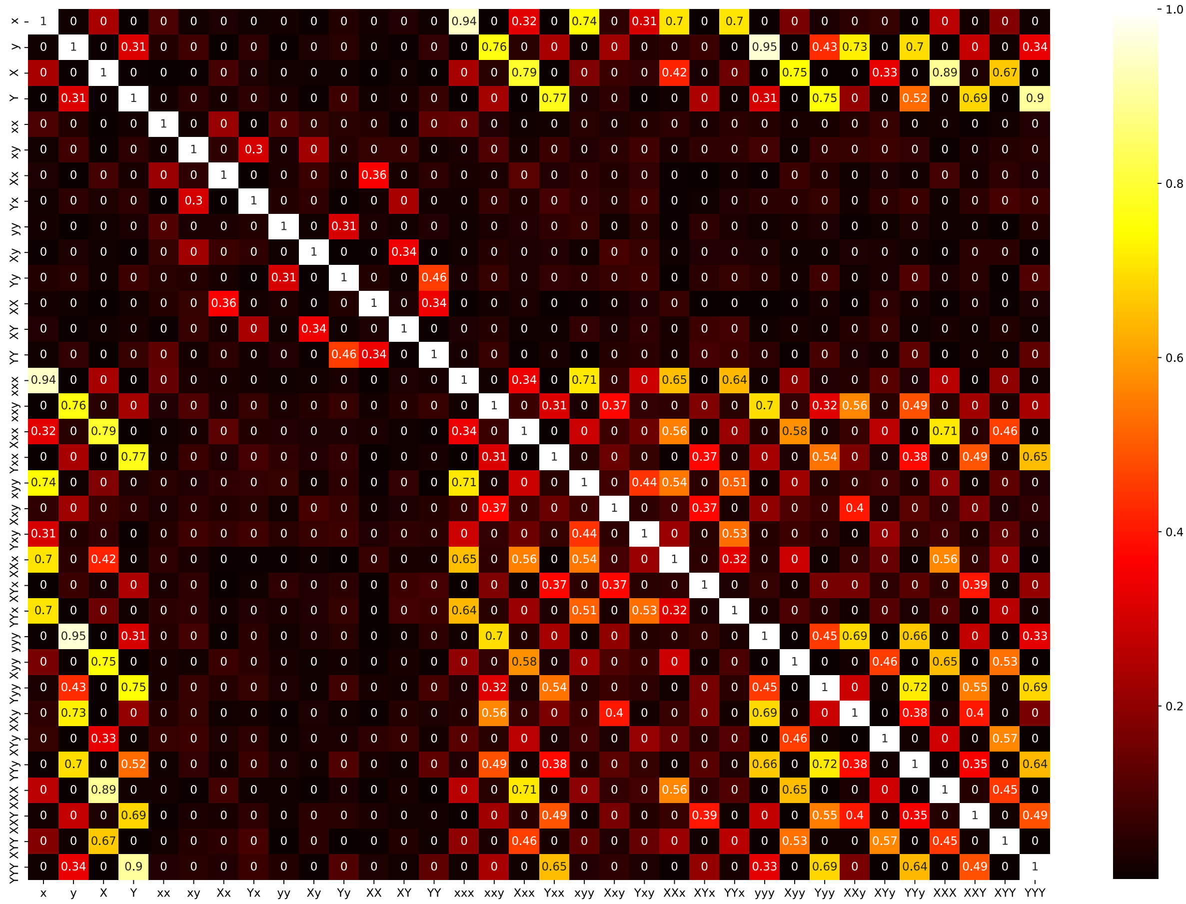
```
Highest dim:Y, Size: 4
Full model: ox R2=0.414, oy R2=0.192
ox_press=0.069, oy_press=0.055
Highest dim:YY, Size: 14
Full model: ox R2=0.588, oy R2=0.404
ox_press=0.508, oy_press=0.471
Highest dim:YYY, Size: 34
Full model: ox R2=0.660, oy R2=0.635
ox_press=3.007, oy_press=2.414
Highest dim:YYYY, Size: 69
Full model: ox R2=0.739, oy R2=0.760
ox_press=11.232, oy_press=7.255
Highest dim:YYYYY, Size: 125
Full model: ox R2=0.798, oy R2=0.778
ox_press=57.337, oy_press=39.011
Highest dim:YYYYYY, Size: 209
Full model: ox R2=0.839, oy R2=0.793
ox_press=302.958, oy_press=193.438
Highest dim:YYYYYYY, Size: 329
Full model: ox R2=0.889, oy R2=0.794
ox_press=1454.042, oy_press=1032.572
```

