Applied opto-mechanical statistical tolerancing techniques

By

Christopher L. Hopkins

A Report Submitted to the Faculty of the

COLLEGE OF OPTICAL SCIENCE

In Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

May 8th, 2015

Abstract

Variation exists in all processes to some degree including those used to manufacture optics. A successful optical system is one where the design takes this into account and plans accordingly. This report looks at methods of allocating tolerances using probability theory to meet system level performance specifications. It is concerned with opto-mechanics and addresses those variables which govern placement of optical surfaces. Using techniques from papers written on the topic, this report takes the reader through the tolerancing process by way of example. The performance of a Double Gauss photographic objective is modeled using second-order approximation techniques and tolerances allocated via a statistical budgeting process. The estimates are then compared against Monte Carlo simulation trials with good results.

Table of Contents

1 INTRODUCTION 1

1.1 BACKGROUND 1
1.2 PURPOSE OF REPORT 2
1.3 OVERVIEW 3
1.4 SCOPE 3
1.4.1 Software Used 4
1.5 USEFUL BENEFITS 4

2 METHODOLOGY 5

2.1 SECOND-ORDER APPROXIMATION OF THE MERIT FUNCTION 5
2.2 DERIVATION OF ESTIMATION EQUATIONS 6
2.2.1 Mean 6
2.2.2 Variance 7
2.2.3 Resulting Equations 8
2.2.4 Optimized vs. non-optimized input variables 9
2.3 COMPARISON OF ESTIMATED VALUES AGAINST MONTE CARLO SIMULATION RESULTS 9
2.4 POLYNOMIAL BEHAVIOR 11
2.5 DETERMINING COEFFICIENTS 13

3 SYSTEM CONFIGURATION 15

3.1 OPTICAL LAYOUT 15
3.1.1 Performance comparison to the Summicron-M 50mm f/2 lens 17
3.2 MECHANICAL LAYOUT 17
3.3 PERFORMANCE METRIC 18
3.3.1 Budget goal 19

4 ANALYSIS MODEL SETUP 20

4.1 CHOOSING WHAT TO MODEL AND IDENTIFYING VARIABLES 20
4.2 MODIFYING THE LENS DATA EDITOR 20
4.3 SETTING UP THE TOLERANCE DATA EDITOR 21

5 ANALYSIS 23

5.1 CHOICE OF INPUT VARIABLE STARTING VALUES AND DISTRIBUTIONS 23
5.2 COLLECTING COEFFICIENTS 24
5.3 ESTIMATING STATISTICS 27
5.4 OPTIMIZING TOLERANCES 27
5.4.1 Process setup 27
5.4.2 Spreadsheet setup 28
5.4.3 Iteration Process 29
5.4.4 Resulting Tolerances 30

6 COMPARISON OF MONTE CARLO SIMULATION RESULTS 31

6.1 TAYLOR APPROXIMATION VERSUS ESTIMATES 31
6.2 APPROXIMATION VERSES LENS MODEL 32

7 CONCLUSION 33

7.1 POTENTIAL AREAS OF FUTURE WORK 34

8 REFERENCES 35

APPENDIX A: PROBABILITY THEORY AND TOLERANCING A-1

A.1 RANDOM VARIABLES AND THEIR PROPERTIES A-1
A.1.1 Expected Value and Variance A-2
A.1.2 Useful Probability Distributions for Tolerancing A-4
A.2 FUNCTIONS OF A SINGLE RANDOM VARIABLE A-7
A.3 COMBINING MULTIPLE RANDOM VARIABLES A-7
A.3.1 Independence A-8
A.3.2 Estimating Variance A-9
A.3.3 Estimating Probabilities using the Central Limit Theorem A-11
A.3.4 Estimating Distributions using Monte Carlo Analysis A-15
A.3.5 Simulating Non-Identically Distributed Random Variables A-19

APPENDIX B: SIMULATION TECHNIQUES B-1

B.1 CREATION OF RANDOM RADIAL POSITIONS VECTORS B-1
B.1.1 Choice of Coordinate System B-1
B.1.2 Radial Distribution Types B-2
B.1.3 Working with Modifiers B-4
B.2 ADDITION OF RADIAL POSITION VECTORS B-6
B.3 ADDITION OF ORIENTATION VECTORS B-9
B.4 MODELING SIMULTANEOUS REQUIREMENTS B-11
B.4.1 Effect of Locational Tolerances on Flat Surface Position and Orientation B-11
B.4.2 Effect of Coaxiality on Axis Position and Orientation B-14
B.4.3 Refining Orientation with Parallelism B-19

APPENDIX C: MATLAB SCRIPTS WRITTEN FOR THIS REPORT C-1

C.1 SCRIPT TO COLLECT COEFFICIENTS FROM THE ZEMAX LENS MODEL C-1
C.2 SCRIPT TO PERFORM THE MONTE CARLO ANALYSIS AND ESTIMATE STATISTICS C-3

1 Introduction

1.1 Background

Tolerances are limits we place on measureable physical properties. Some examples are an object's mass and length, or the image quality of an optical system. Such measureable physical properties are referred to as parameters. Tolerances represent a range of uncertainty on a given parameter's value. They can either serve to bound our expectation of what value a parameter may take, as in the case with measurement accuracy, or they can be used to place a limit on which values are acceptable and thereby which values are cause for rejection.

When tolerances are assigned, they are typically chosen on the basis of fit, performance, and/or manufacturing costs. The first criterion, fit, is the property of components within an assembly to not interfere with each other and also be interchangeable with other like components. Meeting performance means our ability to have a desired outcome by limiting the uncertainty of the input parameters. The final item, cost, is a function of both manufacturing volume and yield for a given process. Yield is influenced by our choice of tolerances. Typically tolerances which improve performance increase costs. Selecting the right tolerances for fit can help mitigate this.

The severity of a tolerance limit (defined as either loose or tight) is relative to the process for producing the parameter. A small production yield at a given tolerance limit means the tolerance is tight for the process and each successful attempt can get costly. However, switching to a manufacturing process where the same tolerance is considered loose may not necessarily save money. While the yield may increase, the new process in general may also be more expensive. The overall process and yield must be looked at together.

The performance of optical systems is particularly susceptible to manufacturing variation. Where the wavelength of light is measured in nanometers and micrometers, variations can be of the same order of magnitude or greater. If not accounted for and managed, this can have a detrimental impact.

There are several ways to handle this. One type of method is to make a measurement of the as produced parameter value and adjusting the system to compensate. This reduces uncertainty from that of the manufacturing process to one of measurement accuracy. For example, a system can be re-optimize based on specific melt data from a batch of optical glass; a lens barrel cut to size once lens curvatures are known; or have the design include some type of adjustment mechanism. It each case, the parameter is measured after it was produced. While the measurement itself has an associated uncertainty, most likely it is less than for the manufacturing process.

Tolerances are another method of controlling uncertainty and are the focus of this report. By inspecting the results of the manufacturing process, tolerances are used as criteria for rejecting or accepting the produced results. This is sometimes referred to as "go/no go" gauging. Within the tolerance limits, there is still uncertainty, but now its range is bounded. Component level parameter uncertainties combine to produce an overall system level performance uncertainty. A measureable system level performance parameter is referred to as a performance metric. It is a particular system property against which the merit of the system is evaluated. Examples include wavefront error, resolution, distortion, etc. So in this way, radius of curvature errors for the lens elements within a camera will play a factor in that camera's image quality.

System yield is determined the same way as component yield. It is based on the variability of the performance metric and the tolerance limits we set for the system to either pass or fail. We have the option of choosing component tolerances so system variability falls well within pass/fail criteria giving a 100% yield. This is referred to as worse case tolerancing. It is easy to perform, but results in tight component tolerances which if possible may be exceedingly expensive. Tolerances that result in yields of 99% or 98%, have the potential to be considerably cheaper and easier to produce, but the analysis becomes more complicated.

Predicting the system yield based on uncertainty in the input parameter values is possible through the understanding of probability and statics. Warren Smith gives a good introduction on the topic in his paper "Fundamentals of establishing an optical tolerance budget" (Smith). Appendix A of this report also reviews the basic principles of probability theory.

1.2 Purpose of Report

The intent of this report is to provide the reader with an example of how to create a tolerance budget based on the principles of probability and statistics. It follows the general development process described by Robert Ginsberg in his paper "Outline of tolerancing (from performance specification to tolerance drawings)" (Ginsberg) and the analysis procedure set forth in Donald Koch's paper "A statistical approach to lens tolerancing" (Koch). This report departs slightly from the method of approximating the performance metric suggested by Koch and instead builds off of methods shown in "Cost-based tolerancing of optical systems" by Youngworth and Stone (Youngworth and Stone, Cost-based tolerancing of optical systems).

A modified version of the process development flowchart presented in Ginsberg is shown in Figure 1.1. The blocks have been organized into three main categories: requirements, system configuration, and analysis. The main focus will be on the analysis category. For the system configuration, a double Gauss photographic objective patented by Walter Mandler et al. (Mandler, Edwards and Wagner) was selected as the optical design while a simplified mechanical layout was created to illustrate the process. Those choices determined the performance and constraints shown in the requirements category.

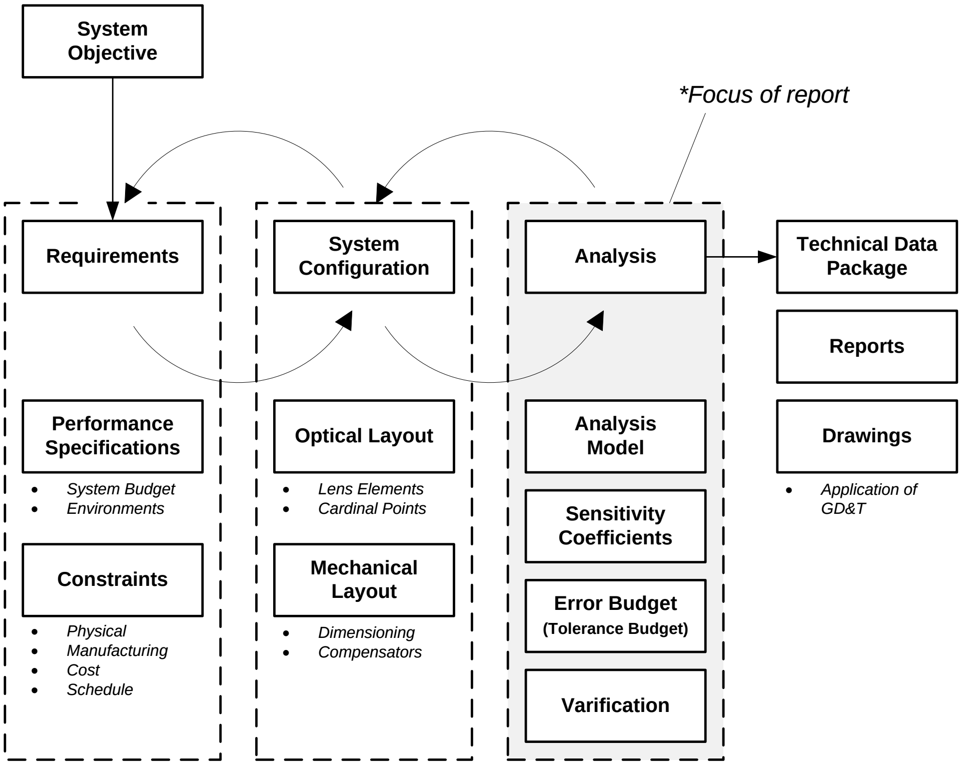


Figure 1.1 Product Development Process Diagram

1.3 Overview

This report begins with a discussion of the methodology used to estimate the system performance metric. It is based on applying a second-order Taylor polynomial to approximate and predict performance. The report then goes through the derivation for estimates of both the mean and standard deviation. Sensitivity coefficients are found by using a quadratic approximation of the performance metric's response to changes in the input parameters.

Next, the specific example of the Double Gauss lens design is introduced and its performance examined. From it, an analysis model is created using the optical design software. This model replicates the opto-mechanical layout with the input parameters of interest. The sensitivity coefficients are collected using the analysis model and fed into the error budget. The tolerances in the error budget are optimized to produce the desired output in the system performance metric. Finally, the optimized tolerances are fed back in the optical design software where a Monte Carlo simulation is run to compare results. This is a simplified description of the process.

1.4 Scope

The example used in this report is of a rotationally symmetric, all-refractive imagine system with focus compensation. The process could also be applied to other types of optical systems that have rotational symmetry with or without compensators and perhaps even non-rotationally symmetric systems. The code used to interface with the optical design software may not function as intended with the latter.

This exercise examines only those tolerances associated with the position and orientation of lens elements due to manufacturing variability in the lens housing (also referred to as a lens barrel). Achromats are treated as a single lens. The analysis could easily be expanded to include other parameters such as lens radii or index if desired. The number of input parameters was chosen so there are enough to combine in a statistically meaningful way. More variables only add to the analysis run time without contributing much more to understanding the process. Each of the environmental parameters is set to the optical design software default values.

All input parameters are given a normal distribution with a cutoff of plus/minus three standard deviations. The normal distribution is chosen because it simplifies the estimates of the mean and standard deviation. This is an optimistic choice for production statistics. Other distributions could be substituted. Again this selection was made to demonstrate the process in a succinct manner.

The performance metric used is the on-axis weighted average of the rms wavefront error at 0.480 µm, 0.546 µm, and 0.644 µm (F', e, and C' spectral lines). Because the original merit function used to optimize the example lens is not known, this performance metric is selected to serve as a proxy for the lens' general imaging performance over the visible spectrum.

1.4.1 Software Used

The example lens is model using Zemax optical design software. An algorithm written in MatLab communicates with Zemax using the Zemax DDE toolbox (Griffith) to collecting sensitivity coefficients. Matlab is also used to estimate the mean and variance of the performance metric. Monte Carlo simulations are run in both Zemax and MatLab. Where Zemax simulates performance statistics using the lens model, MatLab simulates them using the second-order approximation. Finally, the error budget is created and optimized in Microsoft Excel. All MatLab algorithms can be found in Appendix C.

1.5 Useful Benefits

The author found the exercise of running through the tolerancing process to be very helpful in his understanding of the subject and several other related areas. This report will be of interest to anyone wanting to see an example of how to perform a tolerance analysis plus any of these other subjects.

- Use of a second-order Taylor polynomial to approximate a complex system
- Example of incorporating a design's mechanical layout in the optical design software
- Interfacing MatLab with Zemax
- Setting-up Monte Carlo simulations
- Probability and statistics

2 Methodology

In this report, rms wavefront error is chosen as the metric to evaluate the performance of the example lens. Its mean and standard deviation due to perturbation of the lens' input parameters are estimated by way of the delta method. The delta method approximates a continuous function using a Taylor polynomial with the idea that it is easier to find the moments of the Taylor polynomial than it is of the original function (Casella). The exact form of the function being approximated does not even need to be known, there must however exist some way of finding its derivatives.

This section presents the derivation for finding the mean and standard deviation of a second-order Taylor polynomial used to approximate a function of multiple random variables. It is similar to the example Casella and Berger go through for the second-order delta method with a single input variable. All input parameters are assumed to be independent. The resulting derived equations are the same as shown in Youngsworth (Youngworth, Statistical truths of tolerancing assignment in optical design), but slightly different that in Koch. Each also has a slightly different form of the equation used to approximate the merit function. The author believes this is due to how the coefficients are represented.

After the derivation is complete, the estimates are compared to statistics generated from a Monte Carlo simulation run on the Taylor polynomial with good results.

2.1 Second-order approximation of the merit function

A function of a single variable $f(x)$ can be approximated in the vicinity of $x = \alpha$ by

$$f(x) \approx f(\alpha) + \left(\frac{df}{dx}\right)_{\alpha}(x-\alpha) + \frac{1}{2}\left(\frac{d^2 f}{dx^2}\right)_{\alpha}(x-\alpha)^2 \tag{1.1}$$

where α is a constant and derivatives are evaluated at α. The order of the polynomial is a reference to the largest exponent present. The above equation is a second-order Taylor polynomial. If the last term were removed, it becomes first-order. The higher the order, the better the Taylor polynomial approximates $f(x)$.

A function of more than one variable $f(x_1, x_2, \ldots, x_n)$ can similarly be approximated by a multivariate Taylor polynomial. Given two vectors $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]$, the difference can be written as a vector $\mathbf{\Delta x} = [(x_1 - \alpha_1), (x_2 - \alpha_2), \ldots, (x_n - \alpha_n)]$. The Taylor polynomial written in matrix notation, similar to that shown in Youngworth (Youngworth and Stone, Cost-based tolerancing of optical systems), is

$$f(\mathbf{x}) \approx f(\boldsymbol{\alpha}) + \mathrm{J}(\mathbf{x})_{\alpha}\,\Delta\mathbf{x}^{\mathrm{T}} + \tfrac{1}{2}\Delta\mathbf{x}\,\mathrm{H}(\mathbf{x})_{\alpha}\,\Delta\mathbf{x}^{\mathrm{T}}. \tag{1.2}$$

The two terms $\mathrm{J}(\mathbf{x})$ and $\mathrm{H}(\mathbf{x})$ are referred to as the Jacobian and Hessian matrix respectively and represented as

$$\mathrm{J}(\mathbf{x})=\left[\frac{\partial f}{\partial x_1},\frac{\partial f}{\partial x_2},\dots\frac{\partial f}{\partial x_N}\right],\quad \mathrm{H}(\mathbf{x})=\begin{bmatrix}\frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N}\\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2}\end{bmatrix}.$$

The Jacobian is vector of partial derivatives while the Hessian is a square matrix of second-order partial derivatives. The individual terms are evaluated at $x_i = \alpha_i$.

Eq.(1.2) can also be expressed in a series of summations as

$$f(\mathbf{x}) \approx f(\boldsymbol{\alpha}) + \sum_{i=1}^{N}\left(\frac{1}{2}\frac{\partial^2 y}{\partial x_i^2}\Delta x_i^2 + \frac{\partial y}{\partial x_i}\Delta x_i\right) + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\Delta x_i \Delta x_j. \quad (1.3)$$

2.2 Derivation of estimation equations

Now supposing x is a random variable with some continuous probability distribution (represented as a capital letter X), then $f(\mathbf{X}) = Y$, where $\mathbf{X} = [X_1, X_2, \dots, X_n]$ is a vector of random variables, will itself be a random variable with a mean value and standard deviation. We want to estimate what these values are provided that we know the mean and standard deviation for X_i and assuming that $X_1, X_2, \dots, X_n$ are all independent.

2.2.1 Mean

The first step to estimating the mean is to rewrite eq.(1.2) and take the expectation of both sides where the expectation of Y is written as $\mathrm{E}[Y]$,

$$Y \approx y_0 + \mathrm{J}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}} + \tfrac{1}{2}\Delta\mathbf{X}\,\mathrm{H}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}}$$
$$\mathrm{E}[Y] \approx \mathrm{E}\left[y_0 + \mathrm{J}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}} + \tfrac{1}{2}\Delta\mathbf{X}\,\mathrm{H}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}}\right]$$

Y is evaluated at $\boldsymbol{\alpha}$ and $y_0 = f(\boldsymbol{\alpha})$. Next, the expectation operator is distributed among the terms on the right side and noted that if c is a constant, then $\mathrm{E}[c] = c$, $\mathrm{E}[cX] = c\mathrm{E}[X]$, and $\mathrm{E}[\Delta X] = 0$.

$$\begin{aligned}\mathrm{E}[Y] &\approx \mathrm{E}[y_0] + \mathrm{J}(\mathbf{x})\mathrm{E}\left[\Delta\mathbf{X}^{\mathrm{T}}\right] + \tfrac{1}{2}\mathrm{E}\left[\Delta\mathbf{X}\,\mathrm{H}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}}\right]\\ &\approx y_0 + \tfrac{1}{2}\mathrm{E}\left[\Delta\mathbf{X}\,\mathrm{H}(\mathbf{x})\Delta\mathbf{X}^{\mathrm{T}}\right]\end{aligned}$$

The remaining terms are then written out as a series of summations

$$\mathrm{E}[Y] \approx y_0 + \frac{1}{2}\mathrm{E}\left[\sum_{i,j=1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\Delta X_i \Delta X_j\right]$$

The expected value of the product of two random variables is the covariance of the two variables, cov[X,Y] and if $X_i = X_j$ then cov[X_i, X_j] = var[X_i]. All terms where $i = j$ are collected together.

$$\begin{aligned}\mathrm{E}\left[Y\right] &\approx y_0 + \frac{1}{2}\sum_{i,j=1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\mathrm{cov}\left[\Delta X_i, \Delta X_j\right] \\ &\approx y_0 + \frac{1}{2}\left(\sum_{i=1}^{N}\frac{\partial^2 y}{\partial x_i^2}\mathrm{var}\left[\Delta X_i\right] + 2\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\mathrm{cov}\left[\Delta X_i, \Delta X_j\right]\right)\end{aligned}$$

The assumption was made that all the inputs are independent and therefore cov[X,Y] = 0, $X \neq Y$. What we are left with is then

$$\mathrm{E}[Y] \approx y_0 + \frac{1}{2}\sum_{i=1}^{N}\frac{\partial^2 y}{\partial x_i^2}\mathrm{var}\left[\Delta X_i\right]. \tag{1.4}$$

2.2.2 Variance

The variance of a random variable X is defined as var[X] = E[X^2]-E[X]^2. We begin by taking the variance of both sides of eq.(1.2)

$$\mathrm{var}\left[Y\right] \approx \mathrm{var}\left[y_0 + \mathrm{J}\left(\mathbf{x}\right)\Delta\mathbf{X}^{\mathrm{T}} + \tfrac{1}{2}\Delta\mathbf{X}\,\mathrm{H}\left(\mathbf{x}\right)\Delta\mathbf{X}^{\mathrm{T}}\right]$$

The variance of any constant is zero, so y_0 is eliminated and the remaining terms are rewritten a series of summations.

$$\mathrm{var}\left[Y\right] \approx \mathrm{var}\left[\sum_{i=1}^{N}\frac{\partial y}{\partial x_i}\Delta X_i + \frac{1}{2}\sum_{i,j=1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\Delta X_i \Delta X_j\right]$$

Next the terms where $i = j$ are separated from those where $i \neq j$

$$\mathrm{var}\left[Y\right] \approx \mathrm{var}\left[\sum_{i=1}^{N}\left(\frac{1}{2}\frac{\partial^2 y}{\partial x_i^2}\Delta X_i^2 + \frac{\partial y}{\partial x_i}\Delta X_i\right) + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{\partial^2 y}{\partial x_i \partial x_j}\Delta X_i \Delta X_j\right]$$

Noting that var[$aX + bY$] = a^2var[X]+b^2var[Y]+2ab cov[X, Y] and that cov[X, Y] = 0, we have a general expression for the variance of Y that is valid for any distribution of X_i.

$$\mathrm{var}\left[Y\right] \approx \sum_{i=1}^{N}\left(\frac{1}{4}\left(\frac{\partial^2 y}{\partial x_i^2}\right)^2 \mathrm{var}\left[\Delta X_i^2\right] + \left(\frac{\partial y}{\partial x_i}\right)^2 \mathrm{var}\left[\Delta X_i\right]\right) + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left(\frac{\partial^2 y}{\partial x_i \partial x_j}\right)^2 \mathrm{var}\left[\Delta X_i \Delta X_j\right] \tag{1.5}$$

It was stated earlier that ΔX_i has a mean of zero, therefor $\mathrm{E}[\Delta X_i\ ] = 0$. With that, it is possible to further simplify $\mathrm{var}[\Delta X_i^2\ ]$ and $\mathrm{var}[\Delta X_i\ ,\ \Delta X_j\ ]$ (keep in mind that all covariance terms are zero).

$$\begin{aligned}\mathrm{var}\left[\Delta X_i^2\right] &= E\left[\Delta X_i^4\right] - E\left[\Delta X_i^2\right]^2 \\ &= E\left[\Delta X_i^4\right] - \left(\mathrm{var}\left[\Delta X_i\right] + E\left[\Delta X\right]^2\right)^2 \\ &= E\left[\Delta X_i^4\right] - \mathrm{var}\left[\Delta X_i\right]^2\end{aligned}$$

and, if $i \neq j$, then

$$\begin{aligned}\mathrm{var}\left[\Delta X_i \Delta X_j\right] &= E\left[\Delta X_i^2 \Delta X_j^2\right] - E\left[\Delta X_i \Delta X_j\right]^2 \\ &= E\left[\Delta X_i^2\right] E\left[\Delta X_j^2\right] - E\left[\Delta X_i \Delta X_j\right]^2 \\ &= \left(\mathrm{var}\left[\Delta X_i\right] + E\left[\Delta X_i\right]^2\right)\left(\mathrm{var}\left[\Delta X_j\right] + E\left[\Delta X_j\right]^2\right) - E\left[\Delta X_i \Delta X_j\right]^2 \\ &= \mathrm{var}\left[\Delta X_i\right]\mathrm{var}\left[\Delta X_j\right] + E\Delta X_j^2\,\mathrm{var}\left[\Delta X_i\right] + E\Delta X_i^2\,\mathrm{var}\left[\Delta X_j\right] \\ &= \mathrm{var}\left[\Delta X_i\right]\mathrm{var}\left[\Delta X_j\right].\end{aligned}$$

So that the previous eq.(1.5) can be rewritten as

$$\begin{aligned}\mathrm{var}\left[Y\right] \approx \sum_{i=1}^{N}&\left(\frac{1}{4}\left(\frac{\partial^2 y}{\partial x_i^2}\right)^2 \left(E\left[\Delta X_i^4\right] - \mathrm{var}\left[\Delta X_i\right]^2\right) + \left(\frac{\partial y}{\partial x_i}\right)^2 \mathrm{var}\left[\Delta X_i\right]\right) \\ &+\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left(\frac{\partial^2 y}{\partial x_i \partial x_j}\right)^2 \mathrm{var}\left[\Delta X_i\right]\mathrm{var}\left[\Delta X_i\right]\end{aligned} \tag{1.6}$$

where $\mathrm{E}[\Delta X_i^4]$ is referred to as the fourth moment of the distribution of X_i.

2.2.3 Resulting Equations

If X_i is normally distributed X_i ~n(0,σ^2), then $\mathrm{var}[\Delta X_i\ ] = \sigma_i^2$ and $\mathrm{E}[\Delta X_i^4] = 3\sigma_i^4$. Making these substitutions into the previous equations gives the formula used to estimate the mean shift $\Delta\mu_{\mathrm{Y}}$ and standard deviation σ_{Y} of the merit function

$$\Delta\mu_Y \approx \frac{1}{2}\sum_{i=1}^{N}\frac{\partial^2 y}{\partial x_i^2}\sigma_{\Delta X_i}^2 \tag{1.7}$$

$$\sigma_Y^2 \approx \sum_{i=1}^{N}\left(\frac{1}{2}\left(\frac{\partial^2 y}{\partial x_i^2}\right)^2 \sigma_{\Delta X_i}^4 + \left(\frac{\partial y}{\partial x_i}\right)^2 \sigma_{\Delta X_i}^2\right) + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left(\frac{\partial^2 y}{\partial x_i \partial x_j}\right)^2 \sigma_{\Delta X_i}^2 \sigma_{\Delta X_j}^2 \tag{1.8}$$

This report refers to the off-diagonal terms of the Hessian matrix as cross-terms or cross-derivatives. If the cross-terms are zero then eq.(1.8) for the standard deviation σ_Y is further simplified to

$$\sigma_Y^2 \approx \sum_{i=1}^{N}\left(\frac{1}{2}\left(\frac{\partial^2 y}{\partial x_i^2}\right)^2 \sigma_{\Delta X_i}^4 + \left(\frac{\partial y}{\partial x_i}\right)^2 \sigma_{\Delta X_i}^2\right) \tag{1.9}$$

Non-zero cross-terms represent a coupling effect of independent input variables on the performance metric. An example of a non-zero cross-term is two lenses that have a degree-of-freedom in the same direction. If the cross-term is zero between two variables, then those two variables are orthogonal such as the decenter of a lens in both the x and y-axes. Cross-terms can contribute significantly to the estimate for standard deviation and must be taken into account.

2.2.4 Optimized vs. non-optimized input variables

If $\partial f\left(\mathbf{X}\right)/\partial\Delta X_i = 0$ when $\Delta X_i = 0$, then $f(\mathbf{X})$ is optimized (well-corrected) with respect to ΔX_i. The nominal value of X_i results in $f(\mathbf{X})$ being at a minimum value. Otherwise, if it is non-zero, then it is referred to as non-optimized for that particular value. This can occur if a system constraint exists that prevents the optimum value from being reached. The following figure illustrates the difference.

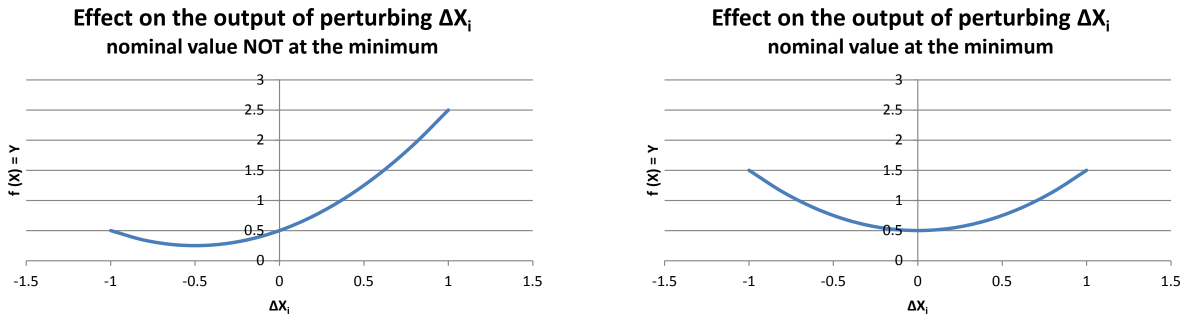


Figure 2.1 Example of the merit function's response to a non-optimized input variable (left image) verses one that is optimized (right image).

Most often the radial position of lens elements are at their optimum value r = 0. Spacing between elements must content with minimum gaps for manufacturability and therefore may not be optimal with respect to the performance metric.

2.3 Comparison of estimated values against Monte Carlo simulation results

The estimates derived in the previous section for the mean and standard deviation are next compared against Monte Carlo simulation results of the second-order Taylor polynomial. The comparison uses a fictitious system with N = 5 independent, identically distributed random variables each with a distribution of X_i ~n(0, 0.1^2). Three different cases each with a different set of sensitivity coefficients are considered. The first is well-corrected with no cross-terms. The second adds cross terms that are

50% of the value of the terms along the main diagonal of the Hessian matrix. Finally, the third adds non-zero first-order partial derivatives making the nominal values of X_i no longer optimized.

Because the mean shift is dependent only on the main diagonal of the Hessian matrix, it is the same for all three cases. Its estimate is

$$\Delta\mu_Y \approx \frac{5}{2}(10)(0.1)^2 = 0.250$$

The system coefficients; estimate calculation for the output standard deviation; and results of the Monte Carlo simulation are presented in the following figures. All results show the estimates are in close agreement with the simulation statistics. It is worth noting that the first-order derivatives of the Jacobian have the greatest impact on the resulting output distribution.

<u>System Coefficients:</u>

$$J(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \quad H(x) = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{bmatrix}$$

<u>Estimate of the Standard Deviation:</u>

$$\sigma_Y \approx \left(\frac{5}{2}(10)^2(0.1)^4\right)^{1/2}$$
$$\approx 0.158$$

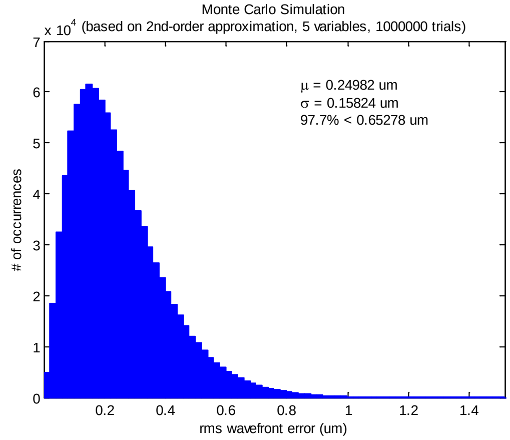


Figure 2.2 Statistics generated from a Monte Carlo simulation of the Taylor polynomial compared with estimation results for the standard deviation of a system where the inputs are all well optimized and there are no cross-terms

<u>System Coefficients:</u>

$$J(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \quad H(x) = \begin{bmatrix} 10 & 5 & 5 & 5 & 5 \\ 5 & 10 & 5 & 5 & 5 \\ 5 & 5 & 10 & 5 & 5 \\ 5 & 5 & 5 & 10 & 5 \\ 5 & 5 & 5 & 5 & 10 \end{bmatrix}$$

<u>Estimate of the Standard Deviation:</u>

$$\sigma_Y \approx \left(\frac{5}{2}(10)^2(0.1)^4 + 10(5)^2(0.1)^4\right)^{1/2}$$
$$\approx 0.224$$

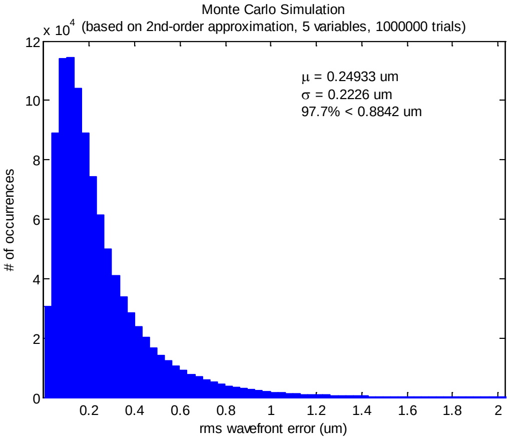


Figure 2.3 Statistics generated from a Monte Carlo simulation of the Taylor polynomial compared with estimation results for the standard deviation of a system where the inputs are all well optimized and cross-terms exist

System Coefficients:

$$J(x)=\begin{bmatrix}10\\10\\10\\10\\10\end{bmatrix};\quad H(x)=\begin{bmatrix}10&5&5&5&5\\5&10&5&5&5\\5&5&10&5&5\\5&5&5&10&5\\5&5&5&5&10\end{bmatrix}$$

Estimate of the Standard Deviation:

$$\sigma_Y \approx \left(\frac{5}{2}(10)^2(0.1)^4+5(10)^2(0.1)^2+10(5)^2(0.1)^4\right)^{1/2}$$
$$\approx 2.247$$

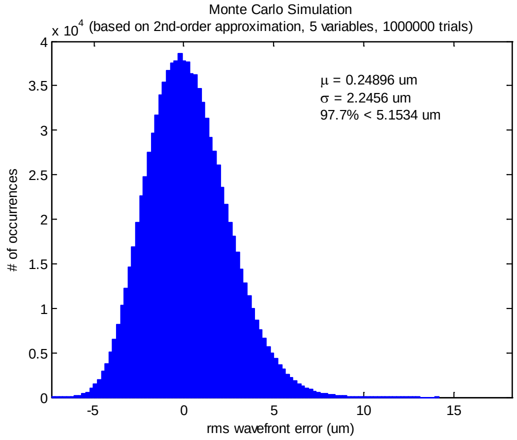


Figure 2.4 Statistics generated from a Monte Carlo simulation of the Taylor polynomial compared with estimation results for the standard deviation of a system where the inputs are not optimized and cross-terms exist

2.4 Polynomial Behavior

With the tools now in place to analyze the Taylor polynomial, next we will examine its behavior. Specifically we will look at how cross-terms and the number of input variables effect the output distribution on a well-corrected system (first-derivatives are zero). Again this uses the fictitious system of the previous section.

Let us first look at how many input variables it takes for the output to approximate a normal distribution (cross-terms are zero) in accordance with the Central Limit Theorem. By definition, being well-corrected, the output will have its minimum value when the inputs are at their respective means. The value of the output can only ever increase. When the mean shift is on the order of the standard deviation or less, the distribution is "pressed-up" against this minimum absolute value. This is shown for the case where N=2 in the following figure (Δμ/σ = 1).

As more variables are added, the mean shifts away from the minimum absolute value faster than the standard deviation increases. In other words, the ratio of Δμ/σ increases as N does. This has the effect of giving the left tail room to form and resemble a normal distribution. Koch suggests eight independent variables are sufficient for creating a budget based on a normal distribution. The author believes this would serve as the bare minimum. More variables will help improve the accuracy of the estimate.

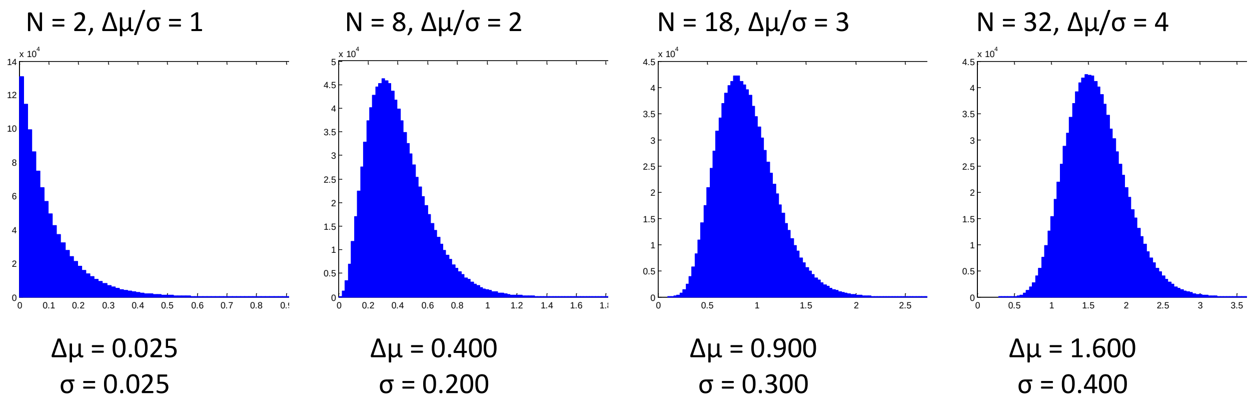


Figure 2.5 Change in the merit value distribution for an optimized system of inputs with no cross-terms as the number of variables is increased.

Next we want to examine the impact cross-terms have on the output distribution. In this study, all cross-terms are equal and each is expressed as a percentage of the individual values along the main diagonal of the Hessian matrix. In the Monte Carlo analysis of the previous section, a cross-term of 5 and main diagonal value of 10 represents a 50% weighting of cross-terms.

The next figure shows the necessary number of input variables N to achieve a Δμ/σ = 3 for a cross-term weighting of 0%, 10%, and 20%. As the percentage increases, the number of input variables needed increases dramatically.

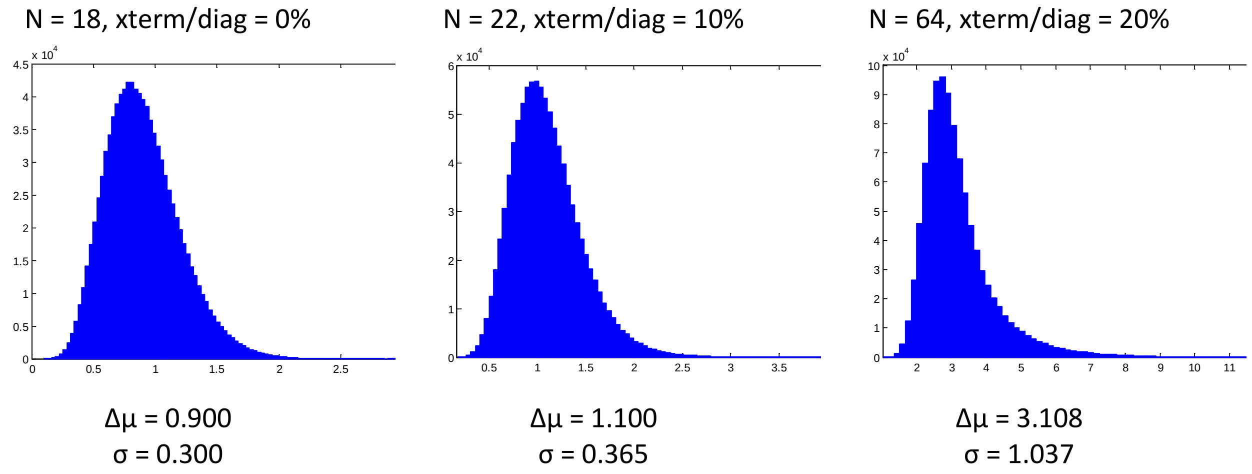


Figure 2.6 Comparison of the number of independent input variables required to achieve the same ratio of Δμ/σ for three systems with different cross-term weighting percentages.

The relationship between the number of input variables, Δμ/σ, and cross-term weighting is presented in the next figure. Two things to note are that 1) after a certain number of input variables, the slope becomes nearly constant and 2) there is a limiting value for Δμ/σ that gets smaller as the relative influence of cross-terms increases. So if the approximation includes a lot of non-orthogonal variables, it may be that the output never assumes a near normal distribution regardless of how many there are.

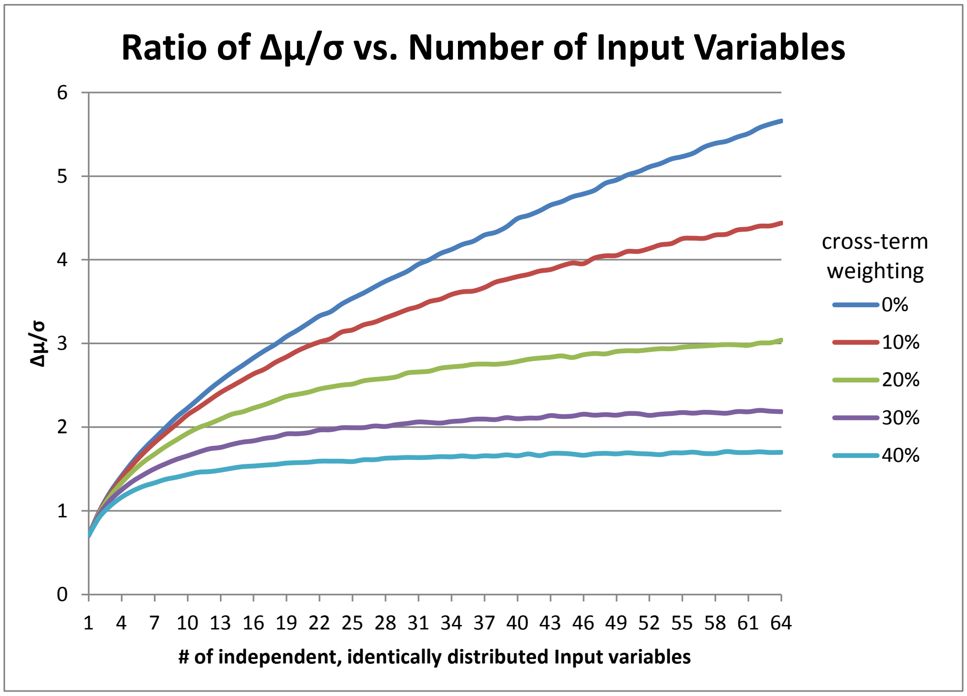


Figure 2.7 Ratio of Δμ/σ verses the number of independent identically distributed input variables in a given system.

The change that occurs in Δμ/σ when the cross-terms are removed from the Hessian matrix can serve as a gauge for their relative importance. This property will be useful later when creating a tolerance budget.

2.5 Determining Coefficients

A second-order Taylor polynomial approximates the relationship between the input variables and the output as the quadratic function

$$y = ax^2 + bx + c$$

The first and second order derivatives of this function are $2ax+b$ and $2a$ respectively. If $|x|<<|b|$, then the first derivative is simply b. Therefore if we know the coefficients a and b, we can calculate the derivatives.

Given the three points (x_0, y_0), (x_1, y_1), (x_2, y_2) shown in the figure below it is possible to calculate these coefficients and therefore the derivatives with the requirement that $x_1 = -x_2$, $x_0 = 0$, and $y_0 = c$.

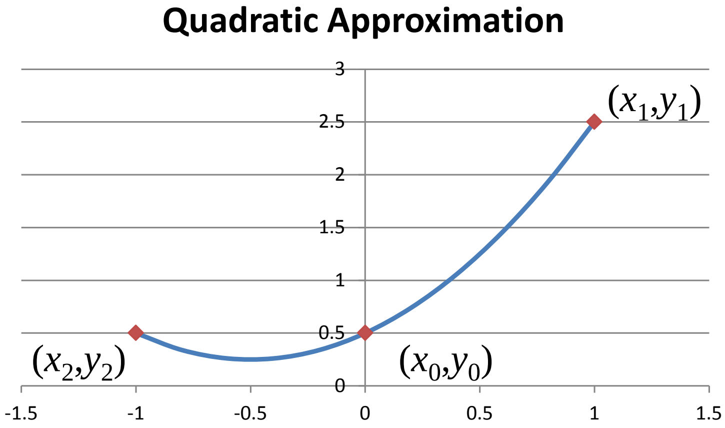


Figure 2.8 Example of a quadratic curve and the three points used to define it.

$$a = \frac{1}{2x_1^2}\left(y_1 + y_2 - 2y_0\right) \qquad \frac{\partial^2 y}{\partial x^2} = \frac{1}{x_1^2}\left(y_1 + y_2 - 2y_0\right)$$

$$b = \frac{1}{2x_1}\left(y_1 - y_2\right) \qquad \frac{\partial y}{\partial x} = \frac{1}{2x_1}\left(y_1 - y_2\right)$$

Extending this concept, the second-order partial derivative of two variables becomes

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{f\left(x+\Delta x, y+\Delta y\right) - f\left(x-\Delta x, y+\Delta y\right) - f\left(x+\Delta x, y-\Delta y\right) + f\left(x-\Delta x, y-\Delta y\right)}{4\Delta x\Delta y} \quad (1.10)$$

This is the equation as presented in Youngworth with a slight change in notation (Youngworth and Stone, Cost-based tolerancing of optical systems).

Depending on the quality of the quadratic approximation fit, the estimate will be sensitive to the choice of ±Δx relative to its tolerance limits. This error will show up when comparing results against Monte Carlo simulations done in the optical design software. Youngworth recommends choosing values that are at the tolerance limit or more to minimize this error. The author has noted that this indeed helps.

3 System Configuration

The example lens for this report is the Double Gauss photographic objective described in US Patent# 4,123,144 (Mandler, Edwards and Wagner) under example number 9 which has been re-scaled to a focal length of 50 mm. The author has found that this particular example most closely simulates the Summicron-M 50 mm f/2 prime focus lens produced by Leica as described in Jonas & Thorpe and the Leica sales literature. The layouts in both the patent and Leica sales brochure are shown side-by-side in the following figure.

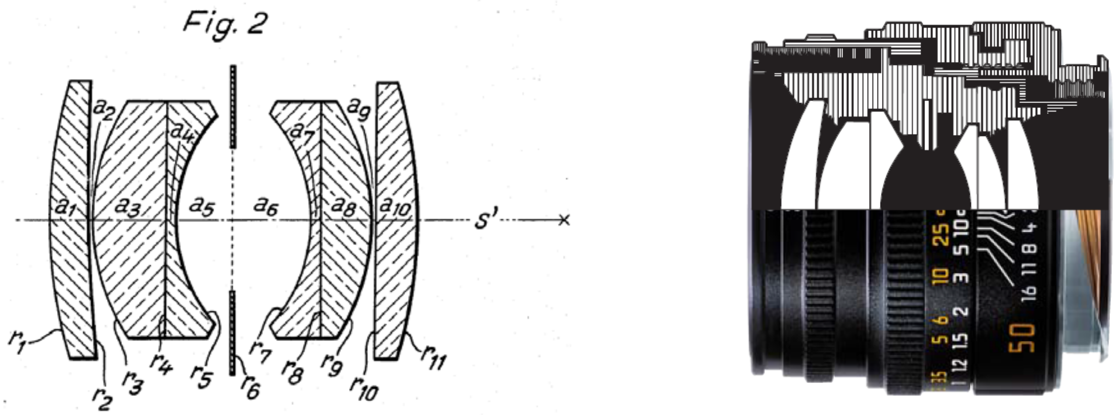


Figure 3.1 A comparison between the figure referenced by example 9 of the patent (Mandler, Edwards and Wagner) (left image) and a cut-away illustration of the Summicron-M 50 mm f/2 (Leica Camera AG) (right image).

This lens was selected for its excellent image quality and manufacturability. Designed in 1975, it is considered one of the best standard 50 mm lenses available (Jonas and Thorpe). A complete re-design incorporating aspheric surfaces was put on the market in 2013.

There are several constraints placed on this design to increase manufacturability and reduce costs. The surface between each achromat lens pair is flat (r = ∞) along with the inside surface of the rear lens. Both the outer surfaces of the front and rear lenses have matched radii as do the outer surfaces of the two achromats. The overall length is fixed as are the air gaps between achromats and the outer lenses. Despite the loss of design freedom to correct aberrations, performance is still good (Jonas and Thorpe).

3.1 Optical layout

To create the model lens, the example 9 lens in the patent is rescaled to the Summicron-M 50 mm f/2 specifications shown in the sales brochure. The following is a list of the design parameters used.

Table 3.1 Design parameters used in creating the model lens

	Eff. Focal Length	Exit Pupil Dia.	f/#	Half Field-of-View
Summicron-M 50 mm f/2*	52.3 mm	25.1 mm	2.08	23.5
Model Lens	52.3 mm	25.0 mm	2.08	22

**Information proved by Leica sales brochure*

The resulting surface data for the model lens is presented in Table 3.2. In determining the clear aperture (CA) of each lens, the stop is fixed at f/2 on-axis then the field angle of 22 degrees added and vignetting factors set. This results in the semi-aperture shown in the table. The lens surface outer diameter (O.D.) is set 10% greater that the clear aperture diameter.

The most difficult part of modeling this lens was finding appropriate glass. Several of the glass types originally used are suspected to have been from Schott and discontinued. Jonas & Thorpe noted the same issue in their effort to model this lens as well.

Table 3.2 Lens data table

Surface	Radius (mm)	Thickness (mm)	Mfg	Glass Code (d)	Semi-Aperture (mm)	Lens O.D. (mm)
1	31.334	5.000	Sumita	788474	14.449	31.78
2	87.463	0.200			13.317	29.30
3	21.067	7.500	Hikari	668419	11.366	25.00
4	∞	1.500	Schott	728284	9.728	25.00
5	13.419	5.650			8.154	17.94
6 (stop)		7.000			7.826	15.65
7	-14.475	1.000	Ohara	626357	7.269	16.00
8	∞	4.000	Ohara	717476	9.181	22.12
9	-21.067	0.200			10.051	22.12
10	∞	4.500	Ohara	717479	11.828	27.42
11	-31.334					

Next is the optical layout of the model lens. The fields shown are at 0, 10, 15.4, and 22 degrees.

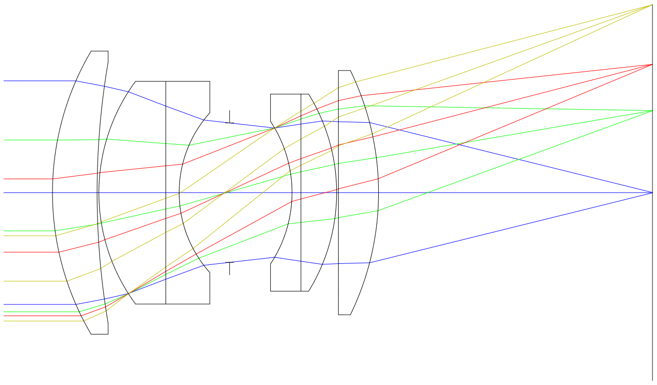


Figure 3.2 Optical layout of the double Gauss photographic objective

The z-axis position of the focal plane is adjusted to minimize the merit function. This is done through a damped least squares algorithm which is part of the optical design software. More about the merit function is discussed later under the section titled performance metric.

3.1.1 Performance comparison to the Summicron-M 50mm f/2 lens

The Leica sales brochure includes a plot of the MTF vs. field angle at the f/2 stop position with white light. This data is presented next to the same type of plot generated from the model lens with the vignetting factors removed. It is unknown if the sales brochure data is as-designed or as-built or if it includes vignetting factors. The on-axis MTF of the model lens at 5 line pairs per mm (LP/mm) is 95% while 40 LP/mm is about 50%. The MTF stays flat with about 10% variation at 5, 10, 20, and 40 LP/mm out to 17 degrees of field angle where it begins to drop-off. Total field angle is 22 degrees. Overall the model lens matches the Leica data very closely.

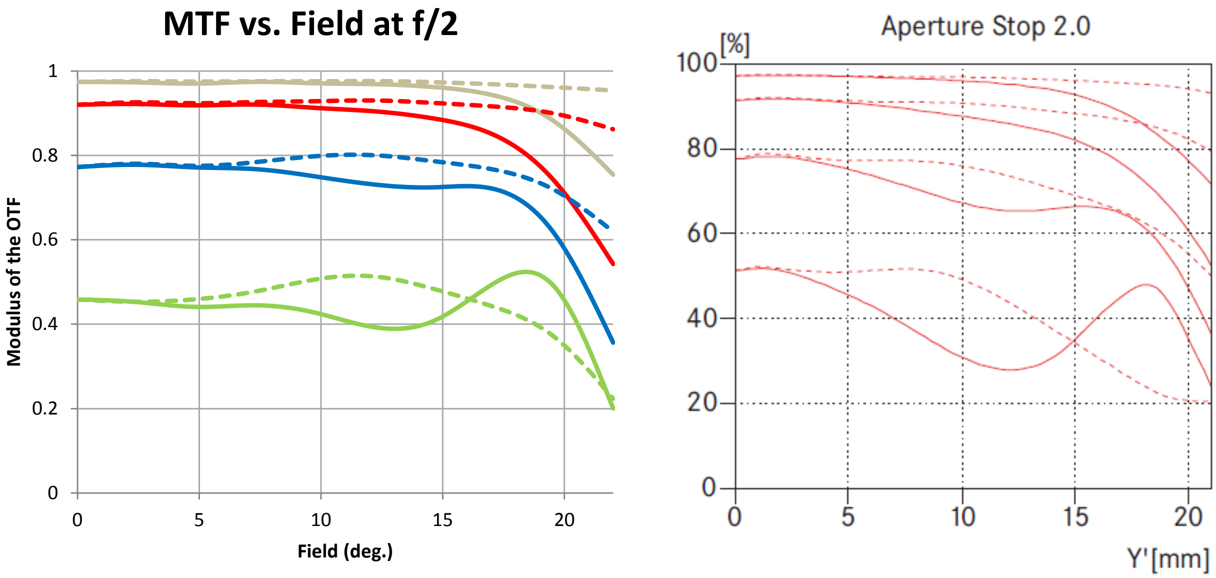


Figure 3.3 Comparison of MTF verses field at f/2 between the model lens (left image) and the Summicron-M 50 mm f/2 (Leica Camera AG).

3.2 Mechanical Layout

The mechanical layout shown in Figure 3.4 is a greatly simplified version of a 50 mm prime lens (a lens of fixed focal length) used in 35 mm film photography. It has a fixed aperture and adjustable focus. The major components shown are the adjustable housing, fixed body, bayonet adapter, focus adjustment ring, and light shade. The Leica M-mount bayonet adapter was chosen because the flange distance of 27.8 mm is compatible with the 29.3 mm back focal distance of the lens design.

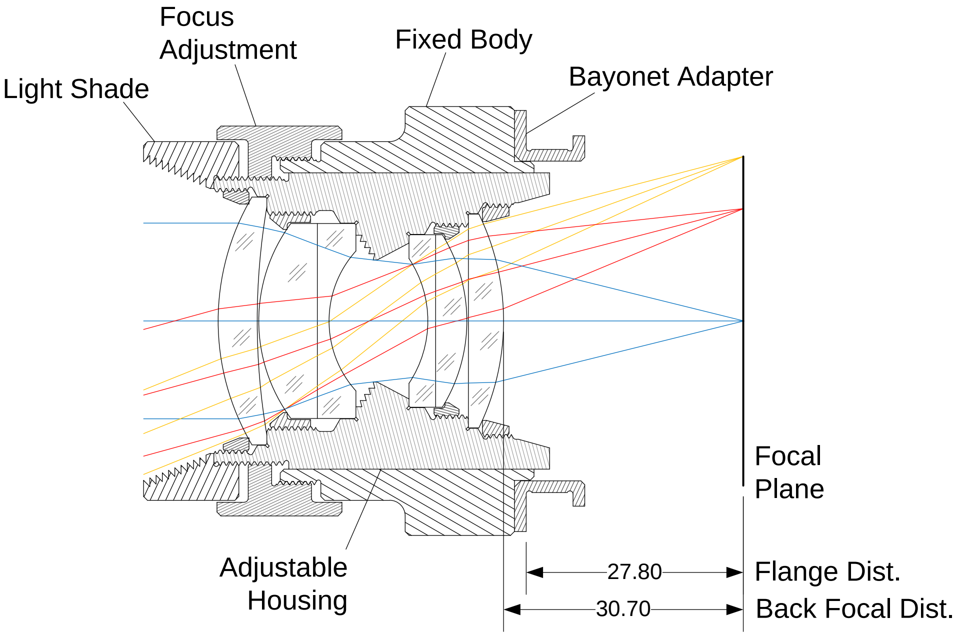


Figure 3.4 Mechanical Layout of a simplified housing for the double Gauss photographic objective

Focus adjustment is accomplished by turning the focus adjustment ring. The ring simultaneously mates with right-hand threads on the fixed body and left-hand threads on the adjustable housing. The housing slides within the fixed body and is kept from rotating itself by a pin that rides in a slot (not shown). The housing translates at twice the speed as the ring relative to the fixed body as the ring is rotated.

All of the lens elements mount in a similar fashion. The first lens element and both achromats are meniscus shaped with sag flats ground perpendicular to the optical axis. The last lens element mounts up against its optical surface. All have a ground outer diameter coaxial to the optical axis and each is meant to be dropped in place without additional centering adjustment. Finally, they are all held in place via threaded retainers.

To simplify the analysis, the following considerations were made. The adjustable aperture stop normally found on such a lens was omitted because it increases the design complexity without adding anything of value to this exercise. Also, the adjustable housing is a one-piece barrel design.

3.3 Performance Metric

The performance metric is calculated by the optical design software using a merit function. The merit function includes the exact inputs and method for combining them together to arrive at the performance metric's value. In the normal course of designing an optical system, the merit function used to optimize the layout is known. That is not the case here with this example. Instead a suitable merit function is chosen that is representative of image quality.

In looking at the optical path difference in the following figures, on-axis and at the 15.4 degree field (70.7% of full field), the peak-to-valley (PTV) wavefront error is between 1 λ and 2 λ in most cases with a max of 3 λ at 0.664 µm on-axis. These values are arrived at by shifting the focal plane using the merit function described below. A general rule of thumb suggested by the Zemax manual is to use wavefront error if the PTV error is < 2 λ which is the method chosen for this report. The on-axis wavefront error appears to be a suitable proxy for image quality. Any other metric could be selected, but this will work for detailing the process.

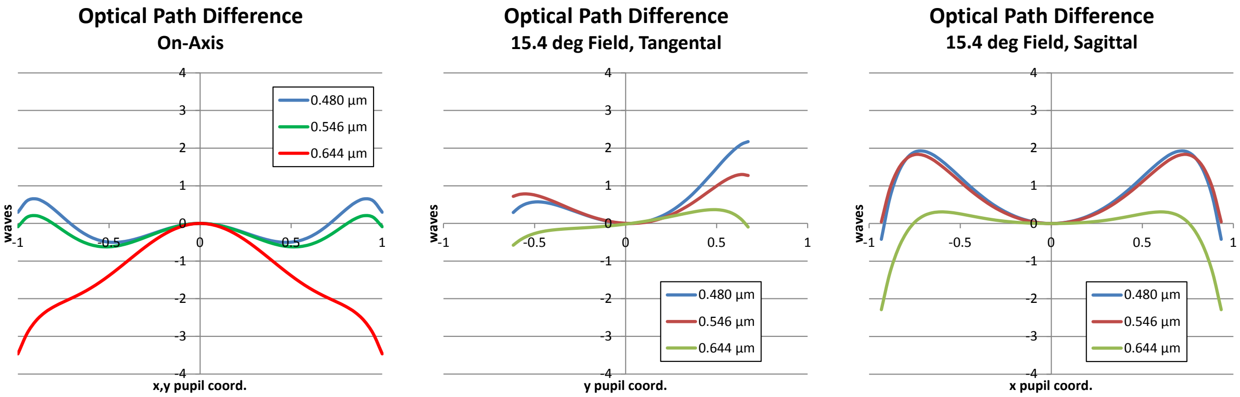


Figure 3.5 The optical path difference as a function of pupil coordinates, wavelength, and field height for the model lens.

The software allows the user to either select a default or create a custom merit function. This report uses the default merit function RMS wavefront centroid with a pupil integration method of Gaussian quadrature consisting of 5 rings and 6 arms. It was found the value of the merit function stabilized at these numbers. Only the on-axis fields for 0.480 µm, 0.546 µm, and 0.644 µm (F', e, and C' spectral lines) are evaluated at f/2. The performance metric is the square root of the weighted sum of squares between the target and actual values for each of these wavelengths over the pupil. The position of the focal plane is optimized by the software to minimize the merit value using a damped least squares algorithm.

The resulting merit function output is 0.5508 µm rms or about 1 λrms at 0.546 µm. This is the result when all of the input parameters are at their respective nominal values and is represented by y_0 in the Taylor polynomial approximation.

3.3.1 Budget goal

As a budgeting goal, tolerances are chosen such that manufacturing variations of the lens barrel contribute about 1 λrms at 2σ to wavefront error. It is therefore desired that the system has an rms wavefront error with a 97.7% (2σ) yield at or less than 2 λrms at 0.546 µm (1.10 µm).

4 Analysis Model Setup

The purpose of the analysis model is to represent within the optical design software the physical relationship between the input variables as shown in the mechanical layout. It is easy to get this wrong as it is difficult to verify the accuracy of the setup. Also, it may not be possible to represent in the optical design software all the particular degrees of freedom in the mechanical layout. The key to a good model is careful setup and understanding its limitations. The following are the major steps.

1. Decide what to model
2. Identify contributing variables
3. Modify lens data editor with pivot surfaces as needed
4. Setup tolerance data editor with operands & surface grouping

4.1 Choosing what to model and identifying variables

To illustrate the process, the analysis is limited to placement of the four lens elements of the double Gauss objective within the adjustable housing (lens barrel). Each achromat is considered as a single lens element when considering its position. The placement of each lens is normally associated with six degrees-of-freedom, but because it is rotationally symmetric about the optical axis (z-axis), only five are non-trivial. There are three translations (along the x, y, & z-axes) and two rotations (about the x and y-axes). In total there are a minimum of 20 (5-dof X 4 lenses) independent variables to consider. There could be additional ones that act on the same degree-of-freedom; such as if the lens barrel were broken up into sub-cells, but that is not the case here. 20 variables are enough to demonstrate the methods herein statistically without being overwhelming.

4.2 Modifying the lens data editor

To simulate tilt, two surfaces are specified in Zemax. All surfaces up to and including those specified are pivoted as a group about the location of the first surface. If the two surfaces are the front and back of a lens, the lens will pivot about its front vertex in the axis specified by the choice of tolerance operator. In the mechanical layout, the orientation of each lens is determined by the orientation of its respective mounting surface (see Figure 4.1). The only lens where this coincides with its front vertex is the last element. The correct pivot location for the other three elements are added as dummy surfaces by modifying the lens data table. The procedure is only required if an analysis involves tilts. For shifts along the optical axis and decenters this step can be skipped.

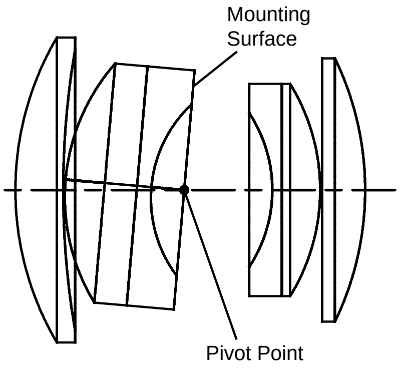


Figure 4.1 Example showing the pivot location of the first achromat

The next figure shows the distance from the front vertex of the lens to its respective mounting surface. The pivot surface needs to be the first surface of the group in the lens data editor and thickness adjusted accordingly. Note that the last lens pivots on its front vertex so no dummy surface was needed.

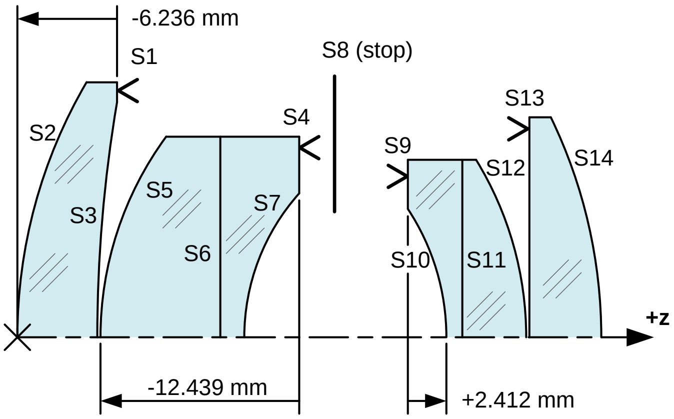


Figure 4.2 Lens layout showing mounting surface locations

Table 4.1 Revised lens data table showing pivot surfaces

Surface	Radius (mm)	Thickness (mm)	Mfg	Glass Code (d)	Semi-Aperture (mm)
1 - pivot		-6.236			
2	31.334	5.000	Sumita	788474	14.449
3	87.463	12.639			13.317
4 - pivot		-12.439			
5	21.067	7.500	Hikari	668419	11.366
6	∞	1.500	Schott	728284	9.728
7	13.419	5.650			8.154
8 (stop)		4.588			7.826
9 - pivot		2.412			
10	-14.475	1.000	Ohara	626357	7.269
11	∞	4.000	Ohara	717476	9.181
12	-21.067	0.200			10.051
13 - pivot	∞	4.500	Ohara	717479	11.828
14	-31.334				

4.3 Setting up the tolerance data editor

To vary a single dimensional parameter such as thickness, tilt, or decenter, Zemax must actually modify two numbers because it operates in a sequential fashion. There is usually a driving dimension and a compensating dimension which moves in the opposite direction by an equal amount. By choosing the various surface pairs, it is possible to simulate different effects

Consider the S4 surface. When selected as the driving surface, the choice of compensator can model one of three different changes in a physical parameter as listed in the table and illustrated in the figure below.

Table 4.2 Example of various surface groupings and their meaning

Case #	Driving Surface	Compensating Surface	Physical Parameter
1	S4	S5	Thickness change of the front achromat element
2	S4	S6	Thickness change of the rear achromat element
3	S4	S7	Change in the front vertex to mounting flat dist.

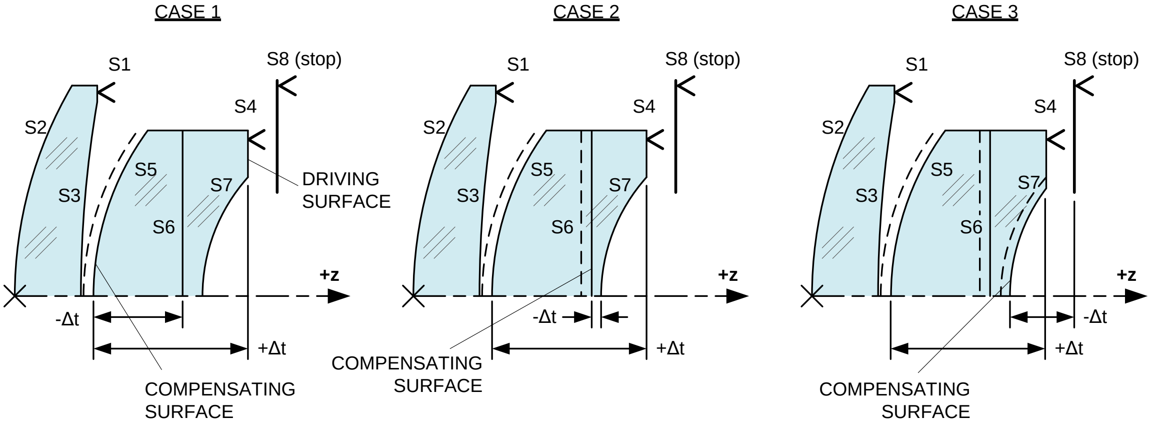


Figure 4.3 Example showing various surface groupings and their meaning

The above three cases are for modeling variations in the achromat itself. In this analysis, we are only concerned with the location of the achromat as determined by variations in the lens barrel. To model that change along the z-axis, S3 becomes the driving surface and S7 the compensating surface.

In a similar fashion, surface pairs are chosen for each parameter to be varied. The parameter is identified via a tolerance operator which tells the optical design software what to change and the axis involved (if the particular parameter is indeed associated with an axis). The tolerance operators used in this analysis along with their associated surfaces are presented in the next two tables.

Table 4.3 Tolerance operators used in the analysis and their description

Operator	Description	Axis
TTHI	Surface thickness change	Translation in Z
TEDX	Decenter	Translation in X
TEDY	Decenter	Translation in Y
TETX	Tilt	Rotation about X
TETY	Tilt	Rotation about Y

Table 4.4 Setup of the tolerance data editor

	TTHI		TEDX/Y, TETX/Y	
Lens/Lens Group	Driving Surf.	Comp. Surf.	Driving Surf.	Comp. Surf
L1	3	14	1	3
L2/3	3	7	4	7
L4/5	8	12	9	12
L6	12	14	13	14

5 Analysis

We are now ready to start the analysis. This is where we derive a set of component tolerances which will meet the goal of our particular performance metric. The inputs needed to begin the analysis are the analysis model, the stated budget goal, and any pre-defined tolerance limits base on the system configuration (such as minimum air gaps between elements). Figure 5.1 illustrates the process steps for performing the analysis.

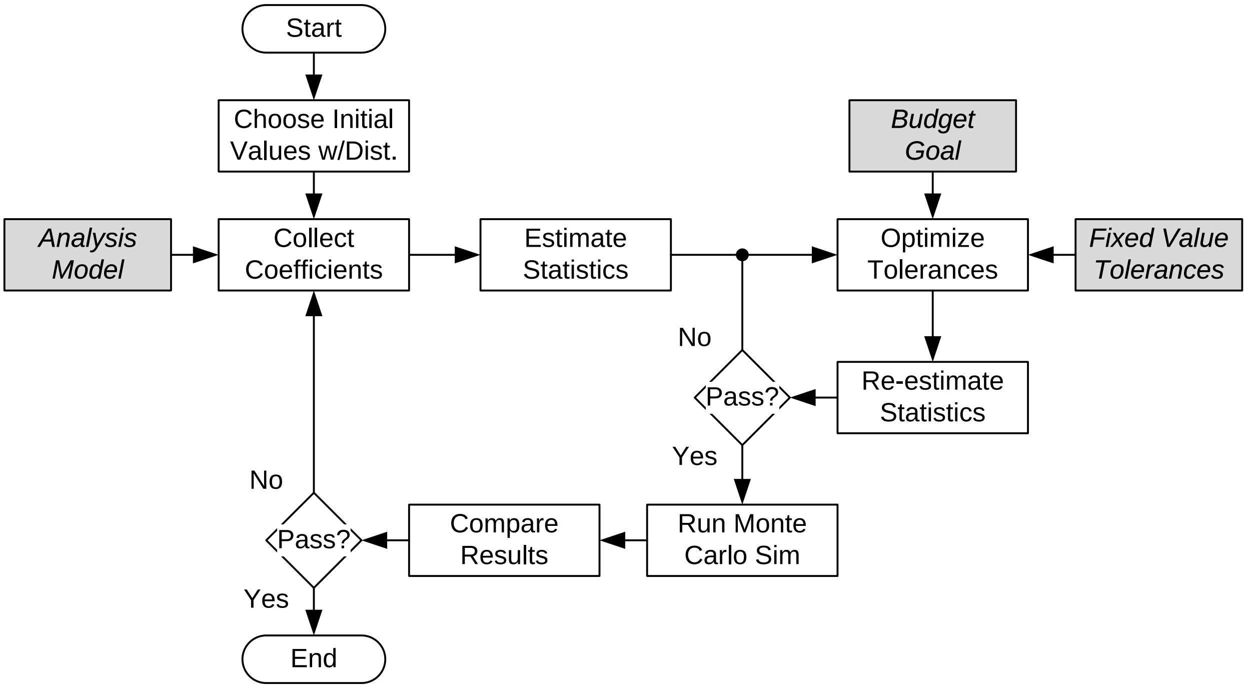


Figure 5.1 Process flow used in performing the tolerance analysis. Process inputs shown in gray.

5.1 Choice of input variable starting values and distributions

All of the input variables for this analysis are defined as normally distributed with the mean centered on their respective nominal values. The tolerance limits represent the ±3σ value. Koch suggests matching the shop production statics if available and his paper presents a list of various distribution types as does Youngworth (Youngworth, Statistical truths of tolerancing assignment in optical design) and Appendix A of this report.

Before selecting the starting tolerance values, let us examine the effect of perturbing each of the lenses. The following three plots illustrate what happens to the merit function when each lens is perturbed radially, titled, and moved along the direction of the optical axis (axial).

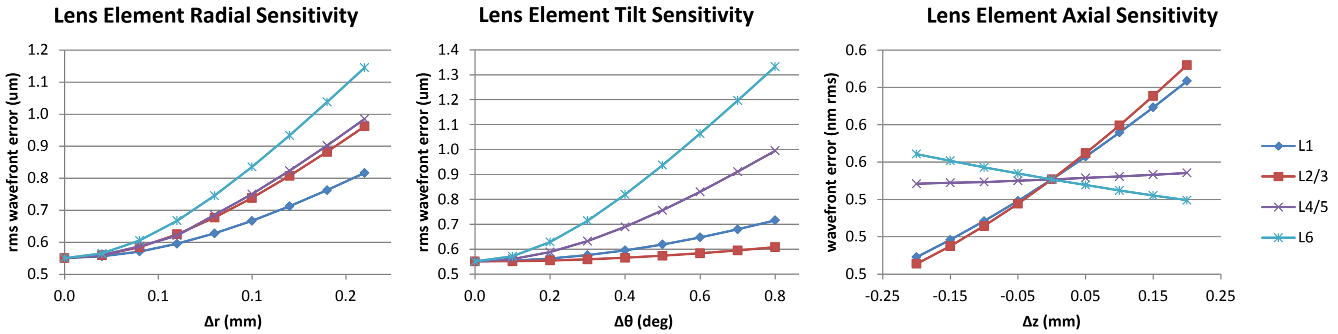


Figure 5.2 Response of the merit function to changes in decenter, tilt, and z-axis translation for each of the four lens elements

The most important requirement has been met; all are continuous over the chosen limit. Radial displacement and tilt are only ever positive and the merit function is well-optimized at their nominal values of zero. This is to be expected for a radially symmetric system. The merit function is not well optimized along the optical axis. This could be the result of competing requirements such as minimum air-gaps for manufacturability.

The starting tolerance values entered into the Zemax tolerance data editor are shown in the next table. The idea is to pick numbers which are thought to be in the neighborhood of the final values. The tolerance operator descriptions are given in Table 4.3.

Table 5.1 Starting tolerance values

	TTHI (±tol mm)	TEDX/Y (±tol mm)	TETX/Y (±tol deg.)
L1	0.200	0.160	0.800
L2/3	0.200	0.160	0.800
L4/5	0.200	0.160	0.800
L6	0.200	0.160	0.800

5.2 Collecting coefficients

The Lens Sensitivity Collection Routine (LSCR) shown in Appendix C.1 was written specifically for this report to calculate the sensitivity coefficients of the lens model. This routine communicates with Zemax to read out the operators and values from the tolerance data editor. It then sends commands to perturb the lens and have Zemax re-calculate the merit function. All perturbations are equal to the values set in the tolerance data editor. The routine uses the new merit function values to build the Jacobian and Hessian matrix.

Recall from section 2 that the Jacobian is a vector of first-order partial derivatives of the merit function with each derivative evaluated at its respective nominal value. It describes the slope of the merit function in its unperturbed state. Non-zero slopes for a particular input variable indicate that its nominal value does not minimize the merit function.

For our example, the Jacobian has 20 elements (four lenses multiplied by five parameters per lens). All parameters exhibiting radial symmetry are zero while translations along the z-axis are non-zero (otherwise not optimized).

Table 5.2 Starting Jacobian vector values for the example lens

L1					L2/3					L4/5					L6				
TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY
.236	0	0	0	0	.266	0	0	0	0	.015	0	0	0	0	-.062	0	0	0	0

The Hessian matrix is a square matrix of second order partial derivatives and is the size of the Jacobian squared (20^2 = 400 elements). The starting coefficients are presented in Table 5.3. Along the main diagonal are those derivatives with respect to a single variable. Larger values indicate the slope is changing faster than smaller values. Note that the well-optimized variables with a slope of zero in the Jacobian tend to have large second derivatives. Looking at Figure 5.2, L6 has the largest tilt slope change. This is verified in the Hessian matrix where tilt second derivatives are 0.52, 0.18, 1.39, and 2.45 for L1, L2/3, L4/5, and L6 respectively.

Also of interest is the symmetry shown in Table 5.3. The cross-derivative terms are symmetric about the main diagonal. There are also terms that run parallel to the main diagonal (indicated with the gray boxes) that are approximately of the same magnitude. These values are associated with perturbations that occur along the same axes. Running perpendicular to the main diagonal are terms that couple between axes (indicated by boxes with outlines). For example, a lens that translates along the x-axis and another lens that tilts about the y-axis have a correlated effect on the merit value to the degree indicated by the coefficient. Wherever the values are zero, the two variables have an independent effect on the wavefront error. A final form of symmetry to note is that if the Hessian matrix is split into sixteen matrices with each one matching the five variables of one lens to another, each 5x5 matrix is comparatively similar.

These coefficients found in the Jacobian and Hessian matrix for the Double Gauss lens are all that is needed to approximate the wavefront error with respect to the parameters we plan to vary. They are unique to this lens and merit function[1]. It can be seen for a complete analysis of all the possible contributors to wavefront error; the Hessian matrix can become very large.

[1] This is analogous to the output of a cryptographic hash function. Such a function generates a unique identifier called a hash for a given set of data. Strong hash functions are unlikely to produce the same identifier for different data. For example, it would be impossible to include the hash for this report here in the body of the report.

Table 5.3 Starting Hessian matrix values for the example lens

		L1					L2/3					L4/5					L6				
		TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY	TTHI	TEDX	TEDY	TETX	TETY
L1	TTHI	**0.28**	0	0	0	0	0.33	0	0	0	0	0.09	0	0	0	0	-0.13	0	0	0	0
	TEDX	0	**20.75**	0	0	1.85	0	-17.85	0	0	-0.72	0	-17.04	0	0	3.13	0	19.13	0	0	-3.61
	TEDY	0	0	**20.75**	-1.85	0	0	0	-17.85	0.72	0	0	0	-17.04	-3.13	0	0	0	19.13	3.61	0
	TETX	0	0	-1.85	**0.52**	0	0	0	2.17	-0.23	0	0	0	2.60	0.55	0	0	0	-2.70	-0.62	0
	TETY	0	1.85	0	0	**0.52**	0	-2.17	0	0	-0.23	0	-2.60	0	0	0.55	0	2.670	0	0	-0.62
L2/3	TTHI	0.33	0	0	0	0	**0.40**	0	0	0	0	0.10	0	0	0	0	-0.16	0	0	0	0
	TEDX	0	-17.85	0	0	-2.17	0	**32.09**	0	0	0.88	0	21.17	0	0	-3.91	0	-24.35	0	0	4.68
	TEDY	0	0	-17.85	2.17	0	0	0	**32.09**	-0.88	0	0	0	21.17	3.91	0	0	0	-24.35	-4.68	0
	TETX	0	0	0.72	-0.23	0	0	0	-0.88	**0.18**	0	0	0	-1.22	-0.28	0	0	0	1.21	0.31	0
	TETY	0	-0.72	0	0	-0.23	0	0.88	0	0	**0.18**	0	1.22	0	0	-0.28	0	-1.21	0	0	0.31
L4/5	TTHI	0.09	0	0	0	0	0.10	0	0	0	0	**0.03**	0	0	0	0	-0.04	0	0	0	0
	TEDX	0	-17.04	0	0	-2.60	0	21.17	0	0	1.22	0.000	**33.92**	0	0	-4.54	0	-25.68	0	0	5.44
	TEDY	0	0	-17.04	2.60	0	0	0	21.17	-1.22	0	0	0	**33.92**	4.54	0	0	0	-25.68	-5.44	0
	TETX	0	0	-3.13	0.55	0	0	0	3.91	-0.28	0	0	0	4.54	**1.39**	0	0	0	-4.98	-1.13	0
	TETY	0	3.13	0	0	0.55	0	-3.91	0	0	-.28	0	-4.54	0	0	**1.39**	0	4.98	0	0	-1.13
L6	TTHI	-0.13	0	0	0	0	-0.16	0	0	0	0	-0.04	0	0	0	0	**0.06**	0	0	0	0
	TEDX	0	19.13	0	0	2.70	0	-24.35	0	0	-1.21	0	-25.68	0	0	4.98	0	**46.54**	0	0	-6.33
	TEDY	0	0	19.13	-2.70	0	0	0	-24.35	1.21	0	0	0	-25.68	-4.98	0	0	0	**46.54**	6.33	0
	TETX	0	0	3.61	-0.62	0	0	0	-4.68	0.31	0	0	0	-5.44	-1.13	0	0	0	6.33	**2.45**	0
	TETY	0	-3.61	0	0	-0.62	0	4.68	0	0	0.31	0	5.44	0	0	-1.13	0	-6.33	0	0	**2.45**

5.3 Estimating statistics

Now that we have the coefficients, an estimate can be made of the delta mean and the standard deviation based on the initial tolerances (both with and without cross-terms). Another algorithm run in MatLab was written to perform this function (see Appendix C.2). The result is

$$\Delta\tilde{\mu}_Y = 0.6535; \quad \tilde{\sigma}_Y = 0.4473; \quad \cancel{\tilde{\sigma}}_Y = 0.2754.$$

The tilde designates these as estimated values. σ is the standard deviation with cross-terms and $\cancel{\sigma}$ represents the standard deviation with cross-terms set to zero. These estimated statistics will be used to scale the mean and standard deviation to arrive at a desired performance and yield.

5.4 Optimizing tolerances

The goal of optimization is to have a set of tolerance values that meet the desired yield percentage at the specified merit value. It is also desirable that each input tolerance chosen contributes an equal amount to the overall standard deviation if possible (Koch). Equal weighting will help with yield predictions based on a standard normal distribution. Sometimes however this is not possible and tolerances must be fixed at a max or min level for some other reason.

5.4.1 Process setup

The budget goal was stated earlier to have a 97.7% yield value of 0.55 µm rms. An educated guess needs to be made for the delta mean and standard deviation $\Delta\mu_Y$ and σ_Y which will meet that.

Assuming ΔY has a normal distribution, a standard normal lookup table shows that Z equals 2.

$$P(\Delta Y \le 0.55) = P\left(Z \le \frac{0.55 - \Delta\mu_Y}{\sigma_Y} \right) = 0.977 \rightarrow \frac{0.55 - \Delta\mu_Y}{\sigma_Y} = 2$$

This gives one relationship between $\Delta\mu_Y$ and σ_Y. The second is an assumption that the ratio between them is the same as the estimated values from the previous step.

$$\Delta\mu_Y + 2\sigma_Y = 0.55; \quad \text{where } \frac{\Delta\mu_Y}{\sigma_Y} = \frac{\Delta\tilde{\mu}_Y}{\tilde{\sigma}_Y}$$

Combining these two relations together results in ratio of the previously estimated statistic to what the new, scaled value is believed to be.

$$\frac{\Delta\mu_Y + 2\sigma_Y}{\Delta\tilde{\mu}_Y + 2\tilde{\sigma}_Y} = \frac{0.55}{0.6535 + 0.4473} = 0.355 \tag{1.11}$$

Using the ratio of 0.355, the scaled statistics are calculated as

$$\Delta\mu = (0.355)\Delta\tilde{\mu}_Y = 0.232; \quad \sigma_Y \approx (0.355)\tilde{\sigma}_Y = 0.159; \quad \cancel{\sigma}_Y \approx (0.355)\tilde{\cancel{\sigma}}_Y = 0.098$$

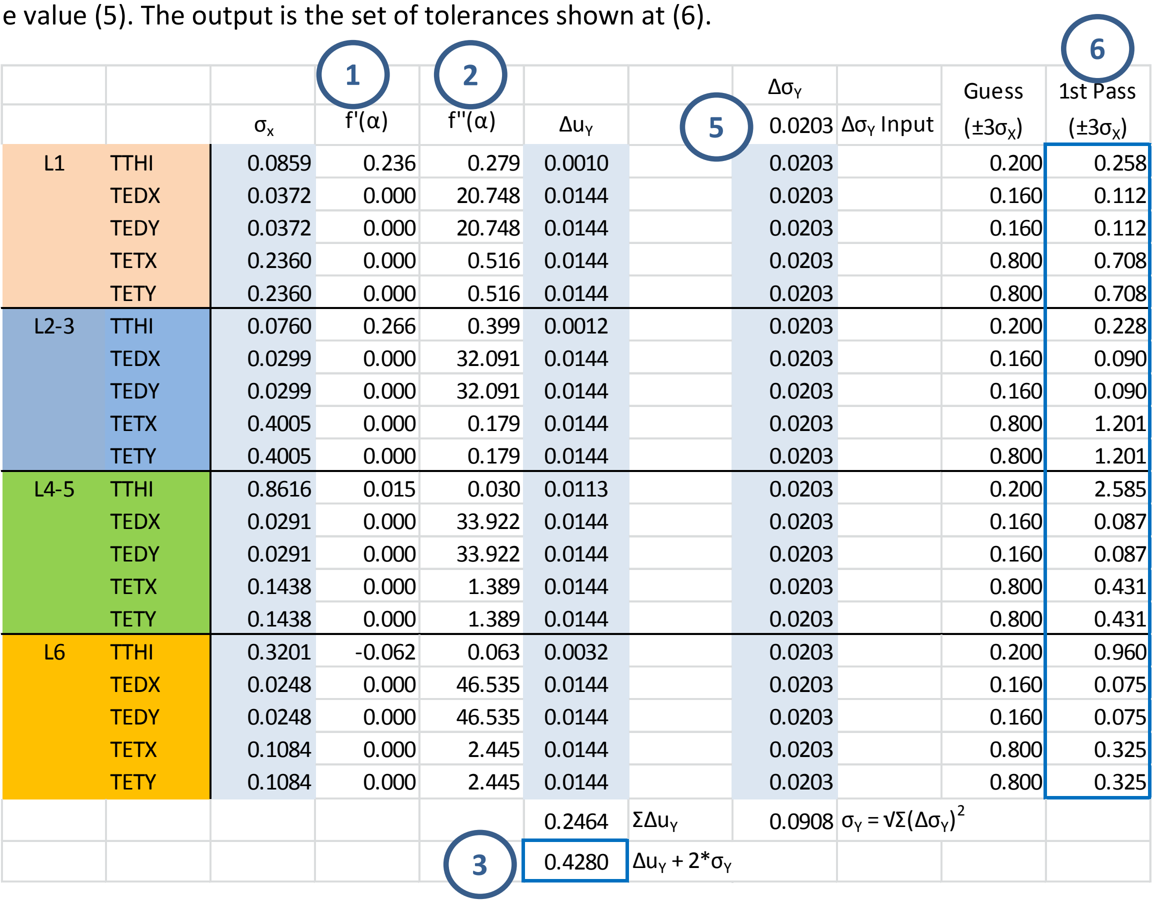
$$\Delta\mu_Y + 2\cancel{\sigma}_Y = (0.355)\left(\Delta\tilde{\mu}_Y + 2\tilde{\cancel{\sigma}}_Y\right) = 0.428 \tag{1.12}$$

Any one of these four statistics could be used in the attempt to optimize the input tolerances. It was found that $\Delta\mu_Y + 2\cancel{\sigma}_Y$ requires the least number of iteration cycles to converge on a solution.

5.4.2 Spreadsheet setup

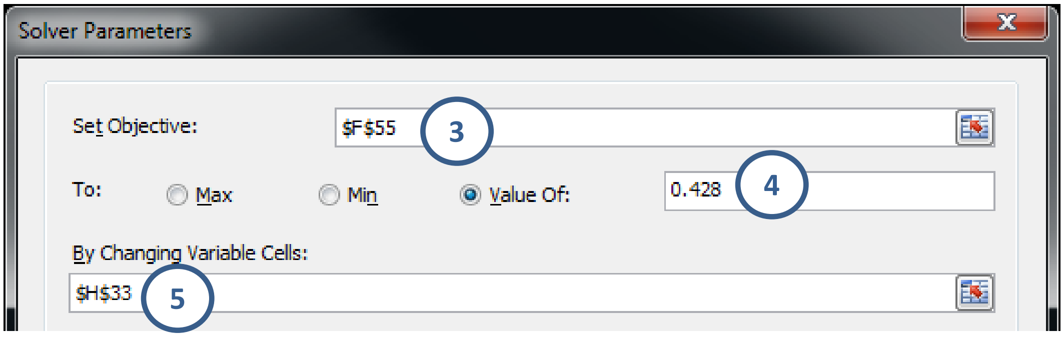
Optimizing the individual input tolerances so they yield the value chosen for $\Delta\mu_Y + 2\cancel{\sigma}_Y$ is done using Microsoft Excel (see Figure 5.3). Setup begins with listing all the input variables along with their respective coefficients from the Jacobian (1) and the main diagonal of the Hessian matrix (2) (cross-terms are excluded). The relationship between σ_X, $\Delta\mu_Y$, and $\Delta\sigma_Y$ is setup according to equations derived in section 2.

Excel has an optimization utility called Solver (see Figure 5.4) which is how the tolerances are found that meet the value calculated in Eq.(1.12). The Solver adjusts the value at (5) until the result shown at (3) matches the value manually input at (4). All the individual contributions of σ_X on $\Delta\sigma_Y$ are set equal to the same value (5). The output is the set of tolerances shown at (6).



							$\Delta\sigma_Y$		Guess	1st Pass
		σ_x	f'(α)	f''(α)	Δu_Y		0.0203	$\Delta\sigma_Y$ Input	$(\pm3\sigma_X)$	$(\pm3\sigma_X)$
L1	TTHI	0.0859	0.236	0.279	0.0010		0.0203		0.200	0.258
	TEDX	0.0372	0.000	20.748	0.0144		0.0203		0.160	0.112
	TEDY	0.0372	0.000	20.748	0.0144		0.0203		0.160	0.112
	TETX	0.2360	0.000	0.516	0.0144		0.0203		0.800	0.708
	TETY	0.2360	0.000	0.516	0.0144		0.0203		0.800	0.708
L2-3	TTHI	0.0760	0.266	0.399	0.0012		0.0203		0.200	0.228
	TEDX	0.0299	0.000	32.091	0.0144		0.0203		0.160	0.090
	TEDY	0.0299	0.000	32.091	0.0144		0.0203		0.160	0.090
	TETX	0.4005	0.000	0.179	0.0144		0.0203		0.800	1.201
	TETY	0.4005	0.000	0.179	0.0144		0.0203		0.800	1.201
L4-5	TTHI	0.8616	0.015	0.030	0.0113		0.0203		0.200	2.585
	TEDX	0.0291	0.000	33.922	0.0144		0.0203		0.160	0.087
	TEDY	0.0291	0.000	33.922	0.0144		0.0203		0.160	0.087
	TETX	0.1438	0.000	1.389	0.0144		0.0203		0.800	0.431
	TETY	0.1438	0.000	1.389	0.0144		0.0203		0.800	0.431
L6	TTHI	0.3201	-0.062	0.063	0.0032		0.0203		0.200	0.960
	TEDX	0.0248	0.000	46.535	0.0144		0.0203		0.160	0.075
	TEDY	0.0248	0.000	46.535	0.0144		0.0203		0.160	0.075
	TETX	0.1084	0.000	2.445	0.0144		0.0203		0.800	0.325
	TETY	0.1084	0.000	2.445	0.0144		0.0203		0.800	0.325
					0.2464	$\Sigma\Delta u_Y$	0.0908	$\sigma_Y = \sqrt{\Sigma(\Delta\sigma_Y)^2}$		
					0.4280	$\Delta u_Y + 2{*}\sigma_Y$				

Figure 5.3 Excel spreadsheet configuration for optimizing tolerances to a specific value of $\Delta\mu+2\cancel{\sigma}$



Set Objective: F55

To: ○ Max ○ Min ● Value Of: 0.428

By Changing Variable Cells:

H33

Figure 5.4 Example screenshot showing the Solver dialog box

5.4.3 Iteration Process

The process for arriving at a set of tolerances which produce the desired yield is an iterative one. An iteration cycle consists of using the MatLab code to estimate $\Delta\mu_Y$, σ_Y and $\cancel{\sigma}_Y$ from the previous iteration's tolerance set. Next, these estimates are put into Eq.(1.11) to find a new value for $\Delta\mu_Y + 2\cancel{\sigma}_Y$. The solver in Excel then calculates input tolerances which have an equal weighting on the merit function and meet the new value for $\Delta\mu_Y + 2\cancel{\sigma}_Y$. This set of tolerances then serve as the basis for making the next estimate of $\Delta\mu$, σ and $\cancel{\sigma}$ on the next iteration cycle. The process continues until the change in $\Delta\mu_Y + 2\cancel{\sigma}_Y$ is insignificant.

Table 5.4 has the results, step-by-step of the iteration process. At iteration cycle 3, the tolerances on z-axis shift and tilts were set to maximum values of 0.1 and 1.0 respectively. At iteration cycle 5 and on, sensitivity coefficients were recalculated per the schedule in Table 5.5. Figure 5.5 shows the rate at which $\Delta\mu_Y + 2\cancel{\sigma}_Y$ converged to a stable value.

Table 5.4 Change in tolerance values for each iteration cycle

Iteration	L1			L2/3			L4/5			L6		
	TTHI	TEDX/Y	TETX/Y	TTHI	TEDX/Y	TETX/Y	TTHI	TEDX/Y	TETX/Y	TTHI	TEDX/Y	TETX/Y
0	0.200	0.160	0.800	0.200	0.160	0.800	0.200	0.160	0.800	0.200	0.160	0.800
1	0.258	0.112	0.708	0.228	0.090	1.201	2.585	0.087	0.431	0.960	0.075	0.325
2	0.255	0.111	0.705	0.226	0.089	1.196	2.570	0.087	0.429	0.952	0.074	0.324
3	0.100	0.118	0.751	0.100	0.095	1.000	0.100	0.093	0.458	0.100	0.079	0.345
4	0.100	0.116	0.737	0.100	0.093	1.000	0.100	0.091	0.449	0.100	0.078	0.338
5	0.100	0.116	0.737	0.100	0.093	1.000	0.100	0.091	0.449	0.100	0.078	0.338
6	0.100	0.107	0.702	0.100	0.082	1.000	0.100	0.079	0.390	0.100	0.065	0.270
7	0.100	0.107	0.702	0.100	0.082	1.000	0.100	0.079	0.390	0.100	0.065	0.270
8	0.100	0.105	0.694	0.100	0.080	1.000	0.100	0.078	0.382	0.100	0.063	0.261
9	0.100	0.105	0.694	0.100	0.080	1.000	0.100	0.078	0.382	0.100	0.063	0.261
Final	0.100	0.104	0.692	0.100	0.080	1.000	0.100	0.077	0.380	0.100	0.063	0.259

**Gray boxes denotes manually fixed tolerances*

Table 5.5 Change in the various optimization parameters for each iteration cycle

Iteration					Update to	
	Δμ est	σ est	~~σ~~ est	Δμ+2~~σ~~	Tolerances	Coefficients
0	0.654	0.447	0.275	0.428	X	
1	0.247	0.155	0.091	0.423	X	
2	0.244	0.153	0.090	0.423	X	
3*	0.247	0.162	0.089	0.409	X	
4	0.238	0.156	0.086	0.409	X	
5	0.284	0.214	0.103	0.379		X
6	0.220	0.164	0.079	0.379	X	
7	0.226	0.174	0.081	0.372		X
8	0.217	0.167	0.078	0.373	X	
9	0.218	0.168	0.078	0.371		X

**Fixed tolerance limit for select variables*

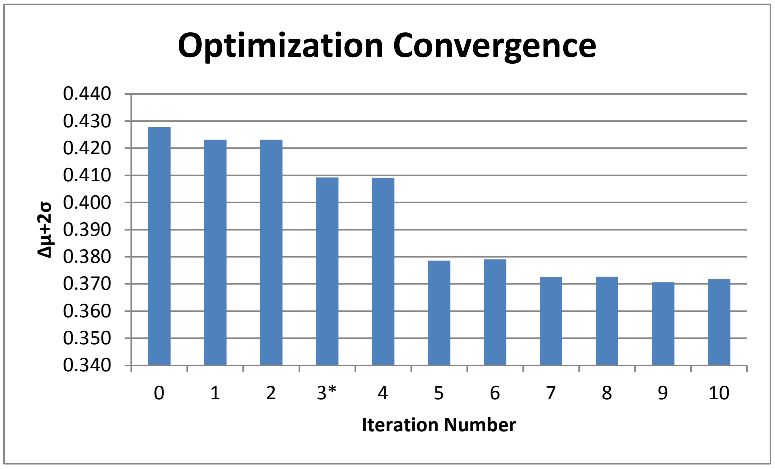


Figure 5.5 Change in Δμ+2~~σ~~ for each iteration cycle

5.4.4 Resulting Tolerances

The output of the iteration process is the set of tolerances shown in the following table. Z-axis tolerances were fixed because of the 0.2 mm nominal air-gap between the front and rear sets of lens elements. Tilts were limited to no more than 1 deg. Because of our starting assumptions, these are all 2σ values of mean-centered, normally distributed variables. Practical implementation depends on the manufacturing process and the application of geometric dimensioning and tolerancing.

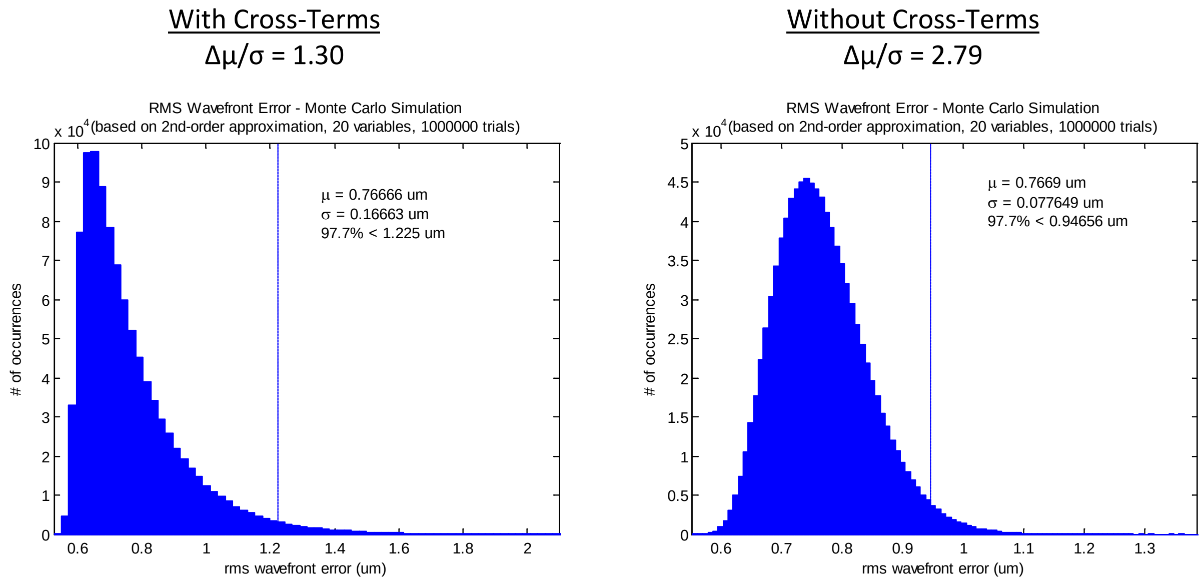
Table 5.6 Tolerances optimized to yield an rms wavefront error of 97.7% at 1.1 μm

Lens/Lens Group	Displacement, z-axis mm (inch)	Decenter in x & y-axis mm (inch)	Tilt about x & y-axis degrees
L1	± 0.100 (± .0039)	±0.104 (± .0041)	±0.692
L2/3	± 0.100 (± .0039)	±0.080 (± .0031)	±1.000
L4/5	± 0.100 (± .0039)	±0.077 (± .0030)	±0.380
L6	± 0.100 (± .0039)	±0.063 (± .0024)	±0.259

6 Comparison of Monte Carlo simulation results

6.1 Taylor approximation versus estimates

A Monte Carlo simulation of the second-order Taylor polynomial is run using the final sensitivity coefficient and tolerance values. The input variables are normally distributed with a standard deviation of 1/3 the tolerance limit. So that the simulation statistics would best represent that of the true distribution, 1,000,000 trials were run. The results are displayed in the next two histograms. The system was simulated with and without the second-order cross-derivatives (cross-terms). The results compared with the estimates are presented in Table 6.1.



With Cross-Terms

$\Delta\mu/\sigma = 1.30$

Without Cross-Terms

$\Delta\mu/\sigma = 2.79$

Figure 6.1 Comparison between Monte Carlo simulations of the merit value distribution when cross-terms are included (left image) and excluded (right image)

Table 6.1 Comparison of simulation results and estimated statistics for the rms wavefront error

	with cross-terms		w/o cross-terms	
	Estimate	Simulation	Estimate	Simulation
Mean (μm)	0.7669	0.7667	0.7669	0.7669
Std. Dev. (μm)	0.1667	0.1666	0.0776	0.0776
97.7% Yield Value (μm)	1.100*	1.225	0.922	0.947

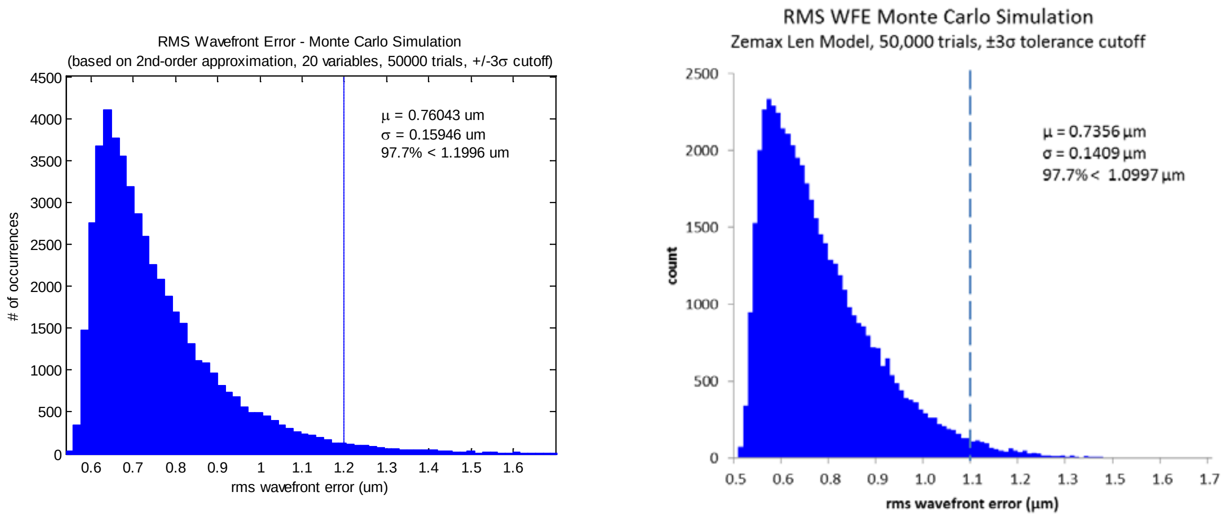
**target goal*

While the estimates for the mean and standard deviation are accurate, the yield is greater than the desired goal of 2 λrms (1.10 μm rms at 0.546 μm). The ratio Δμ/σ for the distribution with and without cross-terms is 1.30 and 2.79 respectively. This means our original assumption of a somewhat normally distributed output was off especially for the distribution with cross-terms. It stands to reason that our estimate for the yield percentage would also be off.

6.2 Approximation verses Lens model

With the same set of tolerances for calculating the sensitivity coefficients from the lens model, a Monte Carlo simulation is next run in the optical design software. All the input distributions are normal with a ±3σ cutoff. The software default is ±2σ. To override the default, a special tolerance operand is added to the tolerance data editor called STAT. The tolerance dialog box is configured so that ray aiming is on and the same merit function for collecting the coefficients is used. The focus is also optimized with each run using damped least squares.

The resulting histogram of the lens model rms wavefront error is shown in Figure 6.2 along with a similar MatLab simulation of the Taylor polynomial. Both histograms show the results of 50, 000 runs. Zemax took 6 hours to process the simulation while MatLab took about 2 seconds. The MatLab simulation was set up with the same ±3σ tolerance cutoff on the input variables.


RMS Wavefront Error - Monte Carlo Simulation
(based on 2nd-order approximation, 20 variables, 50000 trials, +/-3σ cutoff)

RMS WFE Monte Carlo Simulation
Zemax Len Model, 50,000 trials, ±3σ tolerance cutoff

Figure 6.2 Comparison of Monte Carlo simulation results of the rms wavefront error by the Taylor polynomial approximation (left image) and lens model (right image)

The Zemax simulation results have a mean value of 0.736 μm and std. dev. of 0.141 μm for the rms wavefront error compared with 0.760 μm and 0.159 μm for the MatLab results. The expected value which yields 97.7% is 1.200 μm for the MatLab simulation. This is still over the goal of 2 λrms at 0.546 μm. The Zemax results in a 97.7% value of 1.100 which is right on target. While it is nice to have met the goal, closer agreement between the two simulations would have been more desirable.

The author has observed a trend where the statistics of the Taylor polynomial tend to overestimate the statics of the lens model and believes the source of this error is the ability of a quadratic polynomial to accurately approximate the merit function's output. In this report, coefficients were evaluated at each input variable's tolerance limit. There may be a more optimum choice which gives better results.

7 Conclusion

This report has taken the reader through a tolerancing example that complements the various papers on the subject and hopefully added to his or her understanding of the topic. It has shown the process very accurately estimates the statistics of the second-order Taylor approximation. It also very closely predicts the statistics of the lens model. With this capability, comes the potential for predicting product yields.

There are some important things to keep in mind about the assumptions used in the analysis. First, it was performed on input variables that are normally distributed. Alternate distributions can be used but the estimates for the mean and standard deviations become cumbersome if the distribution is not mean centered (such as with a Rayleigh type). The performance metric under investigation needs to be continuous over the tolerance range and the min and max tolerance values must be equal and symmetric about the nominal value. This is because of assumptions made in using the quadratic approximation to determine the first and second-order derivatives. Lastly, the ability to predict yield (or the probability of success) is based on the figure of merit having a normal distribution. Ideally this occurs when variables are independent and identically distributed. Where possible, tolerances should be chosen so they have roughly the same impact on the merit function.

In the course of the analysis, there were several important observations. It was learned that the ratio $\Delta\mu/\sigma$ is a good indicator of how closely the merit value approximates a normal distribution. The more variables included in the analysis, the better the yield prediction will be as long as the input variables are orthogonal. The degree to which they are orthogonal will limit how many variables are needed for the output to resemble a normal distribution. Orthogonally between variables can be determined by the presence of second-order cross-derivatives (also referred herein as cross-terms). Also noted is that the sensitivity coefficients change slightly based on where they are evaluated. While a quadratic curve is a fair approximation, it is not exact. The author believes this is the main reason for the difference in statistics between the Taylor approximation and Zemax lens model.

There are many ways to arrive at a wrong answer. It is easy to make a mistake in the setup and not know it or have an incorrect assumption. Good practice is to have a secondary method to check results. Monte Carlo simulations are an excellent way to do this. Using MatLab, it was a quick process to gather statistics on very large number of trial runs (1,000,000 runs in minutes). Always try to verify answers whenever possible.

While the scope of this analysis was limited to effects of lens placement on rms wavefront error, the process has broad applications. It can be applied to any similar system of input variables such as the thickness, index of refraction, and radius of curvature of lens elements or another performance metric could have been chosen. The utility of the delta method, were a complex function is approximated as a Taylor polynomial, cannot be overstated. Elements of this report can apply to the statistical tolerancing of many systems, optics related or otherwise.

Finally, it should be noted that careful assignment of tolerances alone is not enough to guarantee a successful system. There must also be solid manufacturing and quality assurance in place. Having an accurate tolerance analysis is an advantage, but it is only one of many steps that bring a product from concept to reality.

7.1 Potential areas of future work

- *Look at methods to improve the collection of sensitivity coefficients* - The sensitivity coefficients in this report showed variability to the selection of Δx, the point at which they were evaluated. There may be an optimum ratio of Δx to the tolerance value. This report set Δx equal to the tolerance value under investigation. Understanding this relationship better would improve accuracy between the Taylor approximation and lens model.

- *Automate the analysis optimization routine* – In this report, convergence of the tolerances and sensitivity coefficients were iterated on manually. A MatLab routine could be written to automate this process. Also, there may be a more efficient optimization method that exists or could be developed for this application.

- *Examine alternate input variable distribution and increase model fidelity* - Zemax is limited as to what it can model in regards to the mechanical interface and the types of random variable distributions it can support. In these respects MatLab is much more versatile. With the capability to extract accurate sensitivity coefficients from the lens model, various scenarios can be run in MatLab without Zemax.

- *Perform configuration trade studies* – The techniques presented herein can be useful in performing trade studies to evaluate the merits of design alternatives or look at various dimensioning and tolerancing scenarios.

- *Update the MatLab to Zemax communication protocol* - MathWorks, the software publisher for MatLab, no longer actively supports the DDE protocol which is used by Zemax to communicate with third party software. The effects can be seen in the 64-bit version of MatLab as it will occasionally fail to initiate the DDE connection. There is a risk with each new release that MatLab will no longer include native DDE functionality and the scripts used in this report will no longer work as they are written.

8 References

ASME Y14.5M-1994. *Dimensioning and Tolerancing*. New York: The American Society of Mechanical Engineers, 1994.

Burge, James H. "Error Analysis." *Course notes from OPTI 510L, College of Optical Science, Univ. of Arizona*. n.d.

Casella, George and Roger L. Berger. *Statistical Inference, 2nd Ed.* Duxbury, 2002.

Ginsberg, Robert H. "Outline Of Tolerancing (From Performance Specification To Toleranced Drawings)." *Opt. Eng. 20(2)* (1981): 202175.

Griffith, Derek. *How to Talk to Zemax from MATLAB*. 3 January 2006. Webpage. 2014. <https://www.zemax.com/support/knowledgebase/how-to-talk-to-zemax-from-matlab>.

Jonas, Reginald P. and Michael D. Thorpe. "Double Gauss lens design: a review of some classics." *Proc. SPIE 6342, International Optical Design Conference* (2006): 634202.

Koch, Donald G. "A Statistical Approach to Lens Tolerancing." *Proc. SPIE 0147, Computer-Aided Optical Design* (1978): 71-82.

Leica Camera AG. "Leica SUMMICRON-M 50 mm f/2 Technical Data Sheet." n.d.

Mandler, Walter, Garry Edwards and Erich Wagner. Four-Member Gauss Objective. United States: Patent 4,123,144. 31 Oct. 1978.

Parks, Robert E. "Optical component specifications." *Proc. SPIE 0237, International Lens Design Conference* (1980): 455-463.

Radiant Zemax LLC. *Zemax 12 Optical Design Program User's Manual*. 2012.

Smith, Warren J. "Fundamentals Of Establishing An Optical Tolerance Budget." *Proc. SPIE 0531, Geometrical Optics* (1985): 196.

Youngworth, Richard N. and Bryan D. Stone. "Cost-based tolerancing of optical systems." *Appl. Opt. 39* (2000): 4501-4512.

Youngworth, Richard N. "Statistical truths of tolerancing assignment in optical design." *Proc. SPIE 8131, Optical System Alignment, Tolerancing, and Verification V* (2011): 81310E.

Appendix A: Probability Theory and Tolerancing

This appendix examines the basics of probability theory and how to it applies to tolerancing. Probability theory attempts to predict how often a particular outcome of a random event occurs. Its application is useful because all properties of a system or part will have a random value over some range. The following specific topics are discussed: types of random variables, functions of random variables, probability functions, and the properties of mean and variance. Also looked at are how random variables combine, the central limit theorem, Monty Carlo analysis, and approximations using Taylor polynomials.

A tolerance places a limit on a property's accepted value. Anything outside the accepted value is outside of its tolerance. A property can be anything measurable such as mass, length, index of refraction, wavefront error, etc. It can be found either through a direct measurement or indirectly by calculating its value from the measurement of other properties. For example, mass and volume can be found by direct measurement, but density is calculated by diving mass by volume. Tolerances placed on mass and volume will also affect the tolerance on density.

When performing a tolerance analysis, two different methods can be used: worst case and statistical analysis. The first, referred to as worst case, selects those values that fall in the extreme range of their tolerance zones and will result in the largest tolerance range of the calculated value. As a result, 100% of all possible outcomes will fall within this tolerance range. For example, consider five blocks each with a height of 1.000±.010 inch which are arraigned in a stack one on top of another. The total nominal height of the stack is 5.000 inch and at worst case will always be within ±.050 inch of this value.

The second method is called statistical analysis. It assumes that a property's actual value will be selected at random as opposed to worst case where only the extreme values are chosen. This type of analysis can be more difficult because additional assumptions must be made, but can result in more relaxed tolerances with acceptable system yields.

A.1 Random Variables and Their Properties

The value a property takes on is assumed to occur in a random fashion and that its probably of occurrence can be described by a probability function. The length of a part's feature, index of refraction, density, etc. can be represented by a random variable *X* with a probability function *P*. The probability function represents the likelihood that the random variable *X* takes on any particular value *x* within its range. *P* has the property that it is always ≥ 0 and that all probabilities within the range of *X* add to 1.

Random variables come it two types. They can be either discrete or continuous. Almost all random variables dealt with here are continuous meaning that they can take on any value within the given range. The alternative type is a discrete random variable which can only take on specific values such as the integers 1 through 6 for a single die. Even though discrete random variables are rarely dealt with, they are still useful as a means for bounding a problem so they are included in this discussion.

Besides knowing the type of random variable, there are two other things needed to completely describe it. The first thing is the range of values that *X* can assume, which can be either finite or infinite, and the second is a description of its probability function.

There are a couple of ways to describe the probability function: either with a cumulative distribution function (cdf), probability mass function (pmf), or probability distribution function (pdf). The cdf is defined as

$$F(x) = P(X \le x), \qquad \text{for all } x. \tag{A.1}$$

Note that the cdf is adding up the probabilities as *x* increases. In the limit that *x*→-∞, *F*(*x*)=0 while in the limit that *x*→+∞, *F*(*x*)=1. If *X* is discrete, then the pmf is used which is given as

$$f(x) = P(X = x), \qquad \text{for all } x. \tag{A.2}$$

If *X* is continuous, the pdf is the function *f*(*x*) that satisfies

$$F(x) = \int_{-\infty}^{x} f(\alpha)d\alpha, \quad \text{for all } x. \tag{A.3}$$

Here the pdf is defined in terms of the cdf because *P*(*X*=*x*) = 0 for a continuous random variable.

A.1.1 Expected Value and Variance

Two important properties derived from a random variable's distribution are its expected value and variance. Along with the type of distribution, knowing these two properties can describe the random variable's probability function most of the time. A complete description requires the distribution's moment generating function. The expected value and variance are the first and second moments respectively derived from the moment generating function. For the distributions consider here, all moments greater than the second are zero.

A random variable's expected value is self-explanatory: it's the value the random variable is expected to take. It is the average value of *x* where each *x* is weighted by its probability of occurrence. Where *g*(*X*) is a function of the random *X*, the expected valued of *g*(*X*) is defined as

$$\mathrm{E}[g(X)] = \begin{cases} \int_{-\infty}^{\infty} g(x) f(x) dx & \text{if } x \text{ is continuous} \\ \sum_{x \in \chi} g(x) P(X = x) & \text{if } x \text{ is discrete.} \end{cases} \tag{A.4}$$

For the specific case where *g*(*X*)=*X*, then the term mean is defined as *μ*=E[*X*].

Suppose there is a discrete random variable *X* with *x* = 1, 2, 3, …,*n* where the probability of occurrence is the same for each value of *x* such that *P*(*X*=*x*) = 1/*n*. The expected value of *X* is then

$$\begin{aligned}\mathrm{E}[X] = \mu &= x_1\left(1/n\right) + x_2\left(1/n\right) + \cdots + x_n\left(1/n\right) \\ &= \frac{x_1 + x_2 + \cdots + x_n}{n}.\end{aligned} \tag{A.5}$$

The variance of a random variable is a little less intuitive. It describes the scale of the random variable's probability distribution. The variance of *g*(*X*) is defined in terms of the expected value as

$$\mathrm{var}[g(X)] = \mathrm{E}[g(X) - \mathrm{E}[g(X)]]^2. \tag{A.6}$$

Again, for the case when *g*(*X*)=*X*, the variance of *X* is represented as σ^2 such that

$$\mathrm{var}\left[X\right] = \sigma^2 = \mathrm{E}[X - \mu]^2. \tag{A.7}$$

The standard deviation, *σ* of *X* is the square root of its variance. Using the same example as in Eq.(A.5), the variance is calculated as

$$\begin{aligned}\mathrm{var}\left[X\right] = \sigma^2 &= \left(x_1 - \mu\right)^2\left(1/n\right) + \left(x_2 - \mu\right)^2\left(1/n\right) + \cdots + \left(x_n - \mu\right)^2\left(1/n\right) \\ &= \frac{\left(x_1 - \mu\right)^2 + \left(x_2 - \mu\right)^2 + \cdots + \left(x_n - \mu\right)^2}{n}.\end{aligned} \tag{A.8}$$

By factoring out the mean of *X* in Eq.(A.8), it can be rewritten so that

$$\sigma^2 = \frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n} - \mu^2. \tag{A.9}$$

Noting that

$$\mathrm{E}[X^2] = \frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n}, \tag{A.10}$$

the variance of *X* can be express as

$$\mathrm{var}[X] = \mathrm{E}[X^2] - \mu^2. \tag{A.11}$$

Eq.(A.11) is not only valid for our specific example, but in general for any random variable *X*. The square root of Eq.(A.10) is the RMS value of *X*. Rewriting Eq.(A.11) results in

$$X_{RMS}^2 = \mu^2 + \sigma^2. \tag{A.12}$$

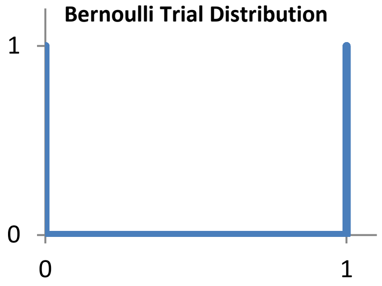
If the mean of *X* is zero, then the RMS value of *X* is equal to its standard deviation.

A.1.2 Useful Probability Distributions for Tolerancing

Listed here are some common distributions which can be used to describe variations. Most have some sort of property that can be measured. For example, the uniform distribution has limits which describe the distribution if they are known. As will be shown in the next section, these distributions can be modified to fit a particular application.

A.1.2.1 Bernoulli Trial

A Bernoulli trial is the simplest distribution listed here. It applies to discrete random variables that have only one of two possible outcomes: either success or failure. The probability of success is given by the parameter *p* which is usually set to *p*=1/2. It is the most conservative assumption to make when the true distribution isn't known. The only assumption more conservative is worst case. The difference is that with Bernoulli trials the parts are assumed to be selected at random. It is the underlying assumed distribution when tolerance values are root-sum-squared (rss). Also, the outcome of multiple Bernoulli Trials forms a binominal distribution.



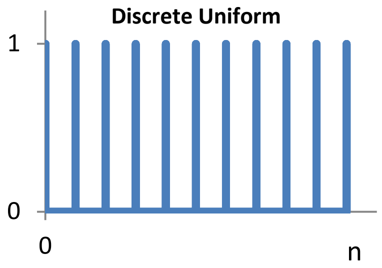
pmf: $P(X=x\,|\,\tfrac{1}{2})=\tfrac{1}{2}^{x}\left(\tfrac{1}{2}\right)^{1-x};\qquad x=0,1$

mean: $\dfrac{1}{2}$

variance: $\dfrac{1}{4}$

A.1.2.2 Discrete Uniform

A discrete uniform distribution is used when a random variable takes on discrete values each with an equal probability. Rolling a six-sided die would be an example of such a distribution where n=6.



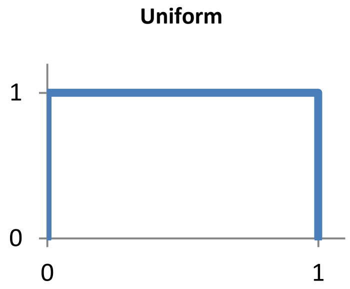
pmf: $P\left(X=x\,|\,n\right)=\dfrac{1}{n};\qquad x=1,2,\ldots,n$

mean: $\dfrac{n+1}{2}$

variance: $\left({}^{1}\!/\!_{12}\right)\left(n+1\right)\left(n-1\right)$

A.1.2.3 Uniform

The uniform distribution is similar to the discrete version, except the random variable can take on any value from zero to one with equal probability. This distribution is a good conservative choice to apply to parameters whose true distribution isn't known.



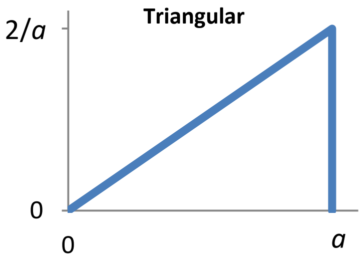
pdf: $f(x) = 1; \quad 0 \le x \le 1$

mean: $\frac{1}{2}$

variance: $\frac{1}{12}$

A.1.2.4 Triangular

The triangular distribution has a special application in radially symmetric systems. It represents the distribution of position vector lengths for points that are uniformly distributed over a circular area. Table C-1 shows a histogram of such a two-dimensional distribution. The limit *a* represents the radius of this circular area.



pdf: $f(x;a) = \frac{2}{a^2} x; \quad 0 \le x \le a, 0 \le a$

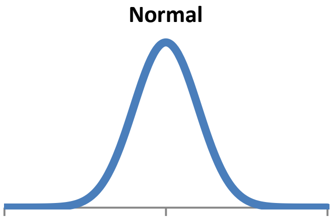
mean: $\frac{2}{3} a$

variance: $\frac{1}{18} a$

RMS: $\frac{1}{\sqrt{2}} a$

A.1.2.5 Normal

The normal distribution is the most commonly encountered distribution for any particular parameter. It is also referred to as a Gaussian distribution. It is the only distribution that when added to itself still has the same form and is integral to the Central Limit Theorem. A normal distribution with a mean of zero and standard deviation of one is referred to as a standard normal distribution, n(0,1).



pdf: $f(x;\mu,\sigma)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right);\quad -\infty\leq x\leq+\infty$

mean: μ

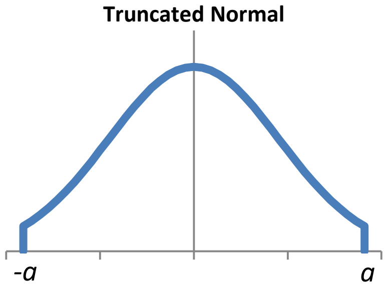
variance: σ^2

A.1.2.6 Truncated Normal

The truncated normal is a normal distribution with the tails removed. This is the expected distribution for variables that have a normal distribution and then undergo 100% inspection. Any value not within $\pm a$ is rejected. The distribution on which the truncated normal pdf is based must be normalized by

$$\operatorname{erf}\left(\frac{a}{\sqrt{2\sigma^2}}\right)=\frac{2}{\sqrt{\pi}}\int_0^{a/\sqrt{2\sigma^2}}t^2dt \tag{A.13}$$

so its cumulative probability remains equal to one. Eq.(A.13) is referred to as the error function. Its purpose is to calculate the cumulative probability that a random variable with a standard normal distribution falls between $\pm a$.



pdf: $f(x;\mu,\theta,a)=\frac{\exp\left(-x^2/2\sigma_0^2\right)}{\sqrt{2\pi\sigma_0^2}\cdot\operatorname{erf}\left(a/\sqrt{2\sigma^2}\right)};\quad -a\leq x\leq a$

mean: 0

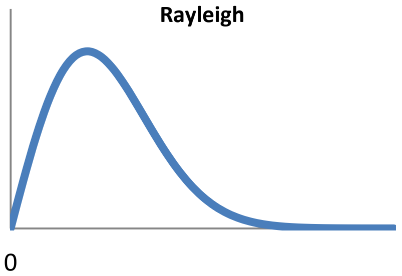
variance: $\sigma_0^2\left(1-2a\cdot f(a)\right)$

A.1.2.7 Rayleigh

The Rayleigh distribution like the triangular distribution is also significant in radially symmetric systems. It represents the distribution of position vector lengths for points that have a Gaussian distribution over a circular area. In polar coordinates such a function is describe by

$$f(r,\theta)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-r^2}{2\sigma^2}\right). \tag{A.14}$$

Polar coordinates are used because it makes analysis of radially symmetric optical systems easier. The Rayleigh distribution can also be truncated like the normal distribution to represent variables that have passed inspection.



pdf: $f(r;\sigma)=\dfrac{r}{\sigma^2}\exp\left(\dfrac{-r^2}{2\sigma^2}\right); \quad 0\le r\le+\infty$

mean: $\sigma\sqrt{\pi/2}$

variance: $\frac{1}{2}(4-\pi)\sigma^2$

RMS: $\sigma\sqrt{2}$

A.2 Functions of a single Random Variable

If the mean and variance of a random variable *X* is known, then it's possible to calculate the mean and variance of the function *Y* = *g*(*X*). This is useful because the known distributions presented in the previous section can be adapted without having to derive the equations for *Y* from scratch.

For a generic function where *Y* = *aX*+*b*,

$$\mathrm{E}[Y]=\mu_Y=a\mu_X+b \tag{A.15}$$

and

$$\mathrm{var}[Y]=\sigma_Y^2=a^2\sigma_X^2\,. \tag{A.16}$$

Suppose a block has a height of 10±0.1 mm. Its maximum value is 10.1 mm while its minimum value is 9.9mm. The height *H* is a random variable which can take on any value between its max and min limits and can therefore be expressed as a function of another random variable. If the height is uniformly distributed between its limits, then *a* = 0.2, *b* = 9.9 and *H* = 0.2*X* + 9.9 where 0≤*x*≤1. Referring to section A.1.2.3 on uniform distributions, the mean is 1/2 and variance 1/12. Applying Eq.(A.15) and Eq.(A.16), the mean and variance for *H* is 0.2(1/2)+9.9 = 10 mm and 0.22(1/12) = .0033 mm^2 respectively. The standard deviation is $0.2/2\sqrt{3}=0.057$ mm.

If instead *H* is discrete being either 9.9mm or 10.1 mm, it would have a Bernoulli distribution where the mean is 1/2 and variance is 1/4. In which case, the mean and variance of *H* are 10 mm and 0.01 mm^2 respectively with a standard deviation of 0.1 mm. Notice in this case that the standard deviation is equal to half the total tolerance range.

A.3 Combining Multiple Random Variables

Knowing how multiple random variables combine forms the basis of statistical tolerance analysis. If a property is dependent on the value of several other properties, knowing the mean and variance of each contributing variable, allows the mean and variance to be calculated for the dependent property.

The first case is where *Y* = X_1+X_2+…+X_n. Calculating the mean is straightforward. It is simply

$$\begin{aligned}\mu_Y &= \mathrm{E}[X_1 + X_2 + \cdots + X_n] \\ &= \mathrm{E}[X_1] + \mathrm{E}[X_2] + \cdots + \mathrm{E}[X_n] \\ &= \mu_{X_1} + \mu_{X_2} + \cdots + \mu_{X_n}.\end{aligned} \tag{A.17}$$

The variance of *Y* is a little more difficult to calculate. Starting with the case where there are only two random variables such that *Y* = *X₁*+*X₂*, the variance of *Y* is calculated as

$$\begin{aligned}\sigma_Y^2 &= \mathrm{var}[X_1 + X_2] \\ &= \mathrm{E}[\left(X_1 + X_2\right) - \mathrm{E}[X_1 + X_2]]^2 \\ &= \mathrm{E}[X_1 + X_2 - \mu_{X_1} - \mu_{X_2}]^2 \\ &= \mathrm{E}[\left(X_1 - \mu_{X_1}\right) + \left(X_2 - \mu_{X_2}\right)]^2 \\ &= \mathrm{E}[\left(X_1 - \mu_{X_1}\right)^2 + 2\left(X_1 - \mu_{X_1}\right)\left(X_2 - \mu_{X_2}\right) - \left(X_2 - \mu_{X_2}\right)^2] \\ &= \mathrm{E}[X_1 - \mu_{X_1}]^2 + \mathrm{E}[X_2 - \mu_{X_2}]^2 + 2\,\mathrm{E}[\left(X_1 - \mu_{X_1}\right)\left(X_2 - \mu_{X_2}\right)] \\ &= \sigma_{X_1}^2 + \sigma_{X_2}^2 + 2\,\mathrm{cov}[X_1, X_2].\end{aligned} \tag{A.18}$$

The last term in Eq.(A.18) is the covariance of X_1 and X_2 and is defined as

$$\mathrm{cov}[X_1, X_2] = E[\left(X_1 + \mu_{X_1}\right)\left(X_2 + \mu_{X_2}\right)]. \tag{A.19}$$

The derivation for calculating variance of more than two variables makes the derivation of Eq.(A.18) look simple by comparison. The assumptions discussed next make the process easier.

A.3.1 Independence

If X_1 and X_2 are independent random variables, then

$$\mathrm{E}[X_1 X_2] = \mathrm{E}[X_1]\mathrm{E}[X_2] \tag{A.20}$$

and the covariance of Eq.(A.18) becomes zero. Now, calculating variance is much easier no matter how many random variables are involved. In general when $Y = X_1+X_2+\ldots+X_n$ and $X_1,X_2,\ldots,X_n$ are independent random variables, then

$$\sigma_Y^2 = \sigma_{X_1}^2 + \sigma_{X_2}^2 + \cdots + \sigma_{X_n}^2. \tag{A.21}$$

Returning to the example of the 10±0.1 mm high block, the standard deviation was found to be 0.057 mm and 0.1 mm for a continuous uniform and discrete Bernoulli distribution respectively. For a stack of *n* blocks, the standard deviation on the height of the stack is

$$\begin{aligned}\sigma_{total} &= \sqrt{a^2\sigma_{X_1}^2 + a^2\sigma_{X_2}^2 + \cdots + a^2\sigma_{X_n}^2} \\ &= a\sigma \cdot \sqrt{n}\end{aligned} \tag{A.22}$$

where *a* is the total tolerance range of 0.2 mm for an individual block and σ_X is the standard deviation of the type of distribution we are applying to the block's height.

Applying the two distributions to Eq.(A.22) results in

$$\sigma_{total} = \begin{cases} \frac{a}{2} \cdot \sqrt{n} & \text{for Bernoulli distribution} \\ \frac{a}{2\sqrt{3}} \cdot \sqrt{n} & \text{for continuous uniform.} \end{cases} \tag{A.23}$$

If there are 9 blocks, the standard deviation is 0.3 mm using the Bernoulli distribution and 0.17 mm for the continuous uniform case.

When *Y* is the product *n* independent random variables, Eq.(A.20) can be expanded upon as

$$\mu_Y = \prod_{i=0}^{n} \mu_i. \tag{A.24}$$

For two independent random variables such that $Y = X_1X_2$, the derivation to calculate variance is

$$\begin{aligned}\sigma_Y^2 &= E[X_1X_2 - E[X_1X_2]]^2 \\ &= E[X_1X_2 - \mu_{X_1}\mu_{X_2}]^2 \\ &= E[X_1^2X_2^2 - 2X_1X_2\mu_{X_1}\mu_{X_2} + \mu_{X_1}^2\mu_{X_2}^2] \\ &= X_{1_{RMS}}^2 X_{2_{RMS}}^2 - 2\mu_{X_1}^2\mu_{X_2}^2 + \mu_{X_1}^2\mu_{X_2}^2 \\ &= X_{1_{RMS}}^2 X_{2_{RMS}}^2 - \mu_{X_1}^2\mu_{X_2}^2.\end{aligned} \tag{A.25}$$

Eq.(A.25) can be expanded upon so that in general, for the product of *n* independent random variables

$$\sigma_Y^2 = \prod_{i=1}^{n} X_{i_{RMS}}^2 - \prod_{i=1}^{n} \mu_{X_i}^2. \tag{A.26}$$

A.3.2 Estimating Variance

Eq.(A.26) can be cumbersome to use. It requires that the RMS values be calculated. There are a couple of ways to estimate the variance of a product. First, it can be approximated without having to calculate the RMS values as

$$\sigma_Y^2 \approx \sum_{i=1}^{n}\left(\prod_{j=1\neq i}^{n} \mu_{X_i}^{n}\right)\cdot \sigma_{X_i}^2 . \tag{A.27}$$

The product of variances in this case have been removed because they are small enough to be ignored.

The second method involves finding partial derivatives (Burge). In addition to estimating the variance of a product, it works on any function that is smooth and continuous. Given a function of multiple independent random variables such that $y = f(x_1,x_2, \ldots,x_n)$, the variance can be approximated as

$$\sigma_Y^2 \approx \sigma_{X_1}^2 \left|\frac{\partial f}{\partial x_1}\right|^2 + \sigma_{X_2}^2 \left|\frac{\partial f}{\partial x_2}\right|^2 + \cdots + \sigma_{X_n}^2 \left|\frac{\partial f}{\partial x_n}\right|^2 . \tag{A.28}$$

The assumption with Eq.(A.28) is that that the function behaves linearly around x_i and each of the variables is independent.

As an example, suppose that a rectangular flat plate is 12 in. x 12 in. x .10 in. and each dimension has a uniformly distributed tolerance of $\Delta t = \pm.01$ inch. What is the variance of the volume? The equation for volume is $f(w,l,h) = w \times l \times h$. The partial derivatives are

$$\frac{\partial f}{\partial w} = l\times h, \quad \frac{\partial f}{\partial l} = w\times h, \quad \frac{\partial f}{\partial h} = w\times l. \tag{A.29}$$

With each dimension having a variance in the form of $(1/12)(2\Delta t)2 = (1/3)\Delta t^2$, the result is

$$\begin{aligned}\sigma_V^2 &\approx \sigma_w^2\cdot(l\times h)^2 + \sigma_l^2\cdot(w\times h)^2 + \sigma_h^2\cdot(w\times l)^2 \\ &\approx \left(\frac{.01^2}{3}\right)(12\times.10)^2 + \left(\frac{.01^2}{3}\right)(12\times.10)^2 + \left(\frac{.01^2}{3}\right)(12\times12)^2 \\ &\approx .7\ \text{in}^6.\end{aligned} \tag{A.30}$$

The standard deviation is the square root of the variance, $\sigma = .83$ in^3. This is the same result as if Eq.(A.27) were used. The exact value is found with Eq.(A.26) as

$$\begin{aligned}\sigma_V^2 &= \left(w^2+\sigma_w^2\right)\left(l^2+\sigma_l^2\right)\left(h^2+\sigma_h^2\right) - w^2\times l^2\times h^2 \\ &= \left[12^2+\left(\frac{.01^2}{3}\right)\right]\left[12^2+\left(\frac{.01^2}{3}\right)\right]\left[.10^2+\left(\frac{.01^2}{3}\right)\right] - 12^2\times12^2\times.10^2 \\ &= .6913\,\text{in}^6.\end{aligned} \tag{A.31}$$

While the distribution for each dimension is known for the flat plate example, what is not known is the type of distribution on the volume. It will be different than that for the linear dimensions. If for example given two dice, the distribution on each would be a discrete uniform with $n = 6$, $\mu = 3.5$, and $\sigma^2 = 2.9$. The sum of the two dice would have a mean of 7, and a variance of 5.8 with the distribution shown in Figure A-1 which while still discrete is definitely not uniform.

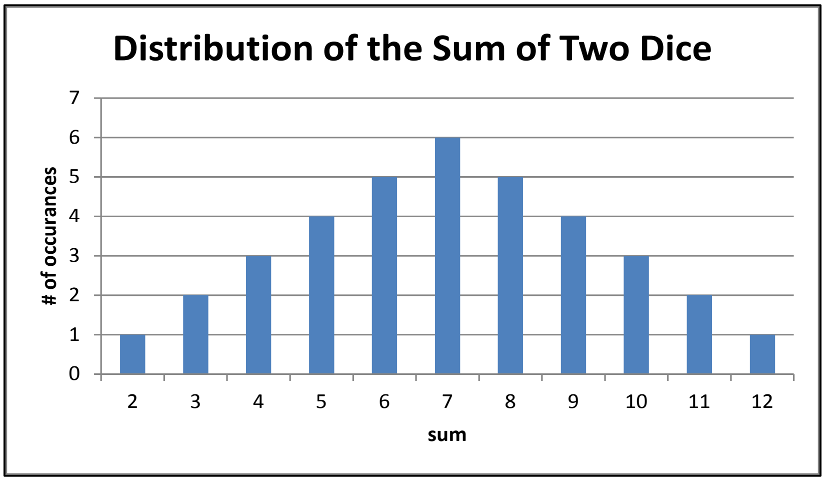


Figure A-1 Distribution of the sum of two dice

Variance and standard deviation by themselves do not mean much if the type of probability distribution to which they apply is not known. Ultimately, the goal is to calculate something useful like a probability of success or yield. To do that, requires some knowledge about how the resulting random variable is distributed.

A.3.3 Estimating Probabilities using the Central Limit Theorem

Building on the dice example from the previous section, look at what happens to the distribution of the mean as the number of dice added together increases. The number *n* of dice thrown forms a random sample from the die's distribution. Each die is identically distributed with the same mean *μ* and standard deviation *σ*. The sample mean is defined as

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i . \qquad \text{(A.32)}$$

In Figure A-2, the distribution is plotted for the cases where *n* = 1,2,3,4. As *n* increases, the mean starts to take on the appearance of a normal (Gaussian) distribution. This turns out to be true for the sum of any set of identically distributed independent random variables regardless of their distribution.

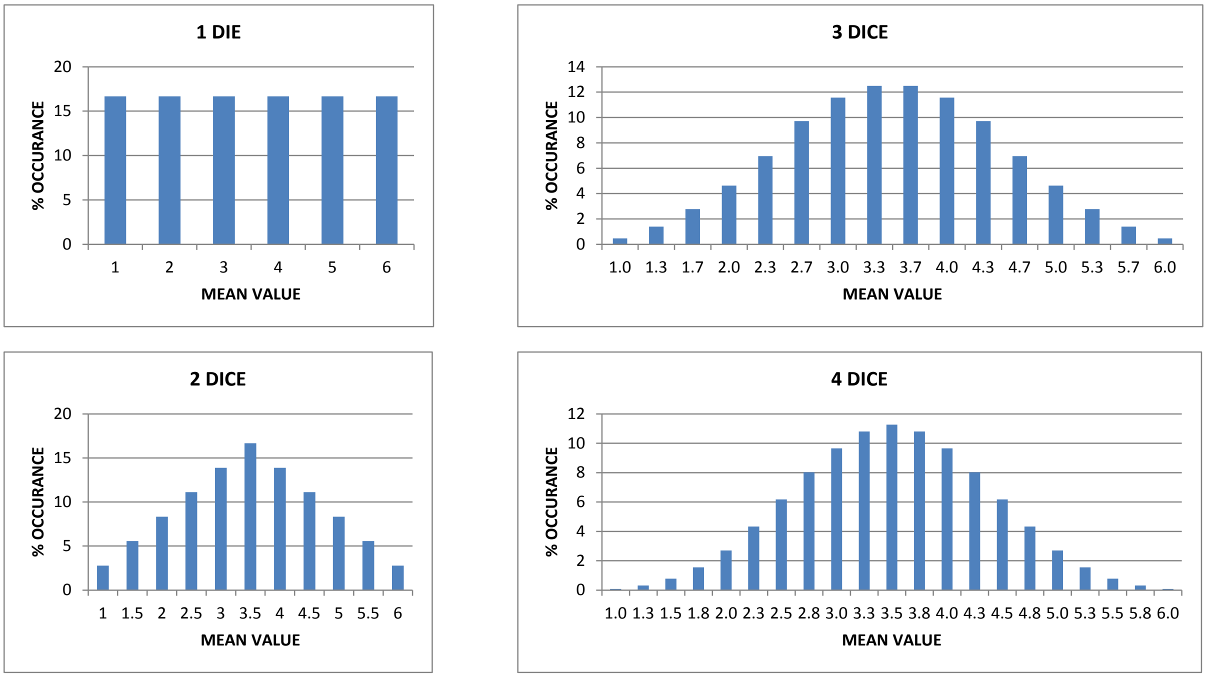


Figure A-2 Change in the mean value distribution with an increasing number of dice

This concept of the sample mean having a normal distribution for large values of *n* is called the Central Limit Theorem (CLT). More specifically, where $\sqrt{n}\cdot\left(\bar{X}-\mu\right)/\sigma$ is the normalized sample mean *Z* with a cdf of *F*(*x*), it states that

$$\lim_{n\to\infty} F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-y/2} dy. \qquad \text{(A.33)}$$

Recall from Eq.(A.3) that the pdf is the function inside the integral. In Eq.(A.33), that function is the standard normal distribution

$$\mathrm{n}(0,1) = \frac{1}{\sqrt{2\pi}} e^{-x/2}. \qquad \text{(A.34)}$$

Another way of thinking about the CLT is that as *n* gets large, $\bar{X}$ approaches μ, and ever smaller fluctuations about μ occur that are normally distributed.

In reality, it doesn't take too many random samples before the CLT becomes useful for making approximations. Table A-1 shows how quickly the 1σ confidence interval probability (the probability that the mean of *X* will be within $\pm\sigma$ of μ) of the dice example coincides with that of the normal distribution. The difference is <1% for three or more dice.

Table A-1 Comparison between a discrete uniform distribution and a normal distribution (for N=6)

N	σ	$P\left(-1\sigma < \bar{X} < 1\sigma\right)$	Difference from n(0,1)*
1	1.71	66.67%	1.6%
2	1.21	66.67%	1.6%
3	0.99	67.59%	0.68%
4	0.85	68.21%	0.06%

Using the normal distribution, it's possible to estimate the probability that the mean will fall within a particular confidence interval (see Table A-2).

Table A-2 Confidence intervals for a normal distribution

Confidence Interval	Probability of Occurrence
1σ	68.27%
2σ	95.45%
3σ	99.73%
4σ	99.99%

The CLT applies to all distributions with a finite variance. Consider a series of Bernoulli trials. A binominal distribution counts the number of successful Bernoulli trials with 1 being a success and 0 a failure. Figure A-3 shows the distribution for 1, 2, 8, and 32 trials. Again, as *n* increases, the distribution takes on a normal appearance.

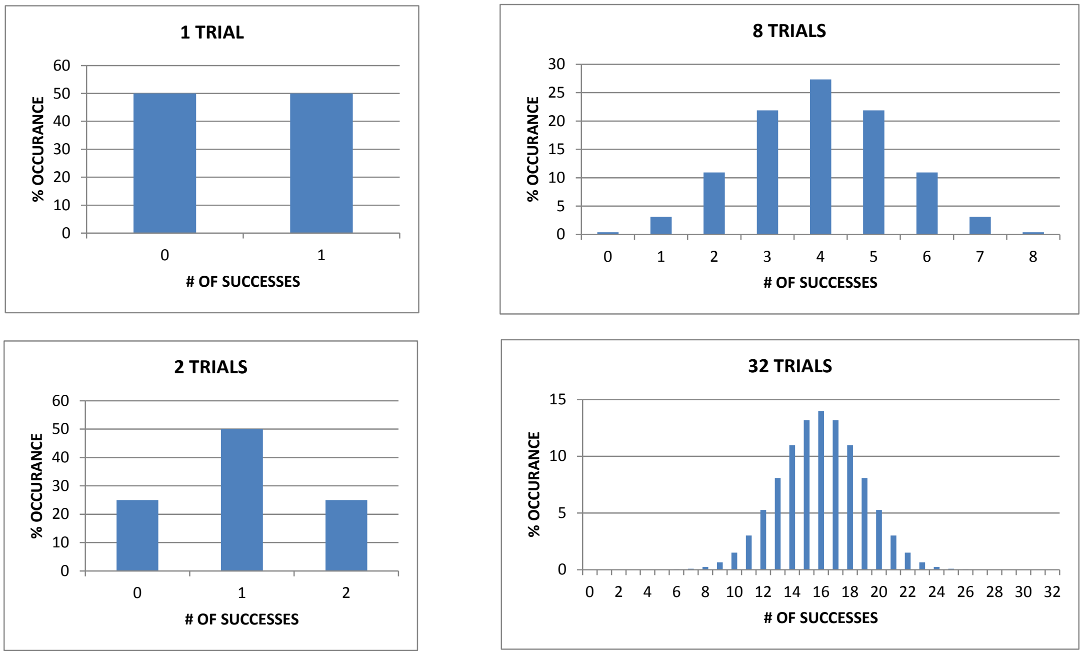


Figure A-3 Binominal distribution for various number of trials

Estimating the probability of the 1σ confidence interval for a binominal distribution is not as straightforward as in the previous dice example. This is because the probability does not correspond well to the 68.27% of a normal distribution. Figure A-4 shows how the probability that the number of successful trials which fall within $\pm\sigma$ of the mean varies with the number of total trials. Because of this variability, other methods must be used to determine the 1σ confidence interval probability.

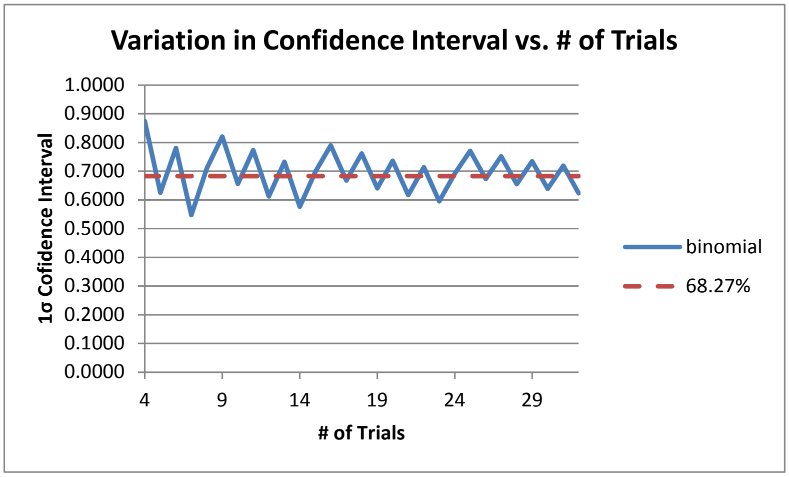


Figure A-4 1σ confidence interval of a binominal distribution as a function of *n* trials

The first method is to calculate the probability explicitly.

$$P\left(-1\sigma \le X \le +1\sigma\right) = P\left(X \le +1\sigma\right) - P\left(X < -1\sigma\right) \tag{A.35}$$

$$P\left(X \le x\right) = \sum_{i=0}^{\lfloor x \rfloor} \left(\frac{n!}{i!\left(n-i\right)!}\right) \cdot p^{i}\left(1-p\right)^{n-i}. \tag{A.36}$$

The floor function is represented by $\lfloor x \rfloor$. It is the largest integer less than or equal *x*. The probability of any one Bernoulli trial being a success is given by *p* which up to this point has been set at ½.

The second method is to approximate the 1σ confidence interval probability using a normal distribution. This is possible because of the CLT. As was shown earlier, using the standard deviation alone does not work. The normal distribution must be messaged a bit. First set μ = *np* and σ = (*np*(1-*p*))^{½} then

$$\begin{aligned} P\left(-1\sigma \le X \le +1\sigma\right) &= P\left(X \le \lfloor +1\sigma \rfloor\right) - P\left(X \le \lceil -1\sigma \rceil - 1\right) \\ &\approx P\left(Y \le \lfloor +1\sigma \rfloor + \tfrac{1}{2}\right) - P\left(Y \le \lceil -1\sigma \rceil - \tfrac{1}{2}\right) \\ &\approx P\left(Z \le \frac{\lfloor +1\sigma \rfloor + \tfrac{1}{2} - \mu}{\sigma}\right) - P\left(Z \le \frac{\lceil -1\sigma \rceil - \tfrac{1}{2} - \mu}{\sigma}\right). \end{aligned} \tag{A.37}$$

The ceiling function is represented as $\lceil x \rceil$. It is the smallest integer that is greater than or equal to *x*. The mean and standard deviation of the binominal distribution are *μ* and σ respectively, *Y* has a n(*μ*,*σ*) distribution, and *Z* a n(0,1) distribution. Eq.(A.37) estimates the 1*σ* confidence interval probability using a lookup table for the standard normal distribution. The factor of ½ added to the standard deviation is called the continuity correction. Table A-3 gives the 1*σ* confidence interval probabilities for several values of *n*. The explicit method is compared to the normal approximation. As can be seen, the approximation is very close.

Table A-3 1σ confidence interval probability for a binominal distribution with *n* trials (p = 1/2)

N	*σ*	P(-1*σ* ≤ *X* ≤ +1*σ*)	Normal approx. w/correction
1	0.50	100%	n/a
2	0.71	50%	n/a
3	0.87	75.00%	75.17%
4	1.00	87.50%	86.64%
5	1.12	62.50%	62.89%
6	1.22	78.13%	77.93%
7	1.32	54.69%	55.03%
8	1.41	71.09%	71.11%
9	1.50	82.03%	81.76%

Going back to the example of a stack of 9 blocks where each block had a Bernoulli distributed tolerance of either 9.9 mm or 10.1 mm, the standard deviation of the stack height using Eq.(A.23) was 0.3 mm. The next step is to find the 1*σ* confidence interval probability for the total height. For a binominal distribution with *n* = 9 and *p* = ½, *μ* = 4.5 and *σ* = 1.5. Using Eq.(A.35) and Eq.(A.36), the 1*σ* confidence interval is 4.5±1.5 and P(3 $\leq$ *X* $\leq$ 6) = 82.03% compared against 81.76% if the normal distribution approximation from Eq.(3.18) is used. This is the probability that the height of the 9 blocks stacked together will fall within the 1*σ* confidence interval of 9±0.3 mm.

In conclusion, there are several important things to remember when using the CFT to estimate a distribution. It is only valid for distributions with a finite variance. The variables must be independent and should be identically distributed. They do not have to be identically distributed to come up with an estimate, but large differences should be avoided.

A.3.4 Estimating Distributions using Monte Carlo Analysis

Instead of estimating the distribution as in the previous section, another method is to find it by running a Monty Carlo simulation. This is a program ran on a computer that generates a very large number of random samples. The user defines the population distribution the samples are drawn from and the function for the output variable. The more random samples, the closer the approximation will be to the output variable's true distribution. The method to calculate the minimum number of random samples necessary to achieve a certain level of confidence is covered in Casella & Berger. The simulations rely on raw computational power as the number of trials can be on the order of 10^6 or 10^7. Unlike using the CLT to estimate distribution, a Monty Carlo simulation does not require that variables be identically distributed.

A.3.4.1 Generating Random Variables

To produce a random sample from a desired population distribution, the Monty Carlo algorithm uses the population's cdf. The random number generator returns a value from zero to one. This corresponds to the probability axis (y-axis) value. The shape of the cdf determines the resulting x-value. Figure A-5 shows two examples. If the random number generator returns the number 0.8, then the discrete uniform cdf (shown on the left) returns a value of 5 while the normal cdf (shown on the right) returns 0.45.

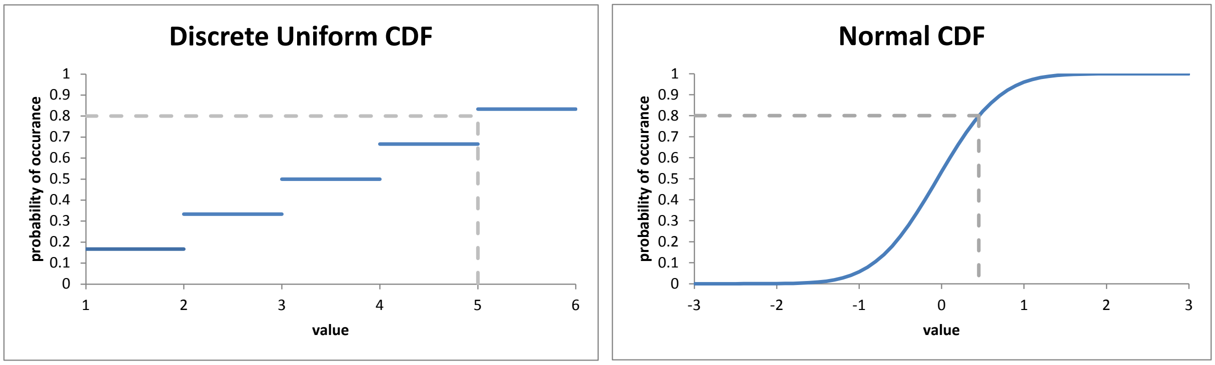


Figure A-5 Example cdf's used to generate a random number

STANDARD NORMAL RANDOM NUMBER GENERATOR

To use the concept described in the previous section requires that the cdf be inverted. For any given cdf such that *y* = *F*(*x*), it is necessary to solve for *x* in terms of *y*. The steps necessary to invert a normal cdf (shown in the right hand image of Figure A-5) are shown next as an example. The cdf is defined as

$$F(x) = y = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{x} e^{-(\alpha-\mu)^2/2\sigma^2} d\alpha. \quad \text{(A.38)}$$

Using the error function erf(x) defined in Eq.(A.13), Eq.(A.38) can be rewritten as

$$y = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}}\right)\right]. \quad \text{(A.39)}$$

Solving for *x* results in

$$x = \sqrt{2\sigma^2} \cdot \mathrm{erf}^{-1}\left(2y-1\right) + \mu; \quad 0 \le y \le 1. \tag{A.40}$$

Shown next is how to generate random number with a standard normal distribution using Eq.(A.40) with MatLab. The results are shown in Figure A-6. Notice that the algorithm requires fewer lines than what is needed to plot the results.

```
%Random Number Generator
%Standard Normal Distribution
y = rand(1,100000);
x = sqrt(2).*erfinv(2.*y-1); %0<y<1

%Results
figure;
hist(x,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
P=sum((x <= 1)&(x >= -1))/100000;
gtext(['P(-1\sigma<x<+1\sigma)= ',num2str(P)]);
```

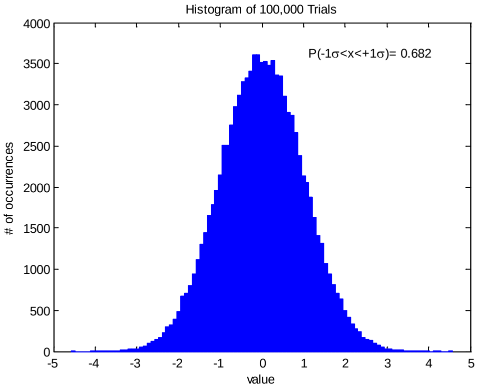


Figure A-6 MatLab script and results for generating random numbers with a standard normal distribution

TRUNCATED STANDARD NORMAL RANDOM NUMBER GENERATOR

It is a simple task to modify Eq.(A.40) to produce random numbers with a truncated standard normal distribution. Adding the error function to normalize by the cumulative probability that the variable is between ±a results in

$$x = \sqrt{2\sigma^2} \cdot \mathrm{erf}^{-1}\left(\mathrm{erf}\left(\frac{a}{\sqrt{2\sigma^2}} \right) \cdot \left(2y-1\right) \right) + \mu; \quad 0 \le y \le 1. \tag{A.41}$$

A small change to the above MatLab code produces the truncated normal distribution shown in Figure A-7.

```
%Random Number Generator
%Truncated Standard Normal Distribution
a = 2;
a = erf(a/sqrt(2));
y = rand(1,100000);
x = sqrt(2).*erfinv(a.*(2.*y-1)); %0<y<1

%Results
figure;
hist(x,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
```

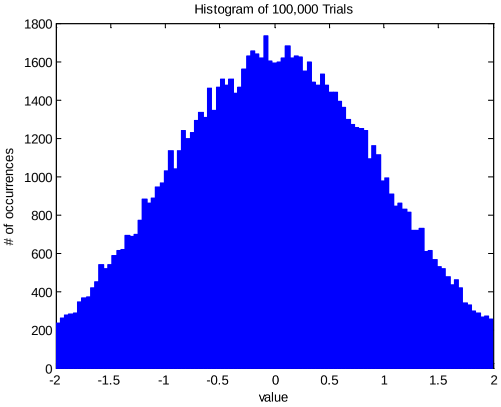


Figure A-7 MatLab script and results for generating random numbers with a truncated standard normal distribution

A.3.4.2 Simulation Examples

STACK OF NINE BLOCKS

The simulation for the stack of nine blocks was performed using the code shown in Figure A-8. The resulting distribution is also shown. In section A.3.3, the one sigma confidence interval was calculated explicitly as 82.03% and estimated using the CLT to be 81.76%. Monty Carlo analysis using 1,000,000 trials puts it at 82.026% in agreement with the explicit calculation.

The sixth line of code rounds the random value to the nearest tenth. It is necessary because small differences in the floating point decimal can make the answer inaccurate.

```
%Monty Carlo Simulation: Stack of 9 Blocks
a = .2;
b = 9.9;
N = 1000000;
y = rand(9,N);
x = sum(a.*round(y)+b); %0<y<1
x = round(10.*x)./10;

%Results
figure;
hist(x);
title('Histogram of 1,000,000 Trials')
xlabel('value');
ylabel('# of occurrences');
P=sum((x <= 90+.3)&(x >= 90-.3))/N;
gtext({['\sigma= ',num2str(std(x))],['P(-
1\sigma<x<+1\sigma)= ',num2str(P)]});
```

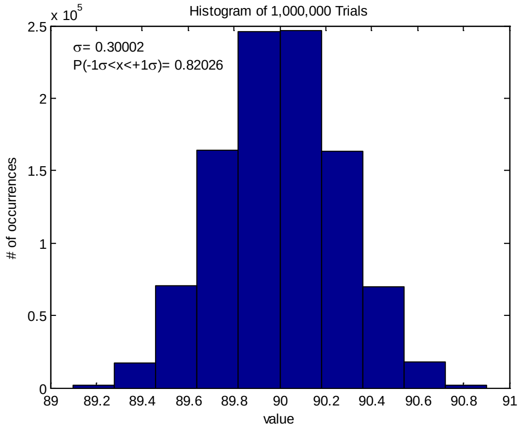


Figure A-8 Total height simulation for a stack of nine blocks

MEAN VALUE OF FOUR DICE

The code to analyze the mean value of four dice is fairly short. In could have been written using just one line instead of three as seen in Figure A-9. The resultant one sigma confidence interval is 68.199% based on 1,000,000 trials compared to an estimate of 68.27% using the CLT.

```
%Monty Carlo Simulation: Mean of 4 Dice
N = 1000000; %Number of trials
y = rand(4,N);
x = sum(ceil(6.*y))/4; %0<y<1

%Results
figure;
hist(x,21);
title('Histogram of 1,000,000 Trials')
xlabel('value');
ylabel('# of occurrences');
P=sum((x <= 3.5+.85)&(x >= 3.5-.85))/N;
gtext({['\sigma= ',num2str(std(x))],['P(-
1\sigma<x<+1\sigma)= ',num2str(P)]});
```

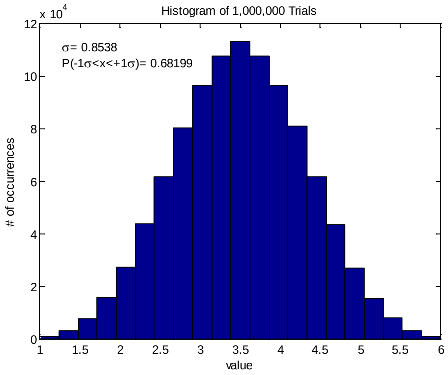


Figure A-9 Simulated mean of four dice

A.3.5 Simulating Non-Identically Distributed Random Variables

With Monty Carlo analyses, the ability to determine the standard deviation does not rely on having identically distributed random variables. This was the main limitation on CLT estimation. Figure A-10 shows the progression in total height distribution of the stack of nine blocks as the tolerance on the 9th block is increased. In the image on the left, all tolerances are equal at ±0.1 mm. The middle and right images show the effect on total height distribution of changing the tolerance on block 9 to ±0.5 mm and ±1 mm respectively.

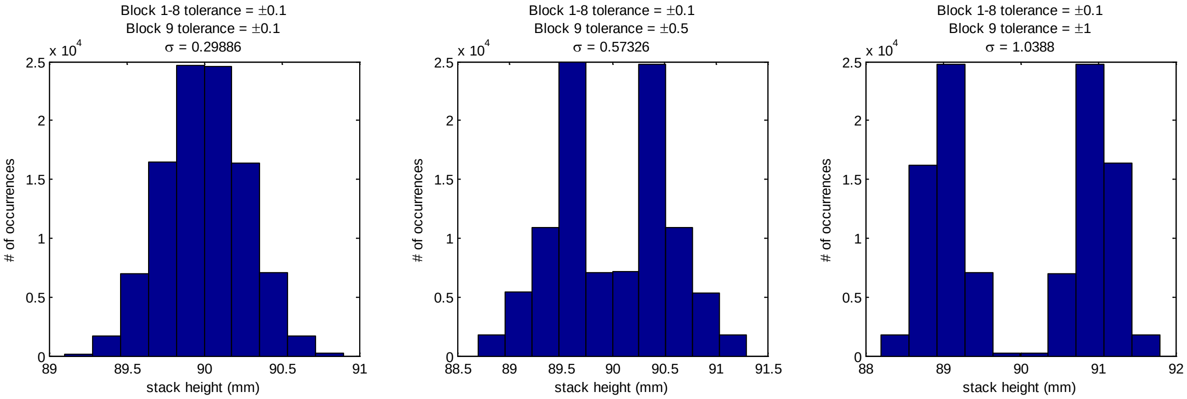


Figure A-10 Height distribution comparison when 1 of 9 blocks has a different tolerance range

Appendix B: Simulation Techniques

In this appendix, methods are shown for analyzing the effect that various geometric feature controls have on statistical distributions. All geometric dimensioning and tolerancing should be interpreted in accordance with ASME Y14.5M-1994.

B.1 Creation of Random Radial Positions Vectors

For purpose of analysis, displacement between parallel axes shall be represented by a position vector. The vector starts at the first axis which serves as the origin and terminates at the second. It also lies in a two-dimensional plane perpendicular to both axes and is therefore fully described using two independent variables.

B.1.1 Choice of Coordinate System

There are two coordinate systems available for representing the position vector: either Cartesian or polar. The variables x and y are used in the Cartesian coordinate system while r and θ are polar variables (Figure B-1). Consider the two-dimensional normal (Gaussian) distribution shown in Figure B-2 below. In the Cartesian system, the x and y-axis distributions are both normal. In polar coordinates, it exhibits radial symmetry as shown by the uniform angular distribution and has a Rayleigh radial distribution.

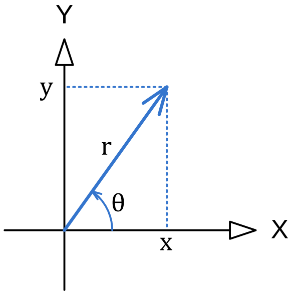


Figure B-1 Position vector in two dimensions

If possible, the choice of a coordinate system should be one that simplifies the analysis. Because of the uniform angular distribution in polar coordinates, the actual angle used in the analysis is trivial so long as it is selected at random. Polar coordinates are a good choice for radially symmetric systems. It can be difficult to model such systems with Cartesian coordinates for any distribution other than Gaussian.

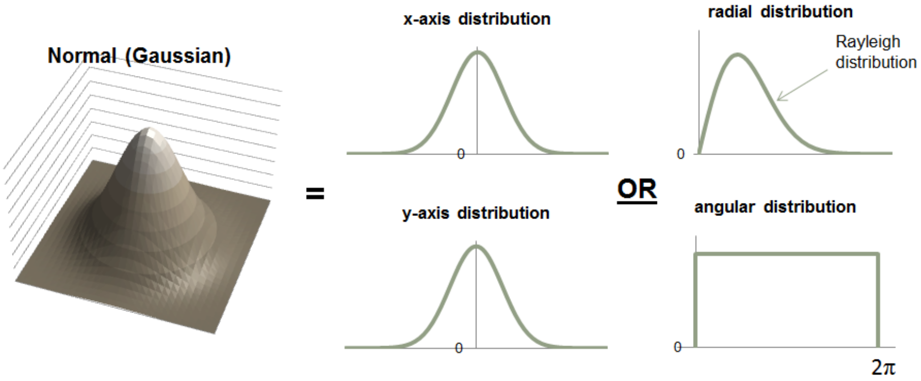
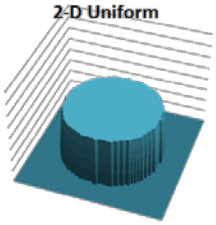
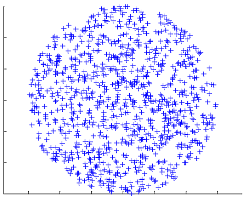
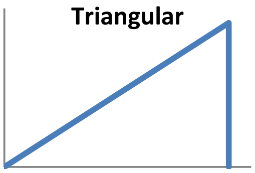
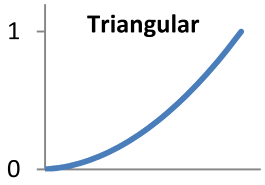
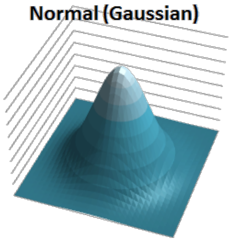
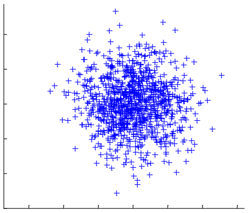
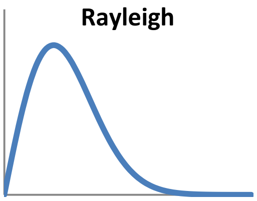
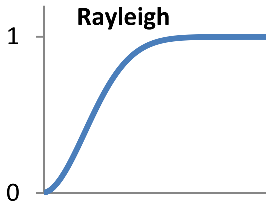
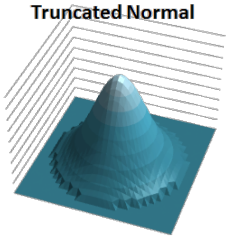
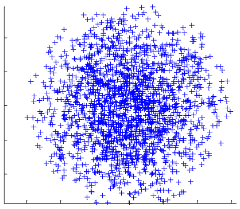
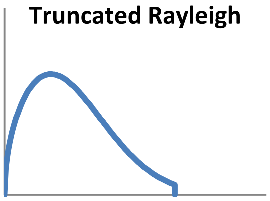
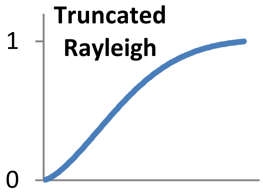


Figure B-2 One dimensional components of a two dimensional normal (Gaussian) distribution

B.1.2 Radial Distribution Types

There are three radial distribution types that are important for simulating a radially symmetric system. The first is the Rayleigh distribution as mentioned previously. The other two are the triangular and truncated Rayleigh. Table B-1 compares all three.

Table B-1 Useful two dimensional distributions

2-D Distribution	Scatter Plot of Random Points	Radial pdf	Radial cdf
			
			
			

The process for creating random numbers with a pdf described in the above table follows the procedure outlined in section A.3.4.1. The cdf is determined by integrating the pdf over the range zero to *r* where *r* is a variable. Next, the cdf is then solved for *r* creating a function that will generate a random variable with the prescribed pdf when the input is a uniform variable between zero and one. This expression is entered into MatLab where the function rand() serves as the uniform independent variable

For each of the three distributions in Table C-1, the cdf and generating functions are listed next. Also shown is how the expression gets implemented in MatLab along with a histogram of the generated random number data set.

B.1.2.1 Triangular

For generating a random variable whose probability distribution is triangular, Eq.(B.1) and Eq.(B.2) are the cdf and resulting generating function respectively.

$$F(r)=\int_0^r \frac{2\alpha}{R^2}d\alpha=\left(\frac{r}{R}\right)^2;\quad 0\le r\le R \tag{B.1}$$

$$r=R\sqrt{t};\quad 0\le t\le 1 \tag{B.2}$$

```
%Random Number Generator
%Triangular Distribution
r_limit = 1.5;
t = rand(1,100000);
r = r_limit*sqrt(t);

%Results
figure;
hist(r,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
```

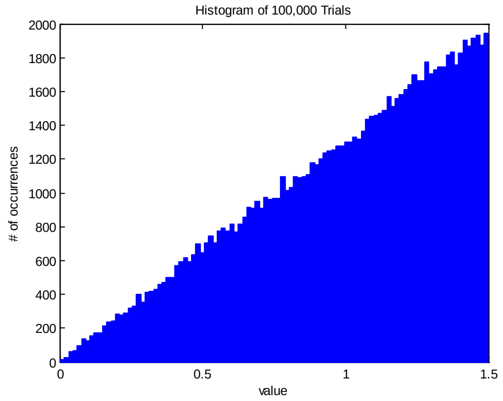


Figure B-3 Triangular distribution random number generator

B.1.2.2 Rayleigh

For generating a random variable whose probability distribution is a Rayleigh function, Eq.(B.3) and Eq.(B.4) are the cdf and resulting generating function respectively. Note that in Eq.(B.4) the expression 1-*t* can be replaced by just *t* which is how it is shown in the accompanying code.

$$F(r)=\int_0^r \frac{\alpha}{\sigma^2}\exp\left(\frac{-\alpha^2}{2\sigma^2}\right)d\alpha=1-\exp\left(-r^2/2\sigma^2\right);\quad 0\le r,\sigma\le\infty \tag{B.3}$$

$$r=\sigma\sqrt{-2\ln\left(1-t\right)};\quad 0<t\le 1 \tag{B.4}$$

```
%Random Number Generator
%Rayleigh Distribution
sigma = .5;
t = rand(1,100000);
r = sigma*sqrt(-2*log(t));

%Results
figure;
hist(r,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
```

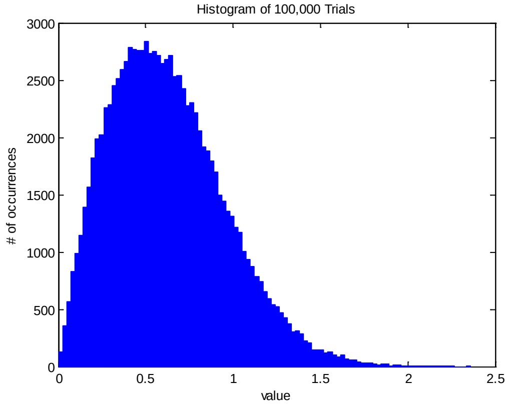


Figure B-4 Rayleigh distribution random number generator

B.1.2.3 Truncated Rayleigh

The Rayleigh distribution can be truncated the same way as the normal distribution (see section A.3.4.1). The variable *t* in Eq.(B.4) is scaled by the area defined by the truncating radius *a* which is calculated from Eq.(B.3). Solving for the generating function results in

$$r=\sigma\sqrt{-2\ln\left[t\left(1-\exp\left(-a^{2}/2\sigma^{2}\right)\right)\right]};\quad 0<t\leq 1, 0\leq a\leq\infty. \tag{B.5}$$

```
%Random Number Generator
%Truncated Rayleigh
sigma = 1;
R = 2.5;
t = rand(1,100000);
a = 1-exp(-R.^2/(2*sigma^2));
r = sigma*sqrt(-2*log(1-a.*t));

%Results
figure;
hist(r,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
```

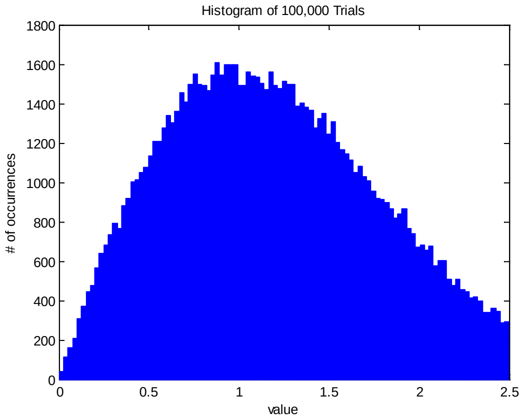


Figure B-5 Truncated Rayleigh distribution random number generator

B.1.3 Working with Modifiers

Adding a material modifier to the feature control frame makes the size of the tolerance zone dependent on the size of the feature. Consider the part shown in Figure B-6.

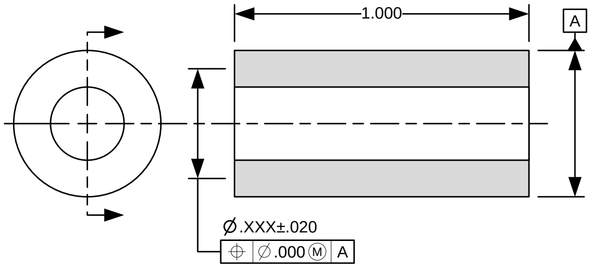


Figure B-6 Drawing application of coaxiality with a material condition modifier

The axis of the through hole is related to datum A with zero positional tolerance at the hole's maximum material condition (MMC). The difference in size between the least material condition (LMC) and MMC is .040 inches in diameter so at LMC the coaxiality tolerance is .040 inches.

The material modifier's effect is to make the radial distribution's limit a varying parameter instead of fixed. Because size is a measurement between opposing feature elements, the distribution on the limit is one-dimensional. Each random size chosen becomes the limit on the distribution from which the radial distance is chosen. If, for example, the size distribution is uniform and the underlying radial distribution is triangular the actual resulting radial is as shown in Figure B-7 along with the MatLab used to run the simulation.

```
%Random Number Generator
%Effect of Material Condition Modifier
N =100000; %number of trials
r_limit =2*rand(1,N); %uniform size dist.
r =r_limit.*sqrt(rand(1,N)); %radial value

%Results
figure;
hist(r,101);
title('Histogram of 100,000 Trials')
xlabel('value');
ylabel('# of occurrences');
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
```

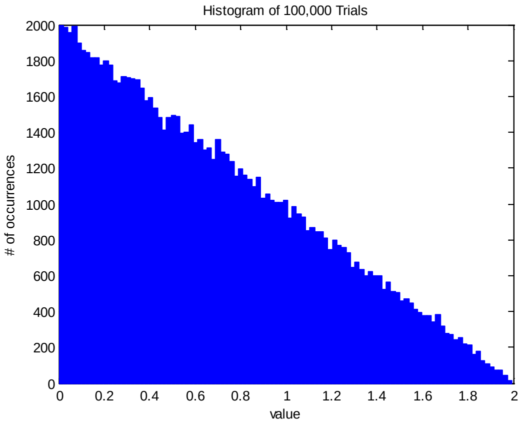


Figure B-7 Effect of material condition modifiers on radial distribution

The resulting triangular distribution is reversed from what has been presented so far. The probability is highest that the axis of the hole will be close to the datum axis. This may seem counterintuitive considering that the underlying distribution for any single limit value is triangular with a probability approaching zero at the origin. To understand why this is happening, the following series of histograms (Figure B-8) show the progression of the resulting distribution as the limit takes on an increasing number of discrete values. Each limit value has its own triangular distribution which overlaps the other triangular distributions. The most overlaps occur near the origin. When the number of discrete limit values is large enough so that it approaches a uniform distribution, then the resultant radial distribution becomes a reverse triangular distribution (Figure B-7).

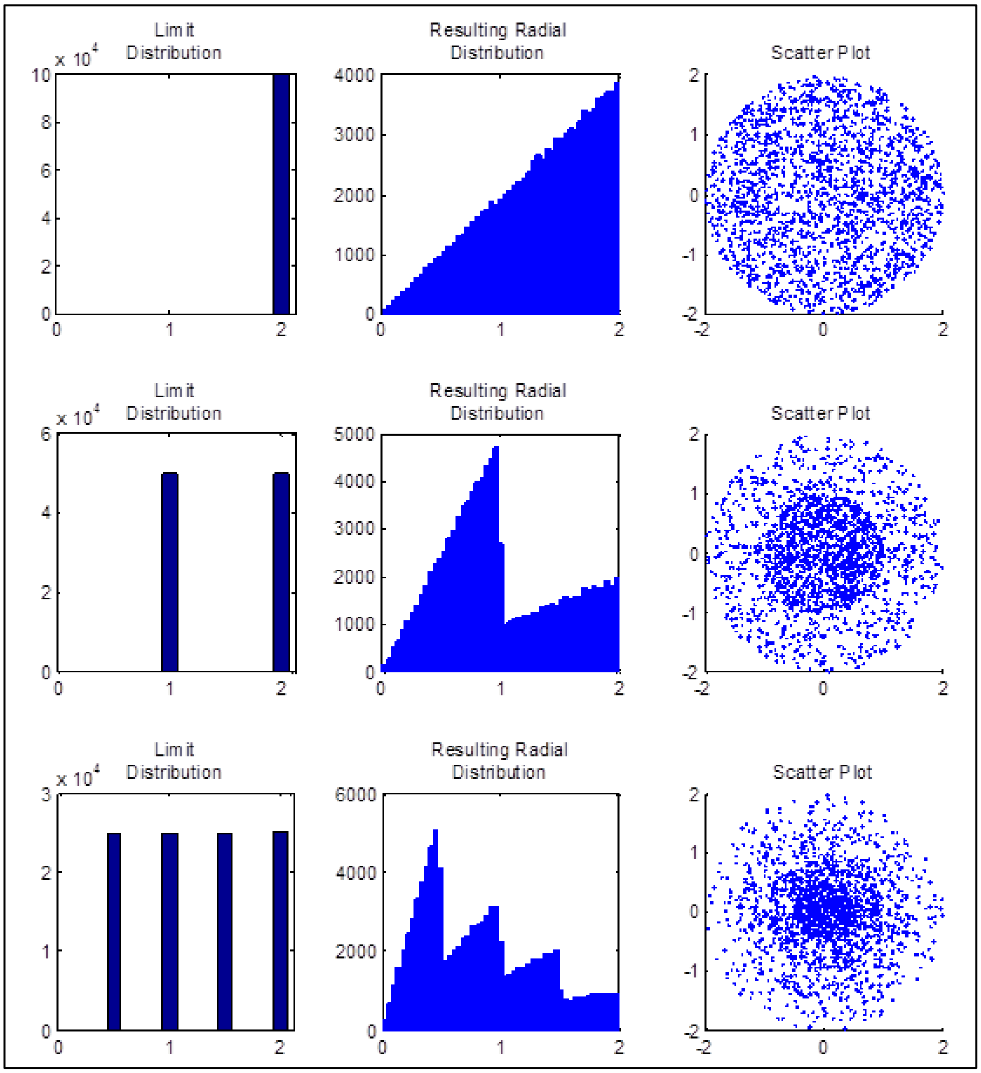


Figure B-8 Progression of distributions

B.2 Addition of Radial Position Vectors

Next we will look at adding random variables together in a way that correctly models the system under analysis. Radial values cannot be simply added together with any meaningful result. They are just one dimension of a two-dimensional vector. There also exists an angle between the two vectors that has to be taken into account. The good news is that while the angle is not trivial, its value is and can be chosen at random from a uniform distribution. That is benefit of radially symmetric systems.

The law of cosines will be used to add position vectors in polar coordinates. It involves less steps than converting back-and-forth to the Cartesian system and is defined as

$$r_{\text{resultant}}^2 = r_1^2 + r_2^2 - 2r_1r_2\cos\theta. \tag{B.6}$$

The variables r_1 and r_2 are the magnitudes of the position vectors 1 and 2 and θ is the angle between them. The variable r_3 is the magnitude of the resultant position vector. The relative angle that this vector makes with the other two does not need to be calculated, because it will never be used. If and when the resultant position is added to a fourth vector, a new random value for θ can be drawn from a uniform distribution.

As an example, consider the part shown in Figure B-9. Two holes of different sizes are located using a coaxiality of .010 inches to the outer diameter which is identified as datum A.

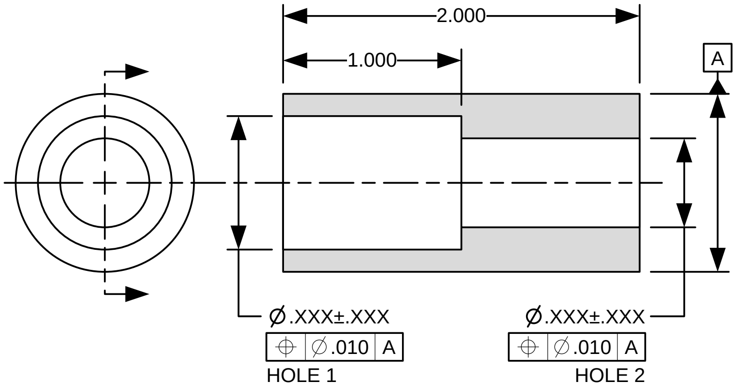


Figure B-9 Application of coaxiality between two features on a drawing

The relative radial distance between the two can be calculated using Eq.(B.6) and the MatLab script below. It assumes that the axis of each hole remains parallel to datum A, the part is radially symmetric, and the random position vector for each axis is independent and uniformly distributed over a two-dimensional area with a diameter of .010 inches.

```
%Monte Carlo Radial Position Analysis
N = 100000;                   %number of trials
m = 2;                        %number of vectors to add
r_lim = .010/2;               %coaxiality requirement (radial)
r = r_lim.*sqrt(rand(m,N)); %generate random radial disp.
r_sum = zeros(1,N);           %initialize sum vector

%Vector addition loop
for i=1:m
    r_sum = sqrt(r_sum.^2+r(i,:).^2-2*r_sum.*r(i,:).*cos(2*pi*(rand(1,N))));
end
```

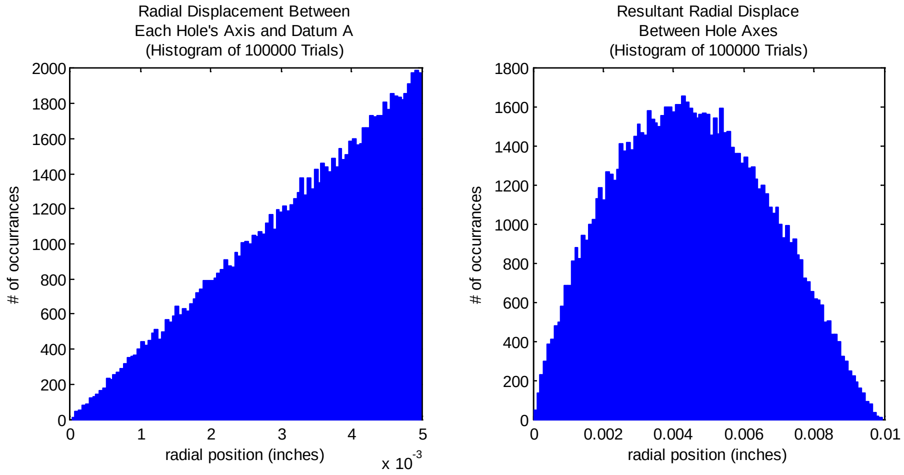


Figure B-10 Results of adding two independent identically distributed radial position vectors (distribution for each individual axis shown left and the resultant distribution after addition shown right)

The resulting distribution for each hole's distance to datum A is shown in the left image of Figure B-10 while the hole-to-hole distance (resultant radial displacement) is on the right. Notice the difference between the two histograms. Adding position vectors has the same effect on radial distances as adding together random variables of any distribution. In one-dimension, the sum becomes normally distributed while in two-dimensions the radial value takes on a Rayleigh distribution. In either case, both are governed by the Central Limit Theorem.

The effect of the Central Limit Theorem is demonstrated in the following two figures. In Figure B-11, the result of adding consecutive independent random variables with identical uniform distributions is shown starting with the distribution of a single variable on the left. Each histogram to the right adds an additional variable and as more variables are added, the distribution takes on a normal shape.

A similar phenomenon is happening in Figure B-12 except the length distribution of the radial position vectors approaches that of a Rayleigh distribution. This is the result adding together individual vectors that have a two-dimensional uniform distribution over a circular area.

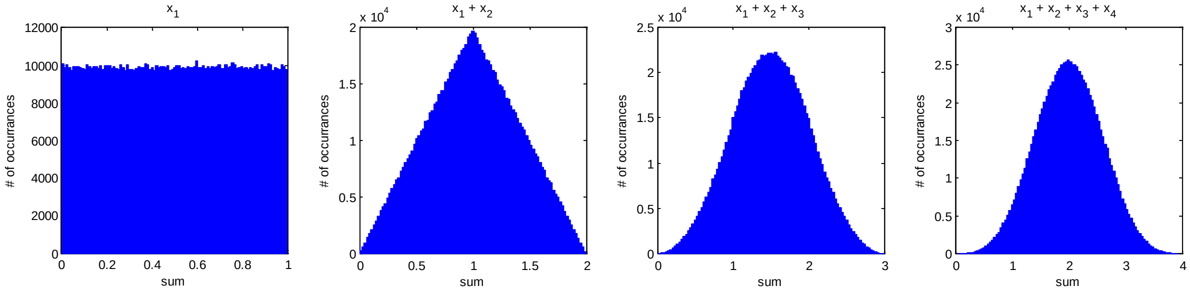


Figure B-11 Results of adding consecutive one dimensional independent uniformly distributed random variables. The left-most image is the distribution of one variable while the right-most image is the sum for four.

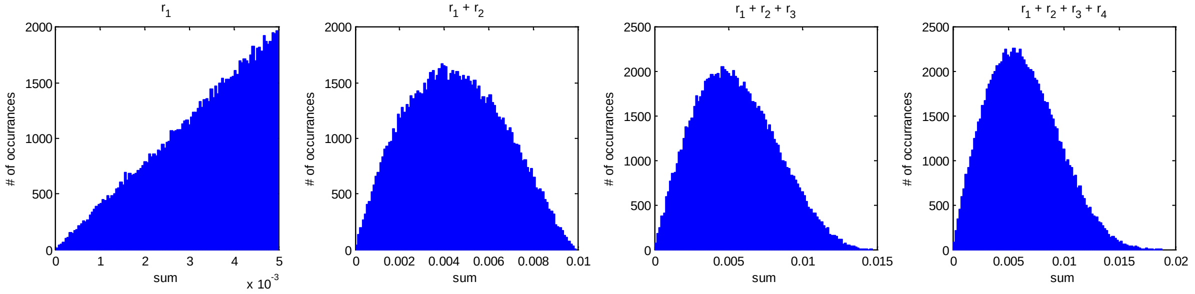


Figure B-12 Resulting radial distribution of using vector addition on consecutive two-dimensional uniformly distributed random vectors. The left-most image is the radial distribution of one vector while the right-most image is the vector sum of four.

B.3 Addition of Orientation Vectors

For the purposes of analyzing the angular relationship between surfaces and axes, a vector will be used to represent the feature's orientation. If the feature is a surface, the vector is the surface normal vector while for axes the vector is parallel to the direction of the axis. In either case, the length is one unit. This vector will be referred to as the angle (or orientation) vector because it will contain the angular information needed in the analysis.

In spherical coordinates, an angle vector can be fully described in three-dimensional space by two independent dimensions: φ and θ (see Figure B-13). φ represents the angle of inclination and θ the azimuthal angle with ranges of $0 \leq \varphi \leq \pi$ and $0 \leq \theta \leq 2\pi$ respectively. The systems under analysis are radially symmetric about the z-axis so like the radial position vector, the value of θ is arbitrary.

The creation of a random angle vector follows the same procedure for creating a random point in a 2-D aperture. To generate a set of random angles (φ) which have an equal (uniform) probability of occurrence anywhere within a given solid angle, a triangular distribution is used. For a Gaussian distribution over the solid angle, φ would have a Rayleigh distribution.

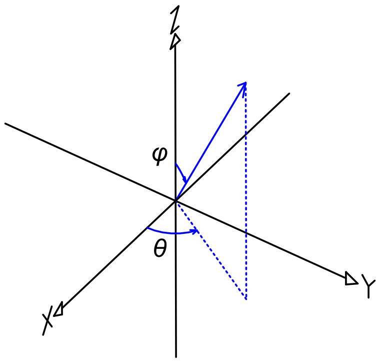


Figure B-13 Definition of variables for angular orientation vectors

Adding together random angles, however, is somewhat different than adding position vectors. The law of cosines, which can be derived from the scalar product (also referred to as the dot product) of two vectors, was used to determine the magnitude of the resultant radial position vector. Here the scalar product will be used to add angle vectors together to get the resultant angle with certain simplifying assumptions.

Given in Cartesian coordinates, the scalar product is defined as

$$|\mathbf{r}_1||\mathbf{r}_2|\cos\alpha = x_1x_2 + y_1y_2 + z_1z_2 \tag{B.7}$$

with α representing the resultant angle between the two vectors. The Cartesian-to-spherical transformation is

$$\begin{aligned} x &= |\mathbf{r}|\sin\varphi\cos\theta \\ y &= |\mathbf{r}|\sin\varphi\sin\theta \\ z &= |\mathbf{r}|\cos\varphi. \end{aligned} \tag{B.8}$$

There are two simplifications that can be made. The first is that all of the magnitudes are unity and the second is that the azimuth of the first vector is zero. The angle is arbitrary. It is only the angle between the two vectors (the θ associated with the second vector) that is important so by setting the first to zero, several terms are eliminated. Doing this results in

$$\cos\varphi_{\text{resultant}} = \sin(-\varphi_1)\sin\varphi_2\cos\theta + \cos(-\varphi_1)\cos\varphi_2; \quad 0 \le \varphi_1, \varphi_2 \le \pi; 0 \le \theta \le 2\pi. \tag{B.9}$$

$\varphi_{\text{resultant}}$ is the resultant angle between the second angle vector and initial z reference axis.

Figure B-14 illustrates how the variables are set-up. The z-axis is the reference; the goal is to relate the second angle vector, R_2 to this axis. First, R_1 is rotated through the angle φ_1 from the z-axis and aligned along the x-axis. R_1 now establishes a new coordinate system and becomes z' from which R_2 is defined using φ_2 and θ. By applying Eq.(B.9), the resultant angle between R_2 and the original z reference axis is computed thus defining R_2 in the original coordinate system.

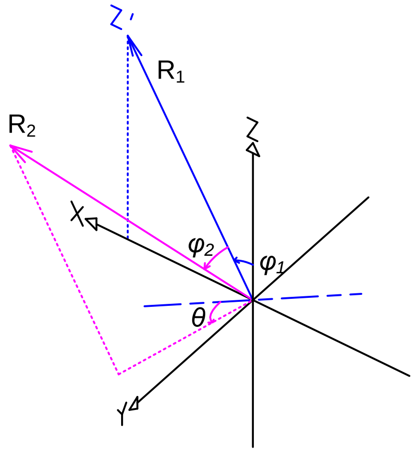


Figure B-14 Addition of angular orientation vectors

This new angle vector can serve as the starting point for adding another angle vector so that in effect it becomes a new R_1. The process repeats itself the same way it did for adding consecutive radial position vectors. There is no need to keep track of θ because the system being modeled is rotational symmetric.

Consider a stack of ten circular disks. The requirement for parallelism between the two faces (wedge) is set at 1.0 degrees and is the same for each disk. The assumed distribution of the actual wedge is uniform over the entire solid angle meaning that φ for each disk will have a triangular distribution. The angular relationship between the top surface of the top disk and the bottom surface of the bottom disk is calculated by the following MatLab Monte Carlo script.

```
%Monte Carlo Wedge Analysis for a 10 Disk Stack
N = 1000000;                    %number of trials
m = 10;                         %number of disks in the stack
phi_lim = pi*1/180;             %angle limit of 1.0 degree
phi = phi_lim*sqrt(rand(m,N)); %random angle w/triangular dist.
phi_sum = zeros(1,N);              %initialize sum vector

%Vector addition loop
for i=1:m
    phi_sum = acos(sin(-phi_sum).*sin(phi(i,:)).*cos(2*pi*rand(1,N))...
        +cos(-phi_sum).*cos(phi(i,:)));
End
```

The resulting histogram for 1,000,000 trials of both the wedge of an individual disk and a stack of 10 disks is displayed in Figure B-15. Even though the individual distributions are triangular, when added-up the result takes on a Rayleigh distribution shape constant with the central limit theorem.

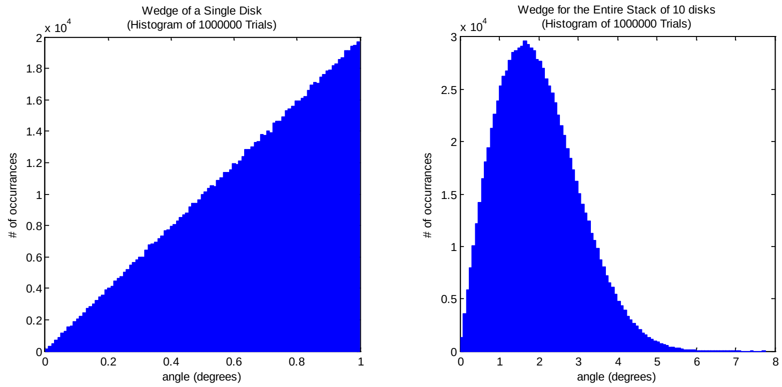


Figure B-15 Resulting wedge angle distribution for a single disk (left) and the entire stack (right)

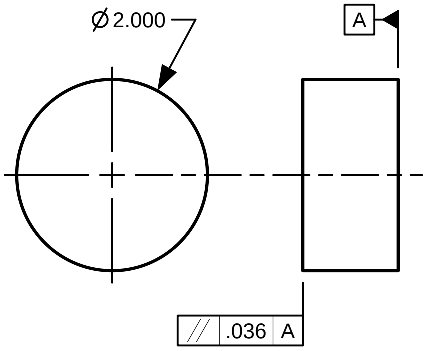


Figure B-16 Application of parallelism on the drawing to achieve 1 degree of maximum wedge between surfaces

Geometric feature controls which govern orientation such as parallelism, perpendicularity, and angularity do not specify an allowable angle directly. Instead they specify a tolerance zone size that the associated axis or plane must stay within. So for the 1.0 degree wedge specified in the previous example, the diameter of the disk must be known to set the parallelism tolerance zone thickness. A two-inch disk needs a .036 inch parallelism to maintain a wedge of 1.0 degrees or less (Figure B-16). The .036 inch tolerance zone is an exact value while the disk diameter could have an associated size tolerance. Depending on the complexity and depth of the analysis, the diameter tolerance may need to be taken into account.

B.4 Modeling Simultaneous Requirements

B.4.1 Effect of Locational Tolerances on Flat Surface Position and Orientation

A locational tolerance zone by itself can control a flat surface's position and orientation simultaneously. The combined variations of both cannot allow the surface to violate the boundary of the tolerance zone.

This means that the two variables are no longer independent. For example, refer to Figure B-17 and Figure B-18. The first figure on the left shows how a locational tolerance zone would be specified on a drawing. Either options 1 or 2 will produce the same results. The size of the tolerance zone is the same used in Figure B-16 to give a maximum wedge angle of 1 degree. The second figure on the right illustrates how the surface might appear as produced. If it is displaced at the limit of its tolerance zone (Δl = .018 inch), the surface would need to be perfectly parallel (φ = 0 degrees) with its reference datum. Conversely, if the surface is at its maximum tilt (φ = 1 degree), the displacement would have to equal its nominal value (Δl = 0.0 inch).

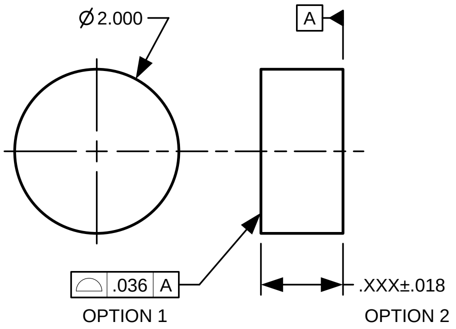


Figure B-17 Drawing application showing locational control of a flat surface (either options 1 or 2 produce the same result)

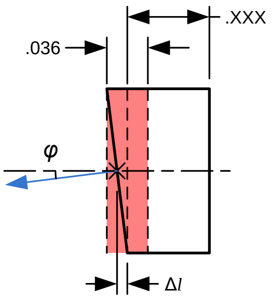


Figure B-18 Possible tilt and position of a flat surface within the tolerance zone

Accurate simulation results depend on knowing the simultaneous distribution for both the surfaces' linear displacement and tilt angle. Guessing a distribution for either is not a straight-forward task because the two are not independent. To tackle this problem, it will be assumed initially that each variable is uniformly distributed; Δl over the width of the tolerance zone and φ over its solid angle. From that assumption, a data set of independent random angles and positions is then generated. Next the data set is filtered keeping only the combinations of angle and position that lie within the tolerance zone. Finally, statistics on the filtered data set can be used to determine the resulting variable distributions.

The process is captured in the following MatLab code. Both the filtered and unfiltered resulting distributions for position and tilt are shown in Figure B-19.

```
%Monte Carlo Simultaneous Requirements Analysis for Location
%Variable definition
D = 2;                  %Disk Diameter
W = .036;               %Tolerance Zone width
N = 1000000;            %Number of Monte Carlo Trials
phi_limit = atan(W/D); %Max tilt angle limit

%Create random data set
phi = phi_limit*sqrt(rand(1,N)); %Random angle w/triangular dist.
del_d = W*(rand(1,N)-.5);        %Random position w/uniform dist.

%Filter
test = ((D/2)*tan(phi)+abs(del_d) <= W/2); %Create Logical Filter Matrix
phi_ = phi(test);                          %Filtered tilt angles
del_d_ = del_d(test);                      %Filtered radial values
```

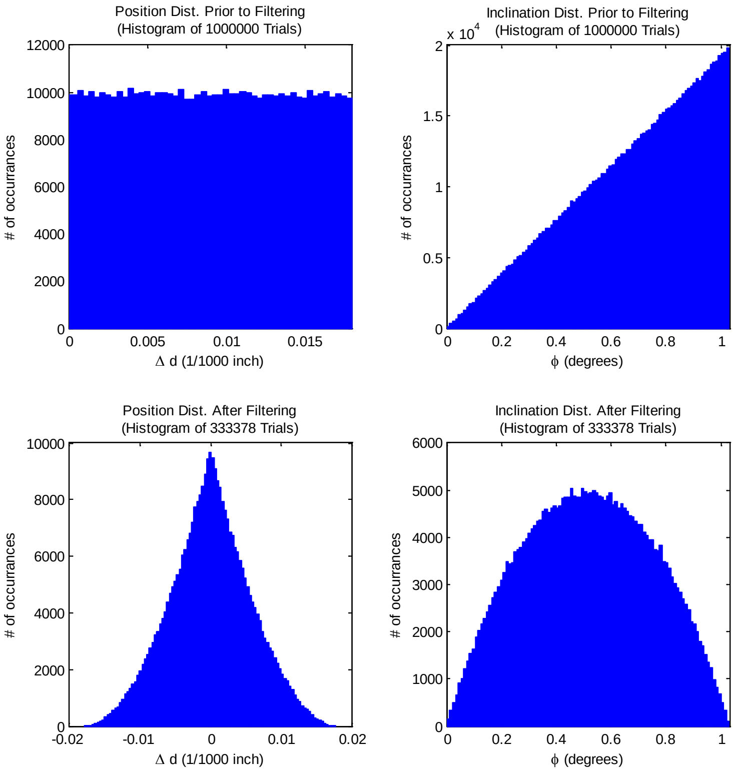


Figure B-19 Distributions on the position and angle of a flat surface. Unfiltered data set (top row) when position and orientation are independent and filtered (bottom row) when both are considered simultaneously

Figure B-20 compares the wedge distribution on a stack of ten disks between the two methods on controlling surface orientation discussed so far. The red data set represents disks whose surface orientation is controlled solely through parallelism according to Figure B-15. The total stack has an RMS wedge value of 2.3 degrees. In the blue data set, each disk is represented according to Figure B-17. It is produced by filtering the red data keeping only the angles that stay within the tolerance zone when simultaneously accounting for the surface’s location. The resulting wedge on the stack decreases from 2.3 degrees RMS to 1.8 degrees RMS.

```
%Monte Carlo Simulation
%Stack of 10 disks - Simultaneous Req.
%Variable definition
D = 2;                 %Disk diameter
W = .036;              %Tolerance zone width
N = 1000000;           %# of trials
stk = 10;              %# of disks in a stack
phi_limit = atan(W/D); %Max tilt angle limit

%Create random data set
phi = phi_limit*sqrt(rand(1,N*stk)); %Angle
del_d = W*(rand(1,N*stk)-.5); %Position

%Find & exclude out-of-bounds angle data
phi_ = phi((D/2)*tan(phi)+abs(del_d) <= W/2);
phi_(1:rem(nnz(phi_),stk)) = []; %Resize

%Reshape
phi_ = reshape(phi_,stk,[]);
phi = reshape(phi,stk,[]);

%Vector addition loop
[~,n] = size(phi_); %# of filtered data trials
phi_sum = zeros(1,N); %Initialize
phi_sum_ = zeros(1,n); %Initialize
for i=1:stk

 phi_sum=acos(sin(-phi_sum).*sin(phi(i,:))...
         .*cos(2*pi*rand(1,N))...
         +cos(-phi_sum).*cos(phi(i,:)));

 phi_sum_=acos(sin(-phi_sum_).*sin(phi_(1,:))...
          .*cos(2*pi*rand(1,n))...
          +cos(-phi_sum_).*cos(phi_(1,:)));
end
```

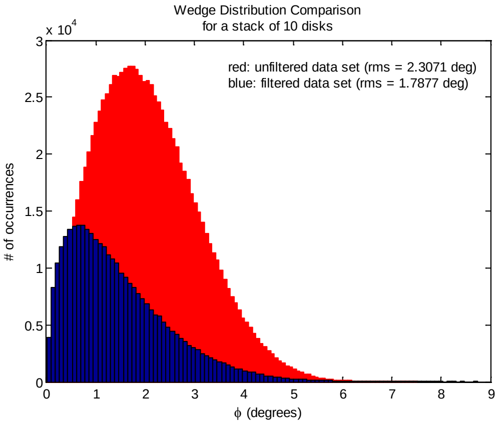


Figure B-20 Wedge distribution comparison on a stack of 10 disks between using parallelism (red data set) versus location (blue data set)

B.4.2 Effect of Coaxiality on Axis Position and Orientation

Coaxiality can control both an axis’ position and orientation simultaneously similar to how a locational tolerance zone can control a flat surface. The combined variations of tilt and decenter cannot allow the axis to violate the boundary of the tolerance zone. If the axis is displaced radially at the limit of the tolerance zone, it would need to be perfectly aligned with its reference axis. The opposite condition is also true; if the axis is tilted at its maximum angle, it would have to have zero decenter.

Simulating the distributions of tilt and decenter is done using a procedure similar to that used in the last section; a data set is generated assuming the variables are independent then filtered according to the tolerance zone. Simulating the effects of coaxiality however breaks radial symmetry so along with decenter and tilt, the azimuthal angle θ will have to be tracked and filtered as well.

As an example, consider the coaxiality requirement on the part shown in Figure B-21 below.

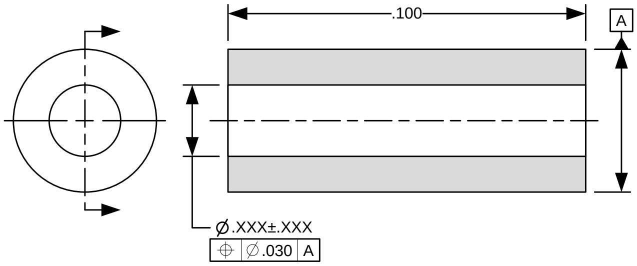


Figure B-21 Drawing application of the coaxiality feature control

The axis of the hole can displace radially some distance r and tilt some angle ϕ from datum A (see Figure B-22 where the z-axis is datum A) as long as it remains completely with the .030 inch diameter tolerance zone. Displacement of the axis is measured at the middle of the tolerance zone's length. This tolerance zone is perfectly centered on datum axis A and extends the length of the feature (.100 inch).

The following algorithm is used to generate a set of random axes. It assumes an initial radial displacement with a uniform probability of occurrence over the .030 inch diameter area and a tilt that also has a uniform distribution over a solid angle defined by the tolerance zone's diameter and length. That data set is then filtered to keep only those axes that stay within the tolerance zone. The algorithm does this by comparing where the axis exits either end of the zone to the cut-off value of .015 inches.

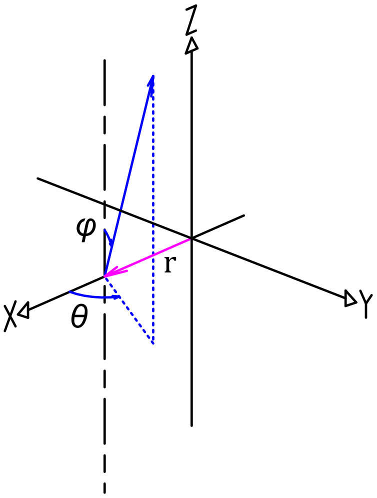


Figure B-22 Definition of simulation variables

```
%Monte Carlo Simultaneous Requirements Analysis for Coaxiality
%Variable definition
D = 30;         %Tolerance Zone Diameter
L = 100;        %Tolerance Zone Length
N = 1000000;    %Number of Monte Carlo Trials

%Create random axis data set
r_limit = D/2;                  %Max Radius Limit
r = r_limit*sqrt(rand(1,N));    %Random Radius w/Triangular Dist.
phi_limit = atan(D/L);          %Max Tilt Angle Limit
phi = phi_limit*sqrt(rand(1,N));%Random Angle w/Triangular Dist.
theta = 2*pi*rand(1,N);         %Angle Between Radial Vector and Tilt

%Filter
%Projected radius due to tilt at end of tol. zone
rrho = (L/2)*tan(phi);
%Compare resultant vector at left end of tolerance zone to tol. dia.
test1 = (sqrt(r.^2 + rrho.^2 - 2*(r.*rrho.*cos(theta))) <= D/2);
%Compare resultant vector at right end of tolerance zone to tol. dia.
test2 = (sqrt(r.^2 + rrho.^2 - 2*(r.*rrho.*cos(theta+pi))) <= D/2);
test = test1&test2; %Create Logical Filter Matrix

%Apply logical filter matrix to original data set
r_ = r(test);           %Accepted radial values
phi_ = phi(test);       %Accepted tilt angles
theta_ = theta(test);   %Accepted rotation angles
```

In Figure B-23 below, the left image shows a set of unfiltered axes in relation to the tolerance zone and the right image is of those axes that remain after filtering. The green surface is the tolerance zone.

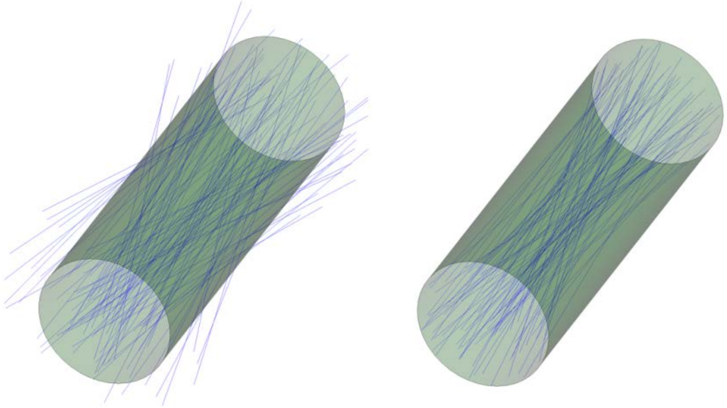


Figure B-23 Visual comparison between the unfiltered axis data set (left) and the filtered (right)

Compared to the initial set of distributions for *r*, *φ*, and θ, the filtered data set exhibits a more Raleigh appearance for r and φ while θ clearly indicates that the system is no longer radially symmetric (see Figure B-24). This means the initial choice of variable distributions does not have a large impact on the outcome. It is also interesting to note that from an initial data set of 1,000,000 random axes; only around 25% were completely within the tolerance zone.

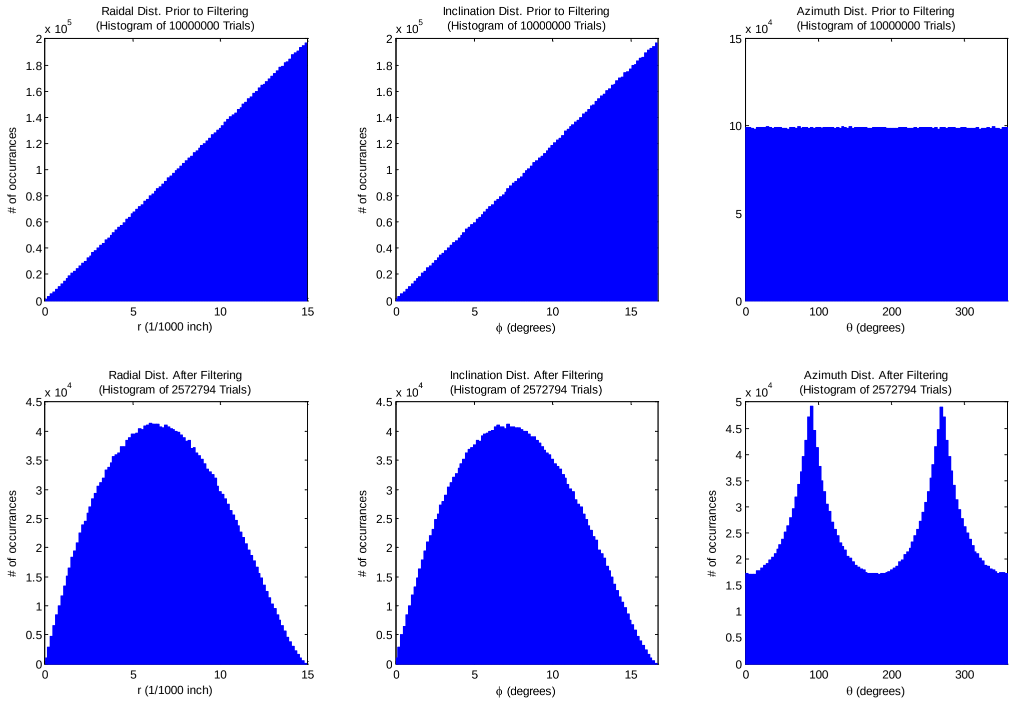


Figure B-24 Resulting distributions on position and angle for the unfiltered data set (top row) and filtered (bottom row)

B.4.2.1 Minimum Distance between Skew Axes

It is important to note that the radial position shown in Figure B-24 is defined as the distance from datum A that the random axis intersects the mid-plane of the tolerance zone. This is not necessarily the minimum distance between the two axes. The generated random axis will most likely be skew to datum A meaning that it will be neither parallel nor intersecting.

Calculating position as was done in the previous section is useful when knowledge of radial position at pre-defined locations is necessary. This is the type of information commonly used by lens design software where surfaces are located at various stations along the z-axis.

The situation may also arise where knowledge of the minimum distance between axes is needed. Given points on a line (or axis) represented by a position vector **r**, the general form for calculating the distance between skew lines is

$$D_{\min} = \left| \left(\mathbf{r}_3 - \mathbf{r}_1\right) \cdot \frac{\left(\mathbf{r}_2 - \mathbf{r}_1\right) \times \left(\mathbf{r}_4 - \mathbf{r}_3\right)}{\left\| \left(\mathbf{r}_2 - \mathbf{r}_1\right) \times \left(\mathbf{r}_4 - \mathbf{r}_3\right) \right\|} \right|. \tag{B.10}$$

Points $\mathbf{r}_1$ and $\mathbf{r}_2$ lie on the first axis while points $\mathbf{r}_3$ and $\mathbf{r}_4$ lie on the second axis. If calculating the distance to datum axis A, several simplifying substitutions can be made. First, by setting datum A coincident with the z-axis, three of the points can be defined as follows:

$$\begin{aligned}\mathbf{r}_1 &= r_1\hat{\mathbf{i}},\\ \mathbf{r}_3 &= 0\hat{\mathbf{i}}+0\hat{\mathbf{j}}+0\hat{\mathbf{k}},\\ \mathbf{r}_4 &= \hat{\mathbf{k}}.\end{aligned} \qquad \text{(B.11)}$$

Note that point $\mathbf{r}_3$ is the origin and $\mathbf{r}_4$ is the unit vector in the z-direction. The second point on the first axis $\mathbf{r}_2$ is where the axis exits the opposite end of the tolerance zone. Where L is the length of the tolerance zone,

$$\mathbf{r}_2 = \left(\frac{L}{2}\tan\varphi\cos\theta - r_1\right)\hat{\mathbf{i}} + \frac{L}{2}\tan\varphi\sin\theta\hat{\mathbf{j}} + \frac{L}{2}\hat{\mathbf{k}}. \qquad \text{(B.12)}$$

Substituting Eq.(B.11) and Eq.(B.12) into Eq.(B.10), results in

$$D_{\min} = \left|r_1\sin\theta\right|. \qquad \text{(B.13)}$$

Note that the minimum distance between datum A and the generated random axis is independent of φ.

Using the same data set as shown in Figure B-24, the minimum distance between skew axes is calculated using Eq.(B.13) for both the filtered and unfiltered data. The results are shown in Figure B-25 below.

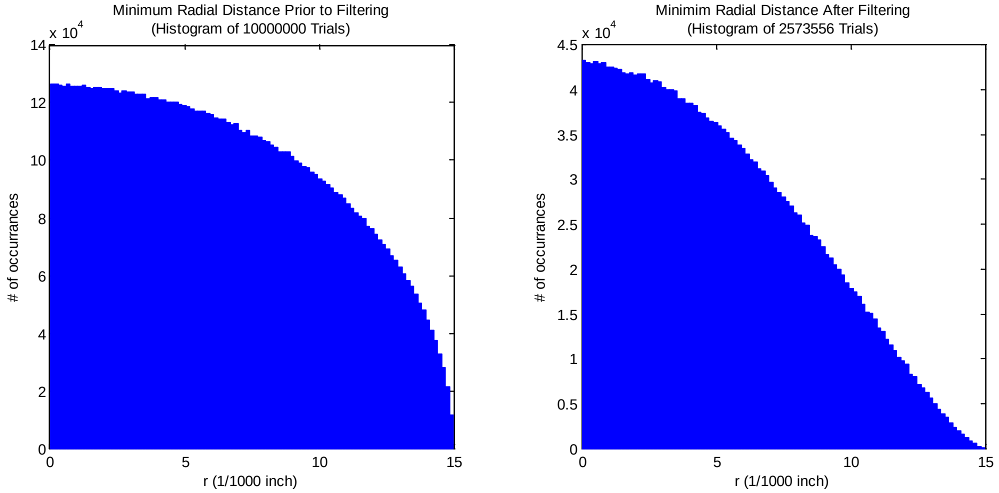


Figure B-25 Distribution of the minimum distance between skew lines. The left histogram is of all the randomly

The distribution for the minimum radial distance is quite different than that for radial position. This is because radial position takes samples where all of the generated random axes cross a plane perpendicular do datum A. It is therefore one of two variables in a two-dimensional distribution. The minimum radial distance can occur anywhere along datum A out to $\pm\infty$. This is obvious looking at the histograms for the unfiltered data. The maximum radial value of .015 inches is where radial position (Figure B-24) has the most occurrences, but when the angle θ is factored in (Figure B-25), the number of occurrences drops to near zero. For the minimum radial distance to be equal to the maximum radial position, θ has to be exactly equal to either 90 or 270 degrees.

B.4.3 Refining Orientation with Parallelism

It is possible to tighten the tolerance on orientation while maintaining the same coaxiality requirement by applying a separate parallelism feature control as shown in Figure B-26.

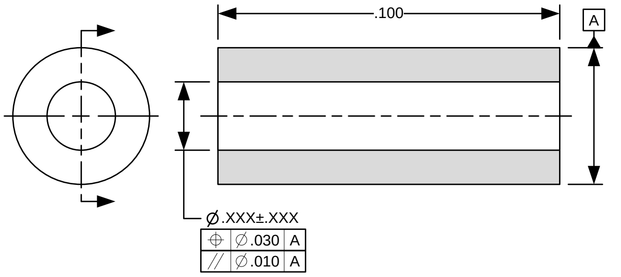


Figure B-26 Application of parallelism geometric control to the drawing

Now any axis must meet both the .030 inch diameter coaxiality requirement and the .010 inch diameter parallelism requirement simultaneously.

The MatLab code from above is modified so that the diameter from which the maximum tilt angle is determined is .010 not .030. Doing so and running the code produces a data set with the distributions shown in Figure B-27. This time, approximately 72% of the data set was kept out of 1,000,000 trials

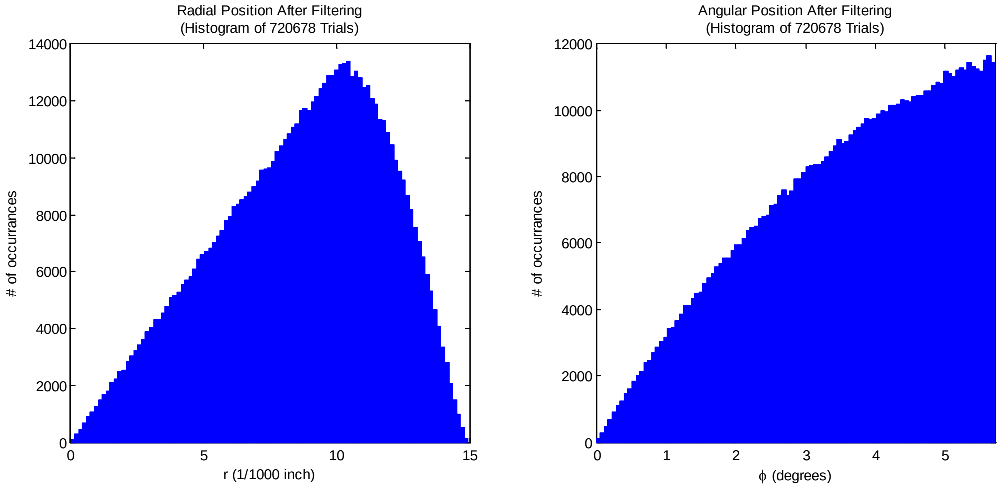


Figure B-27 Resulting distributions for axis position (left) and angle (right) with addition of parallelism requirement

In Figure B-28 below, it can also be seen that the resulting axes are more uniformly parallel. The left image is with the .010 parallelism and the right image is without.

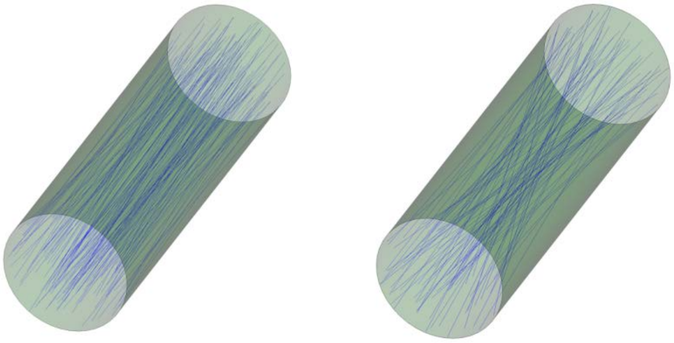


Figure B-28 Visual comparison between having a parallelism requirement (left) versus none (right) on axial orientation

Appendix C: MatLab scripts written for this report

These MatLab scripts interact with Zemax to find the first and second order derivatives to the Taylor polynomial approximation. To run the scripts, copy them into MatLab and include the Zemax DDE toolbox in the search path. It consists of a set of functions written to facilitate communication with Zemax using the Microsoft operating system Dynamic Data Exchange (DDE) protocol. The Zemax DDE toolbox can be downloaded at “https://www.zemax.com/downloads/MZDDE3b.zip”.

C.1 Script to collect coefficients from the Zemax lens model

```
function [J,H,delta,merit_nom,merit] = LSCR(lens_file)
%Lens Sensitivity Collection Routine (LSCR)
%Written by Chris Hopkins (chopkins@arizona.edu) 11/21/2014
%
%Uses quadratic approximation to calculate 1st-order partial derivatives
%(output column vector J) and 2nd order partial derivatives(output square
%matrix H). Reads the Tolerance Data Editor of the input Zemax file.
%Currently recognizes only the tolerance operators TTHI, TEDX/Y, and
%TETX/Y. Max and min tolerance values MUST be equal in magnitude and
%opposite in sign. Must include the Zemax DDE toolbox (MZDDE) in the
%search path.

global surf; %two column matrix which records tolerance operand surfaces
global ray_aim_state; %used to switch ray aiming on/off
ray_aim_state = 0; %initialize state

%Establish communication with Zemax, load the file, and read the TDE
zDDEInit %initialize connection
zLoadFile(lens_file);
[TolTypes,TolMatrix] = zGetTolMatrix; %read the TDE
TolTypes = cellstr(TolTypes); %convert from char array to cell array

%Identify valid tolerance types
idx = logical(strcmp(TolTypes,'TTHI')+...
              strcmp(TolTypes,'TEDX')+...
              strcmp(TolTypes,'TEDY')+...
              strcmp(TolTypes,'TETX')+...
              strcmp(TolTypes,'TETY'));
TolTypes = TolTypes(idx);

surf = TolMatrix(idx,[1 2]); %read out the surface numbers
idx_TTHI = logical(strcmp(TolTypes,'TTHI'));
surf(idx_TTHI,1) = surf(idx_TTHI,1)+1; %remap TTHI surf 1 for coord brks
delta = TolMatrix(idx,[3 4]); %read out the delta tol values
n = sum(idx); %total number of valid tolerance operators

%Evaluating Merit Function Cross Terms
merit_nom = zOptimize(0); %get nominal merit function value
merit = zeros(n,n,4); %initialize array

for i = 1:n
    for j = i:n
        if i==j
            del=(1/2)*delta;
        else
            del=delta;
        end
        for k1 = 1:2
            coord_brk(i);
            AdjustSurf(surf(i,:),TolTypes{i},del(i,k1));
            for k2 = 1:2
                coord_brk(j);
                AdjustSurf(surf(j,:),TolTypes{j},del(j,k2));
                merit(i,j,2*(k1-1)+k2) = zOptimize(0);
                AdjustSurf(surf(j,:),TolTypes{j},-del(j,k2));
                undo_brk(j);
            end %end k2
            AdjustSurf(surf(i,:),TolTypes{i},-del(i,k1));
            undo_brk(i);
        end %end k1
    end %end j
end %end i

%Build the Jacobian Vector (first-order partial derivatives)
J = (1/2)*diag(merit(:,:,1)-merit(:,:,4))./delta(:,1);

%Build the Hessian Matrix (second-order partial derivatives)
H = merit(:,:,1)-merit(:,:,2)-merit(:,:,3)+merit(:,:,4);
H = H + triu(H,1)'; %mirror upper off-diag. terms to the lower off-diag.
delta_mtrx = (delta(:,1)*delta(:,1)')./(ones(n)+3*eye(n));
H = (1/4)*H./delta_mtrx;

%Close communications with Zemax
zDDEClose
end

function AdjustSurf(surf_idx,TolType,delta)
    code = 3*strcmp(TolType,'TTHI')+...
           1*strcmp(TolType,'TEDX')+...
           2*strcmp(TolType,'TEDY')+...
           3*strcmp(TolType,'TETX')+...
           4*strcmp(TolType,'TETY');
switch TolType
    case 'TTHI'
        surf_val(1) = zGetSurfaceData(surf_idx(1), code);
        surf_val(2) = zGetSurfaceData(surf_idx(2), code);
        zSetSurfaceData(surf_idx(1), code, surf_val(1)+delta);
        zSetSurfaceData(surf_idx(2), code, surf_val(2)-delta);
    case {'TEDX','TEDY','TETX','TETY'}
        surf_val(1) = zGetSurfaceParameter(surf_idx(1), code);
        surf_val(2) = zGetSurfaceParameter(surf_idx(2), code);
        zSetSurfaceParameter(surf_idx(1), code, surf_val(1)+delta);
        zSetSurfaceParameter(surf_idx(2)-1, code, surf_val(2)-delta);
end
end

function coord_brk(i)
%subroutine to insert coordinate break
global surf;
RayAimSwitch(0);

%insert coordinate break before 1st surface of lens group
zInsertSurface(surf(i,1)); %insert surface
surf(surf>surf(i,1)) = surf(surf>surf(i,1))+1; %re-adjust surface index
zSetSurfaceData(surf(i,1),0,'COORDBRK'); %set as coordinate break surf

%insert new surface, return to 1st coordinate break
zInsertSurface(surf(i,2)); %insert surface
surf(surf>=surf(i,2)) = surf(surf>=surf(i,2))+1; %re-adjust surface index
radius = zGetSurfaceData(surf(i,2),2);
zSetSurfaceData(surf(i,2)-1,2,radius);
zSetSolve(surf(i,2)-1,1,7,surf(i,1),0);

%insert coordinate break, return to last surface of group
zInsertSurface(surf(i,2)); %insert surface
surf(surf>=surf(i,2)) = surf(surf>=surf(i,2))+1; %re-adjust surface index
zSetSurfaceData(surf(i,2)-1,0,'COORDBRK'); %set as coordinate break
zSetSolve(surf(i,2)-1,1,5,surf(i,2)-2,-1,0,0);

RayAimSwitch(1);
end

function undo_brk(i)
%subroutine to remove coordinate break
```

```
global surf;
RayAimSwitch(0);

zDeleteSurface(surf(i,1)); %delete surface
surf(surf>surf(i,1)) = surf(surf>surf(i,1))-1; %re-adjust surface index
zDeleteSurface(surf(i,2)-1); %delete surface
surf(surf>surf(i,2)-1) = surf(surf>surf(i,2)-1)-1; %re-adjust surface index
zDeleteSurface(surf(i,2)-1); %delete surface
surf(surf>surf(i,2)-1) = surf(surf>surf(i,2)-1)-1; %re-adjust surface index

RayAimSwitch(1);
end

function RayAimSwitch(flag)
%subroutine to disable (flag = 0)/enable (flag = 1) ray aiming
%Zemax may crash if ray aiming is enabled during insertion of coordinate
%breaks.
global ray_aim_state;
p = zGetSystem; %get current system parameters
p([1 4])=[]; %eliminate invalid input parameters
if flag == 0 %turn ray aiming off
    ray_aim_state = p(3);
    p(3)=0;
else %reinstate ray aiming
    p(3) = ray_aim_state;
end
zSetSystem(p(1),p(2),p(3),p(4),p(5),p(6),p(7));
end
```

C.2 Script to perform the Monte Carlo analysis and estimate statistics

```
function [est_del_mean,est_stdv_1,est_stdv_2] = PolySim(N,J,H,delta,nom)
%Monte Carlo Simulation of a 2nd-order multivariate Taylor polynomial
%Written by Christopher Hopkins (chopkins@optics.arizona.edu) 11/18/2014
%-All input variables are given a normal dist. with a standard deviation
%that is 1/3 of the tolerance limit
%-Output is a histogram showing the probability density of the rms
%wavefront error
%-Function inputs are as follows:
%   N: number of trials
%   J: Jacobian: n x 1 matrix of 1st-order partial derivatives
%   H: Hessian: n x n matrix of 2nd-order partial derivatives
%   delta: n x 2 matrix of +/- tolerance limits for each input variable
%   nom: nominal wavefront error value

%Set distribution
std_dev = (1/3)*delta(:,2)'; %standard deviation
n = numel(std_dev); %number of input variables

%Set normal distribution
X = ones(N,1)*std_dev.*randn(N,n);

%Set truncated normal distribution
% a = 3; %number of standard deviations
% X = sqrt(2)*ones(N,1)*std_dev.*erfinv(erf(a/sqrt(2)).*(2.*rand(N,n)-1));

%Monte Carlo run
WFE = zeros(1,N);
for i=1:N
     WFE(i) = nom + J'*X(i,:)' + .5*X(i,:)*H*X(i,:)';
end

%Calculate WFE at 97.7% yield
yield = sort(WFE);
gg = yield(floor(N*.977));

%Results of Monte Carlo Run
```

```
figure;
hist(WFE,101);
h = findobj(gca,'Type','patch');
set(h,'FaceColor','b','EdgeColor','b');
hold on;
xlim([min(WFE) mean(WFE)+8*std(WFE)]);
plot([gg,gg],ylim,'--');

title({'RMS Wavefront Error - Monte Carlo Simulation';
       ['(based on 2nd-order approximation, ',num2str(n),' variables, ',...
       num2str(N),' trials)']});

xlabel('rms wavefront error (um)');
ylabel('# of occurrences');
gtext({['\mu = ',num2str(mean(WFE)),' um'],...
       ['\sigma = ',num2str(std(WFE)),' um'],...
       ['97.7% < ',num2str(gg),' um']});

%Estimated Statistics
%estimate of the mean
est_del_mean = 0.5*(std_dev.^2)*diag(H);

%estimate of the standard deviation with cross-terms
est_var_1 = std_dev.^2*J.^2 + 0.5*(std_dev.^2*H.^2*std_dev.^2');
est_stdv_1 = sqrt(est_var_1);

%estimate of the standard deviation without cross-terms
est_var_2 = std_dev.^2*J.^2 + 0.5*std_dev.^4*diag(H).^2;
est_stdv_2 = sqrt(est_var_2);
end
```