In [209]:

```python
(df_dim3, order)=get_df_of_dim(3)
draw_heatmap(df_dim3.iloc[:,1:-2], "dim3_abs.png")
#corr=np.abs(df_dim3.iloc[:,1:-2].corr())
#print(corr[corr>0.3])
```

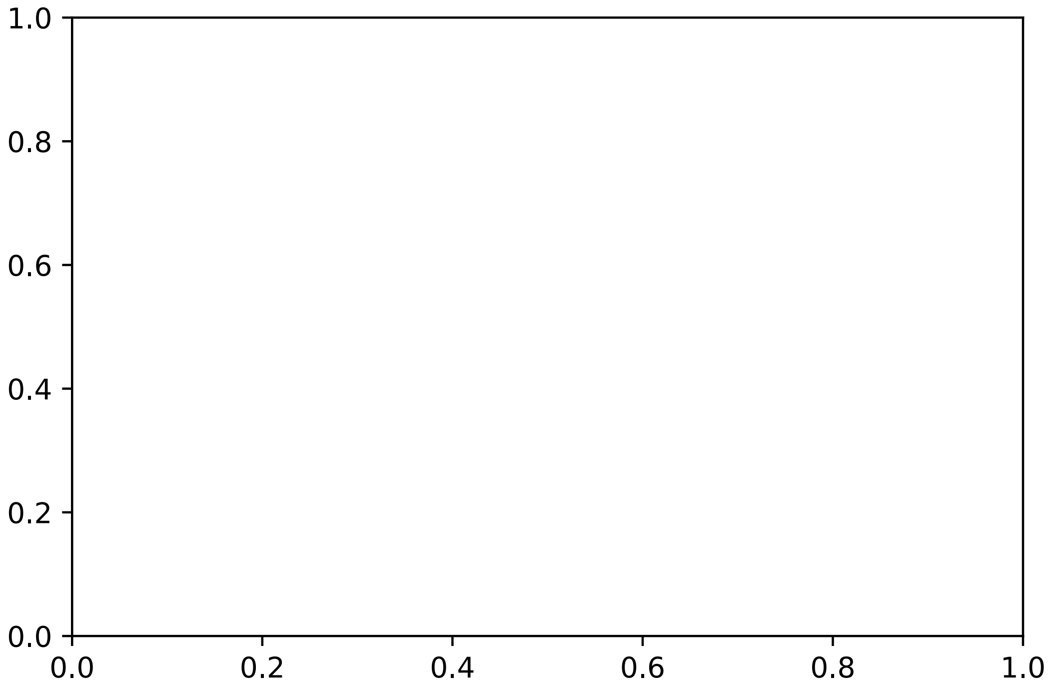
Highest dim:YYY, Size: 34

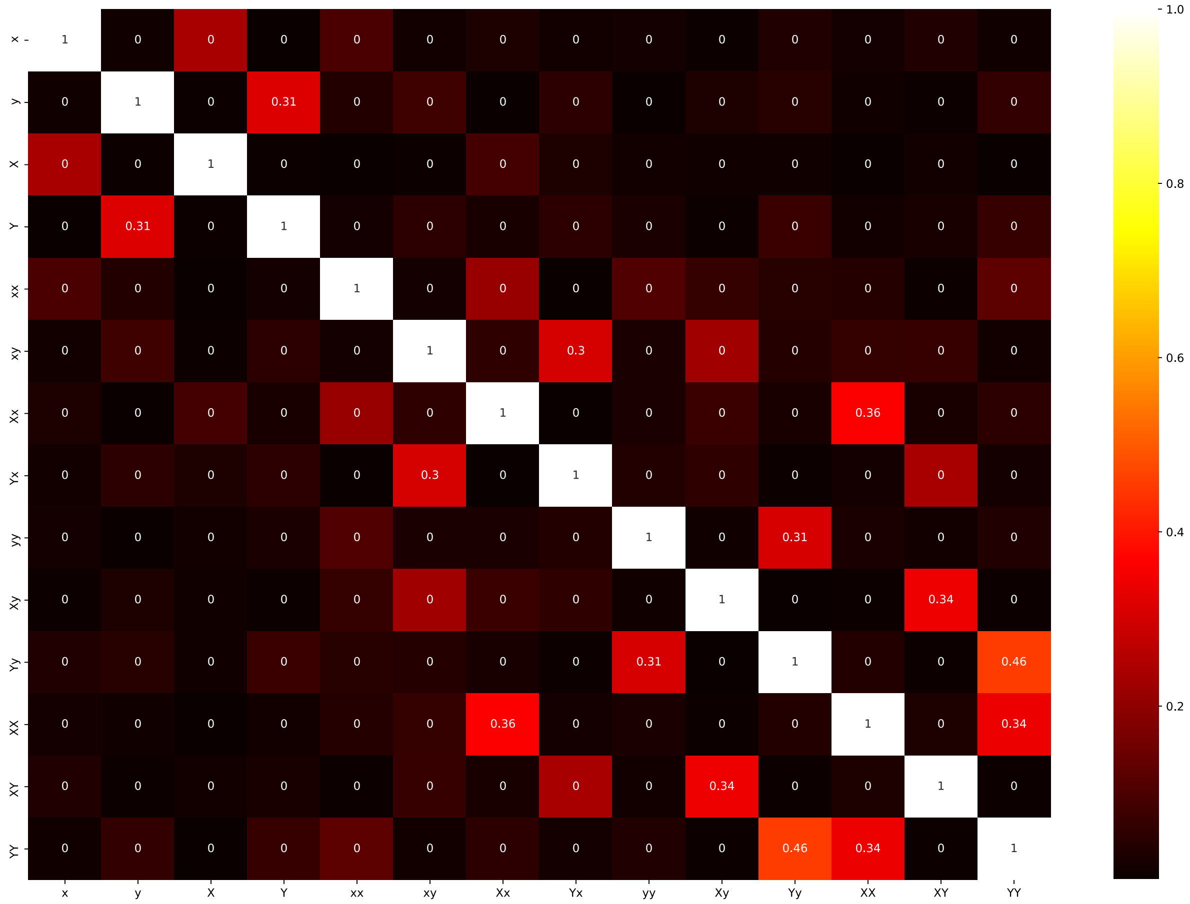




In [210]:
```
(df_dim2, order)=get_df_of_dim(2)
draw_heatmap(df_dim2.iloc[:,1:-2], "dim2_abs.png")
#corr=np.abs(df_dim2.iloc[:,1:-2].corr())
#print(corr[corr>0.3])
```

```
Highest dim:YY, Size:  14
```





In [214]:

```
#Deafult model calculation
df=pd.read_csv("term_project_overlay_data.csv")
num_rows=np.shape(df)[0]
df["XY"]=df["X"]*df["Y"]
df["X2"]=df["X"]*df["X"]
df["Y2"]=df["Y"]*df["Y"]
df["xy"]=df["x"]*df["y"]
df["x2"]=df["x"]*df["x"]
df["y2"]=df["y"]*df["y"]
df["0"]=pd.Series(np.ones(num_rows))
order=["0", "X", "Y", "XY", "X2", "Y2", "x", "y", "xy", "x2", "y2",
"overlay_error_x", "overlay_error_y"]
order1=["0", "Y", "x", "y", "x2", "y2", "overlay_error_x", "overlay
_error_y"]
order2=["0", "X", "Y", "x", "y", "x2", "y2", "overlay_error_x", "ov
erlay_error_y"]
df_ori=df.copy()
df=df_ori[order]
fit(df, order, print_f=True)
get_press(df, order)
#[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, o
rder)
df=df_ori[order1]
get_press(df, order1)
df=df_ori[order2]
get_press(df, order2)

#print(ox_alpha_full)
x_df=df.iloc[:,:-2]
y_df=pd.DataFrame(df.iloc[:,-2])
#param_conf(x_df, y_df, ox_hat_full, 0.05)
#print(oy_alpha_full)
```

```
Full model: ox R2=0.578, oy R2=0.389
ox_press=0.280, oy_press=0.233
ox_press=0.085, oy_press=0.070
ox_press=0.124, oy_press=0.084
```

In [8]:

```
#Calculate and test for dim4, overlay error y.
dim=4
(df, order)=get_df_of_dim(dim)
order_f=backward(dim, False, order)
print(order_f)
df=df[order_f]
[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, or
der_f, print_f=True)
get_press(df, order_f)
print(oy_alpha_full)
```

```
Highest dim:YYYY, Size: 69
Highest dim:YYYY, Size: 69
oy: Delete: yyyy, oy_R2=0.761
cur r2: 0.75978
-----
test statistic: -2.62149
T value: 1.97323
-----
Index(['0', 'x', 'y', 'X', 'Y', 'xx', 'xy', 'Xx', 'Yx', 'yy', 'Xy'
, 'Yy', 'XX',
       'XY', 'YY', 'xxx', 'xxy', 'Xxx', 'Yxx', 'xyy', 'Xxy', 'Yxy'
, 'XXx',
       'XYx', 'YYx', 'yyy', 'Xyy', 'Yyy', 'XXy', 'XYy', 'YYy', 'XX
X', 'XXY',
       'XYY', 'YYY', 'xxxx', 'xxxy', 'Xxxx', 'Yxxx', 'xxyy', 'Xxxy
', 'Yxxy',
       'XXxx', 'XYxx', 'YYxx', 'xyyy', 'Xxyy', 'Yxyy', 'XXxy', 'XY
xy', 'YYxy',
       'XXXx', 'XXYx', 'XYYx', 'YYYx', 'yyyy', 'Xyyy', 'Yyyy', 'XX
yy', 'XYyy',
       'YYyy', 'XXXy', 'XXYy', 'XYYy', 'YYYy', 'XXXX', 'XXXY', 'XX
YY', 'XYYY',
       'YYYY', 'overlay_error_x', 'overlay_error_y'],
      dtype='object')
Full model: ox R2=0.739, oy R2=0.760
ox_press=11.232, oy_press=7.255
[ 5.20468704e-04  6.59718496e-04  1.78824926e-03  8.65829976e-04
  3.93275945e-03  1.25240597e-02  7.72774461e-03  7.55082041e-04
  1.15613479e-04  2.57155839e-02  1.47323035e-04  1.14556790e-04
  1.16934030e-04 -3.15343272e-05 -6.39814033e-05 -4.49774350e-05
 -1.36559843e-04  3.02242845e-05  2.54828256e-06  1.16946282e-04
 -6.40560473e-06  1.25460099e-05  2.76289579e-06  4.21242104e-06
 -8.69238419e-08  7.10531052e-05  7.24033455e-06  7.68192494e-06
  8.06124986e-07 -6.36456785e-07  7.76973768e-07  2.56578919e-09
  7.56830128e-07  4.20368803e-08 -3.04617794e-07 -2.43882021e-04
 -6.19299312e-06 -3.99797037e-06 -7.76041333e-07  1.51485922e-05
  1.15507383e-06  6.50104807e-07 -1.41453666e-07  3.78518339e-07
  4.60342366e-07 -2.71088464e-05 -4.80460264e-07 -8.78138536e-07
 -5.60843859e-08 -1.51001643e-07 -3.61668034e-08 -5.68926622e-08
 -8.24509560e-09 -1.65062430e-08  1.59854132e-08 -8.10547441e-05
 -2.60057808e-07 -1.14082226e-06 -9.49456845e-08 -1.46449087e-07
 -6.35472064e-08 -1.03655124e-08 -2.44163199e-08 -1.33699717e-08
  9.29186292e-09 -6.66059020e-09  2.58358473e-09 -3.71649108e-09
  9.32121261e-10  3.62654303e-09]
```

In [12]:

```
#Calculate and test for dim2, overlay error y.
dim=2
(df, order)=get_df_of_dim(dim)
order_f=backward(dim, False, order)
print(order_f)
df=df[order_f]
[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, or
der_f, print_f=True)
get_press(df, order_f)
print(oy_alpha_full)
```

```
Highest dim:YY, Size: 14
Highest dim:YY, Size: 14
oy: Delete: Xx, oy_R2=0.404
cur r2: 0.40438
-----
test statistic: 0.28510
T value: 1.97011
-----
oy: Delete: XX, oy_R2=0.404
cur r2: 0.40418
-----
test statistic: 0.38377
T value: 1.97007
-----
oy: Delete: Xy, oy_R2=0.402
cur r2: 0.40380
-----
test statistic: 0.81420
T value: 1.97002
-----
oy: Delete: XY, oy_R2=0.401
cur r2: 0.40214
-----
test statistic: 0.56957
T value: 1.96998
-----
oy: Delete: Yx, oy_R2=0.400
cur r2: 0.40132
-----
test statistic: 0.65762
T value: 1.96994
-----
oy: Delete: xy, oy_R2=0.396
cur r2: 0.40024
-----
test statistic: 1.37689
T value: 1.96990
-----
oy: Delete: Yy, oy_R2=0.382
cur r2: 0.39550
-----
test statistic: -2.29737
T value: 1.96986
-----
Index(['0', 'x', 'y', 'X', 'Y', 'xx', 'yy', 'Yy', 'YY', 'overlay_e
rror_x',
       'overlay_error_y'],
      dtype='object')
Full model: ox R2=0.442, oy R2=0.396
ox_press=0.217, oy_press=0.180
[ 8.10852082e-01  1.60295475e-02  1.35749469e-02  2.66576920e-03
  3.52271362e-03 -3.53659903e-03  2.34815636e-03 -1.31748475e-04
 -2.00005742e-05]
```

In [11]:
```
#Calculate and test for dim2, overlay error x.
dim=2
(df, order)=get_df_of_dim(dim)
order_f=backward(dim, True, order)
print(order_f)
df=df[order_f]
[ox_hat_full, oy_hat_full, ox_alpha_full, oy_alpha_full]=fit(df, or
der_f, print_f=True)
get_press(df, order_f)
print(ox_alpha_full)
```

```
Highest dim:YY, Size: 14
Highest dim:YY, Size: 14
ox: Delete: Yx, ox_R2=0.588
cur r2: 0.58848
-----
test statistic: -0.50102
T value: 1.97011
-----
ox: Delete: Yy, ox_R2=0.587
cur r2: 0.58804
-----
test statistic: -0.70844
T value: 1.97007
-----
ox: Delete: Xx, ox_R2=0.585
cur r2: 0.58717
-----
test statistic: -0.99912
T value: 1.97002
-----
ox: Delete: xy, ox_R2=0.583
cur r2: 0.58543
-----
test statistic: -1.07995
T value: 1.96998
-----
ox: Delete: X, ox_R2=0.581
cur r2: 0.58340
-----
test statistic: 1.17125
T value: 1.96994
-----
ox: Delete: YY, ox_R2=0.576
cur r2: 0.58101
-----
test statistic: -1.68155
T value: 1.96990
-----
ox: Delete: Xy, ox_R2=0.569
cur r2: 0.57607
-----
test statistic: -1.95010
T value: 1.96986
-----
ox: Delete: Y, ox_R2=0.561
```

```
cur r2: 0.56938
-----
test statistic: 2.18220
T value: 1.96982
-----
Index(['0', 'x', 'y', 'Y', 'xx', 'yy', 'XX', 'XY', 'overlay_error_
x',
       'overlay_error_y'],
      dtype='object')
Full model: ox R2=0.569, oy R2=0.317
ox_press=0.138, oy_press=0.139
[-3.85509918e-01  6.45878546e-02 -3.45166239e-02  1.23126828e-03
  3.28190408e-03  1.09171428e-03 -4.12123489e-05 -5.64109815e-05]
```

In [ ]: