Summer 2022

Optimization of Rover Wheel Geometries for Planetary Missions

Nikita Amberkar
Embry-Riddle Aeronautical University, amberkan@my.erau.edu

OPTIMIZATION OF ROVER WHEEL GEOMETRIES FOR PLANETARY MISSIONS

By

Nikita Amberkar

A Thesis Submitted to the Faculty of Embry-Riddle Aeronautical University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Aerospace Engineering

August 2022

Embry-Riddle Aeronautical University

Daytona Beach, Florida

OPTIMIZATION OF ROVER WHEEL GEOMETRIES FOR PLANETARY MISSIONS

By

Nikita Amberkar

This Thesis was prepared under the direction of the candidate's Thesis Committee Chair, Dr. Mandar Kulkarni, Department of Aerospace Engineering, and has been approved by the members of the Thesis Committee. It was submitted to the Office of the Senior Vice President for Academic Affairs and Provost, and was accepted in the partial fulfillment of the requirements for the Degree of Master of Science in Aerospace Engineering.

THESIS COMMITTEE

Chair, Dr. Mandar Kulkarni	Member, Dr. Ali Yeilaghi Tamijani
Member, Dr. Alberto Mello	Member, Dr. Sirish Namilae
Graduate Program Coordinator, Dr. Hever Moncayo	Date
Dean of the College of Engineering, Dr. James W. Gregory	Date
Associate Provost of Academic Support, Dr. Christopher Grant	Date

ACKNOWLEDGEMENTS

I would like to thank you prof. Dr. Mandar Kulkarni for helping and supporting me on a topic that I came up with. His mentorship and guidance gave me courage to take up this challenging project and this provided me with a unique aspect to my research. He helped me grow my knowledge and critical thinking skills by asking relevant questions. I also want to thank his student research team for helping me with this research project.

I would like to thank my committee members, Dr. Tamijani, Dr. Mello, and Dr. Namilae. Their encouraging words and detailed feedback on the paper have been important to me. Thank you for providing time out of your schedule to support my research and make this project possible.

I would also like to thank my mom, dad, twin sister, and my younger sister for their encouragement, love, and support. I am also thankful for my friends who helped me move along with this research.

ABSTRACT

Rovers have been launched into space for exploration of the Moon and Mars to collect samples of rock and soil. To continue the explorations, the rovers need to have reliable wheels to drive around. However, due to the soil being soft, the wheels on the rover start to lose traction and the wheels sink while driving to various locations. Previous work in this field has been done experimentally or with the use of simulations. Only a few references report the effect of uncertainties in grouser simulation on the traction efficiency.

The objective of this work was to (a) Understand the effect of uncertainties on wheel traction efficiency, and (b) Design a rover wheel, consider those uncertainties, and then compare results with deterministic optimization. The results are categorized into three different sections. The first section shows the result of a closed-form equation for rover traction efficiency. A closed-form equation was obtained using three different formulas from previous work. The second section provides results on a reliability analysis to understand the effects of uncertainty on traction efficiency. The uncertainty variables chosen were the empiric soil parameter, K_u, the weight of the wheel, w, and the width of the wheel, b. The third section has a result of using the reliability-based design for the wheel considering those uncertainties, in which the design parameters are the normalized height of the grousers, $\hat{h}$, the width of the wheel, b, the radius of the wheel, r, and finally the weight of the wheel, w. In the reliability-based optimization there are two variables that are considered uncertain which are not the design parameters, the soil parameter and torque. In the design parameters, the radius of the wheel is considered uncertain. Once the optimized values are obtained, they are compared to the deterministic optimization. As a result, optimized design variables were obtained.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

NOMENCLATURE

b	Width of wheel
DEM	Discrete element method
D	Diameter of wheel
FEA	Finite element analysis
FORM	First order reliability method
F_R	Rolling force
F_x	Force in the x-direction
$\hat{h}$	Height of grousers normalized by the radius of the wheel
i_x	Slip ratio in the x-direction
K_u	Empiric soil parameter
JPL	Jet Propulsion Laboratory
MCS	Monte Carlo simulations
MER	Mars exploration rover
MPP	Most probable point
NASA	National aeronautics and space administration
r	Radius of wheel
SORM	Second order reliability method
T	Torque
w	Weight of wheel
$\hat{z}$	Sinkage normalized by the radius of the wheel
$\emptyset$	The angle between two successive grousers

1 Introduction

Grousers are features on the surface of wheels that are designed to increase the traction of the wheels while rolling on sand-like terrains. In terms of their function, grousers on the wheels of planetary rovers act similar to the tire treads on conventional automobile wheels. The geometry of such grousers plays a vital role while operating on planetary surfaces as shown in Figure 1.1.

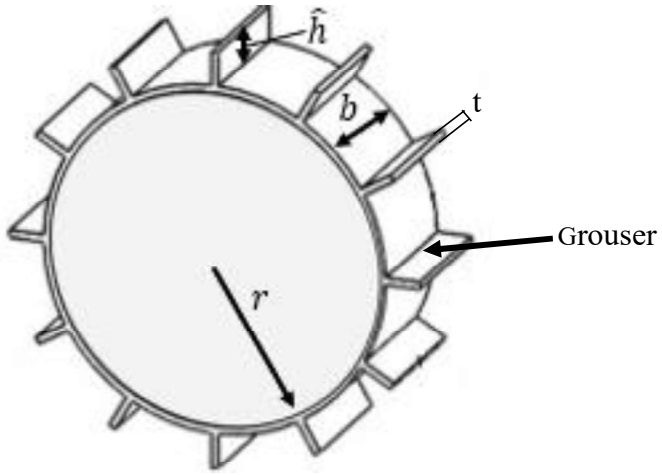

Figure 1.1 Wheel geometry

Planetary surfaces typically consist of terrain with regolith, which can be a soft or rough terrain, hence proving that increasing the traction of the rover wheels is pertinent to the success of the rover mission. The design of the rover wheel with grousers involves the modeling of the interaction of the ground and terramechanics to meet the required performances of rovers in terms of traction and moveability on the regolith. Terramechanics is the study of soil properties, specifically the interaction of wheeled or tracked vehicles on various surfaces. The generation of enough traction to traverse loose, sand-like terrain is important to complete the

missions. This thesis is focused on developing a systematic framework for the design of rover wheels to increase traction efficiency.

Another contribution of this thesis work is analyzing the reliability of rover wheels considering the uncertainty in soil due to various soil types, the uncertainty in applied torque due to signal processing error, and the uncertainty in the wheel dimensions due to manufacturing error. Further, the rover wheels are optimized in the presence of three uncertainties.

1.1 Background

Rovers are used extensively for space exploration and sample collection missions to the Moon and Mars. Challenging prospective missions require rovers that have reliable wheels to navigate the harsh conditions of the planetary regolith. A major issue is the wheels of the rover losing traction due to the regolith being too soft. The tractive force, produced from the interaction between the wheel and ground, determines the rover's ability to accelerate, climb slopes, and cross over obstacles. Due to sinkage, rovers operating on soft soil must handle a higher resistance, and therefore, improving traction efficiency is a major part of achieving an optimal tractive performance [1].

NASA [2] suspected that in the past there was liquid water on Mars, so Spirit and Opportunity were sent to distinct locations on the opposite sides of Mars. In 2003, [2] Spirit was launched to the south side of Mars. Spirit started running into multiple problems. One of the main problems occurred in 2009 when one of the wheels on Spirit became stuck in the soft soil and the other five wheels were unable to generate traction against the ground. The Jet Propulsion Laboratory (JPL) operators tried to free up the rover, but one of the other five wheels ceased functioning. In 2004, [3] NASA launched its twin rover, named Opportunity, to the north side of Mars. However, in 2005, all six of Opportunity's wheels were buried in the soil. JPL operators were able to free it up

from the buried soil and it took them about six weeks to maneuver the rover a few inches at a time. The goals of both rovers were to perform geological investigations and to take photographs. However, while traversing on loose soil both rovers struggled and did not work due to other complications and damage to the wheels.

Today NASA uses chevron and straight grousers on their wheels. For example, in 2011 [4] NASA launched the Curiosity rover to Mars. The wheels have 24 chevron-shaped grousers with cleats to help with traction. One of the issues found with the chevron-shaped grousers was the damage to the wheels and grousers. The damage can jeopardize the wheels' ability to carry the weight of the rover, and if three grousers are damaged it indicates that the wheels have reached 60% of their lifetime. In 2020, [5] NASA launched the Perseverance rover which has 48 straight grousers. The Perseverance rover wheel size was designed to be larger than the Curiosity rover wheel to help with traction. Both the Curiosity and Perseverance rovers are still driving on the surface of Mars. It is thus extremely important to design grousers for optimum performance.

This thesis work will enable the exploration of various wheel and grouser design parameters through deterministic and reliability-based optimization techniques. Generally, the decisions regarding the shape of systems such as rovers are made during the conceptual design phase with the use of medium-fidelity tools. The proposed optimization framework will enhance the conceptual design calculations, thus eventually saving design costs and efforts.

1.2 Reliability Method

Reliability [6] is defined as the probability that a performance function g(X) is greater than zero. In other words, reliability is the probability that, while considering the uncertainties in the system, the performance function yields values in the safe region. Typically, safety is defined as the case when the performance function is positive, that is, $g(X) > 0$. One of the most commonly used reliability analysis methods is called First Order Reliability Method (FORM). In FORM, the

performance function g(X) is approximated by the first-order Taylor expansion about the most probable point (MPP). There are two steps in this method to take into consideration, the first step is to simplify the integrand, so that the contours become more regular and symmetric, while the second step is to approximate the integration boundary g(X) = 0. The Second Order Reliability Method (SORM) [7] uses the second-order Taylor expansion to approximate the performance function of the MPP. It is expected that the approximation of the performance function in SORM is more accurate than that in FORM. However, since SORM requires the second-order derivative, it is not as computationally efficient as FORM. If the number of performance functions is used for evaluations to measure efficiency, SORM needs more function evaluations than FORM.

FORM and SORM will be used to estimate reliabilities of the rover wheel. FORM will be used for reliability-based optimization.

1.3 Importance of Research

Most of the ongoing research on rover wheels focuses on straight or chevron-shaped grouser designs. The current designs of these grousers have been causing problems such as sinking, and getting stuck when traversing on loose soil. Also, most articles have found solutions to increase traction but have not found optimal solutions for a rover wheel design.

In the project the geometry of the rover wheel grousers will be optimized, which will significantly affect the traction on loose sandy surfaces. Reliability methods will also be used to calculate probabilities of failure events and further optimize the wheels while considering uncertainties in the design and non-design parameters. To decrease the probabilities of failure events, a reliability-based optimization method will be used.

1.4 Scope of Thesis

The thesis is split up into six different chapters and is explained below.

Chapter 2 discusses the literature review of the different grouser designs and their results, and a well-known reliability method is discussed. Relevant equations and relevant review of design parameters are also discussed.

Chapter 3 discusses the methodology of this thesis. This section examines the different methods being used to optimize the rover wheel. Those methods include structural optimization, reliability analysis method, and reliability-based optimization for two different cases.

Chapter 4 discusses the deterministic structural optimization formulation that was used for optimizing the rover wheels where no uncertainity is involved. This was done for two cases. The first case is to find the optimum values for the angle between two grousers and the height of the grousers. The second case is to find the optimum values for the angle between two grousers, the height of the grousers, and sinkage.

Chapter 5 discusses the preliminary results obtained by the First-Order and Second-Order Reliability methods for two different scenarios. The first scenario is when there are two uncertainties and the second scenario is when there are three uncertainties. The first case includes the empiric soil parameter, and the weight of the wheel as uncertainties. The second case includes the empiric soil parameter, the weight of the wheel, and the width of the wheel as uncertainties.

Chapter 6 discusses the reliability-based optimization results. Multiple optimization iterations were conducted until the probability of failure decreased.

Chapter 7 discusses the conclusion and future work.

2 Review of the Relevant Literature

Rovers are used to explore different planetary surfaces that humans are unable to reach. The most important quality of these rovers is mobility. Mobility is an essential technology for the successful navigation of rover missions on lunar and planetary surfaces.

2.1 Relevant Equations

For planetary rovers, there have been limited guidelines for wheel design. Inotsume et al. [8] address the design of rigid wheels for planetary rovers in loose granular soil. The key parameters for grouser design were grouser height; $\hat{h}$, sinkage; $\hat{z}$, and the angle between two successive grousers; $\emptyset$. Inotsume et al., ran two types of tests, the first test is doing a single and four wheeled experiment and the second uses the soil flow imaging technique. One of the criteria to evaluate the performance of a wheel they use is the traction efficiency formula. The traction efficiency formula indicates how efficiently the wheel can generate traction on the target soil. The traction efficiency is the ratio between the output and input tractive power of the wheel. Where F_x is the force, r is the radius of the wheel, T is the torque, and i_x is the slip ratio.

$$\eta = \frac{F_x r(1 - i_x)}{T} \tag{2.1}$$

By observing the soil flow from the soil imaging technique, a grouser design formula was derived by Skonieczny et. al [9]. The basic idea for the grouser design formula is that the resistive forward flow can be reduced if a grouser interacts with the soil before the wheel rim advances into the ground. Inotsume et al. observed increasing grouser height has a similar effect as an increase in wheel slip. Taller grousers create a longer gap between the wheel rim surface and the leading edge of the soil surface. With these observations and experimentations, a set of rough guidelines

were produced. In Eq 2.2, $\emptyset$ is the angle between two grousers, i_x is the slip ratio, $\hat{h}$ is the normalized height of grousers by the radius, and $\hat{z}$ is the normalized sinkage by the radius.

$$\emptyset = \frac{1}{1 - i_x}\left(\sqrt{\left(1 + \hat{h}\right)^2 - (1 - \hat{z})^2} - \sqrt{1 - (1 - \hat{z})^2}\right) \tag{2.2}$$

Buchele and Lichtenheldt [10], investigate wheeled locomotion in a milli-g environment and specifically the wheeled locomotion of a small rover. They considered several wheel geometry parameters and driving scenarios were simulated using the discrete element method. According to Buchele and Lichtenheldt, the wheel diameter and wheel width can be optimized. The wider the rims the weight of the rover can be better distributed on a larger surface and can help prevent soil failures underneath the wheel. Therefore, the rolling force formula incorporates the wheel dimensions and the empiric soil parameter to help prevent soil failures.

$$F_r = F_x = K_u\sqrt{\frac{w^3}{b4r^2}} \tag{2.3}$$

Here F_R is the rolling force, w is the weight of the wheel, b is the width of the wheel, r is the radius of the wheel, and K_u is the empiric soil parameter. Note that the empiric soil parameter changes with various locations. For example, Table 2.1 shows a list of soils with their empiric soil parameter [11].

Table 2.1 Empiric soil parameter for each soil type

Soil Type	Empiric Soil Parameter
Clay (Saturated)	0.66
Clay (Unsaturated)	0.34
Sandy clay	0.94
Sand (Dense)	0.3
Rock	0.2

The several types of grouser parameters Buchele and Lichtenheldt considered for optimization were the number of grousers, grouser radius, the height of the grousers, and the grousers curvature. During the simulation, they changed each of the parameters. They found that the grouser radius and grouser curvature have little influence on wheel sinkage. The number of grousers had a considerable influence on wheel locomotion capabilities while the slip and sinkage decreased, the grouser height had a large effect on traction. However, for the height of the grouser, they noticed getting an optimal value for grouser height was difficult. With increasing height, wheel slip can be reduced, and with taller grousers the surface contact increases. Increasing surface contact also means that wheel sinkage increases. Therefore, it is harder to find an optimum value for the height of the grousers.

Smith et al [12], conducted a numerical study using the discrete element method to investigate the performance and mobility impacts of rough terrain for small vehicles. Two distinct types of the wheel were used, one created by Ding and the other by NASA Mars exploration rover (MER). Smith and Peng did a digging simulation, a smooth soil simulation, and a rough soil simulation

validation. Each simulation has its own set of parameters that were entered into DEM. The values used for the simulation were fixed and the wheel properties for each wheel is shown in Table 2.2.

Table 2.2 Wheel properties

Parameter	**Ding**	**MER**
Radius (m)	0.157	0.125
Width (m)	0.165	0.16
Number of grousers	30	30
Grouser width (m)	0.002	0.0021
Torque (Nm)	10	10
∅ (º)	12	12
Sinkage ($\hat{z}$)	0.1	0.1
Weight of the wheel (kg)	8.155	8.155

2.2 Reliability Method

Reliability analysis using first order and second order methods is well-known. Cui et al. [13] propose a reliable design and optimization method of the planetary gears. They calculated the reliability by using the coefficient of variation. That method calculates the reliability by assuming the tangential forces are zero. To put this in real-world application, Cui et, al. uses DEM to simulate the gears and the soil particles to improve the calculation efficiency and compare it with the traditional design. They were able to decrease the failure rate by 17.05%, however they increased the volume of the structure by 36.96%. Peynot [14] proposed to validate a motion-planning approach and demonstrate the value of planning under uncertainty for safe and reliable navigation

on different terrain. The goal was to minimize time or distance and avoid obstacles. Song [15] studies a methodology of reliability prediction for reliability assessment that is developed by means of a case study on lunar rover wheels. Multiple techniques were applied, and optimal design values were obtained to identify the failure modes in the extreme environment on the Moon.

2.3 Gap in literature

In most of the reviewed literature, studies were done experimentally or with DEM, and much less emphasis on systematic optimization of grouser geometries. There are some references where empirical studies are performed to estimate the effect of changing a geometric parameter of the grouser wheels but not a vigorous optimization. Furthermore, there are very few references that report the effect of the uncertainties in the grouser simulation on the traction efficiency. There is no closed-form equation to estimate the traction efficiency that will help in conceptual design. These are the gaps this thesis attempts to fill through reliability-based optimization to improve the rover wheel traction efficiency. The goal of this paper is to use reliability-based optimization to reduce the failure rate of the grousers.

3 Methodology

Most of the ongoing research on rover wheels focuses on straight or chevron-shaped grouser designs. Also, most articles have found solutions to minimize the sinking of wheels and increase traction. However, these articles do not have the optimum solution for maximizing traction. Since the regolith is tough, the grousers should have good durability. Having good durability is important because the wheel can become damaged over time, which could potentially cause the mission to fail. The goal of this research focuses on optimizing traction by using the reliability method, deterministic optimization, and reliability-based optimization.

3.1 Research Approach

A few researchers studied tractive performance using numerical solutions. The objective of this thesis focuses on optimizing traction using the key parameters. First, using the key parameters a close-form equation is obtained. The formulas are then validated with previously done experimentation from the literature. There will be a total of three design variables. After verifying the key parameters and formulas, deterministic optimization will be conducted. The deterministic optimization does not have uncertain parameters. FORM and SORM will be used as reliability methods as preliminary results. In the equation, other variables can change and have uncertainties. The reliability method is conducted to consider those uncertainties. Finally, a reliability-based optimization will be used to solve an optimization problem with uncertainties to find the optimum values for a rover wheel design. The results are categorized into three different sections, the first result is obtaining a closed-form equation for traction efficiency, the second result is performing the reliability analysis, and finally the third result is using the reliability-based optimization to optimize a rover wheel design.

3.2 Closed-form equation

For this thesis, a close-form equation needed to be obtained to estimate the traction efficiency, which was not easily available. The first equation obtained was the traction efficiency equation from Inotsume (Eq 2.1). However, Inostume also mentions the key parameters as the angle between two grousers, the normalized height of the grouser, and the normalized sinkage, so the grouser design formula was obtained (Eq 2.2). Since both equations have a similar variable which is the slip ratio, the grouser design formula was substituted in the traction efficiency formula. Next, Buchele and Lichtenheldt created a formula that calculated the weight of the rover, which can be distributed on a larger surface. This also helps prevent soil failures underneath the wheel in terms of a rolling force and used in the traction efficiency formula (Eq 2.3). As a result, an equation that incorporates soil parameters and also the wheel design parameters were obtained. The normalized height of the grouser is $\hat{h} = \frac{h}{r}$ and the normalized sinkage $\hat{z} = \frac{z}{r}$, where r is the radius of the wheel. The closed-form equation obtained is:

$$\eta = \frac{180K_u}{2T\emptyset\pi}\left(\sqrt{\frac{w^3}{b}}\right)\left(\sqrt{\left(1+\hat{h}\right)^2-(1-\hat{z})^2}-\sqrt{1-(1-\hat{z})^2}\right) \tag{3.1}$$

The importance of this equation is that it will help us to perform and meet the objectives of finding out the effect of uncertainties and to perform optimization. This will be done without DEM or experimentation. Finally, this closed-form equation is obtained will be used for reliability analysis and then the rover wheel design.

3.2.1 Verification

To ensure the formulations are valid, Eq. 2.2 was plotted. The slip ratio versus traction efficiency and compared in Figure 3.1. This was graphed for wheel 31 (Wh31) [16], with a height of 15 mm. The values used for the traction formula was F = 20.35 N, T = 5.27 Nm, and r = 0.135 m. The graphs are similar however, the orange line was done experimentally and then plotted and the blue line was graphed by using the formulations (Eq 3.1). A close match between the two lines verifies the equation for the traction efficiency.

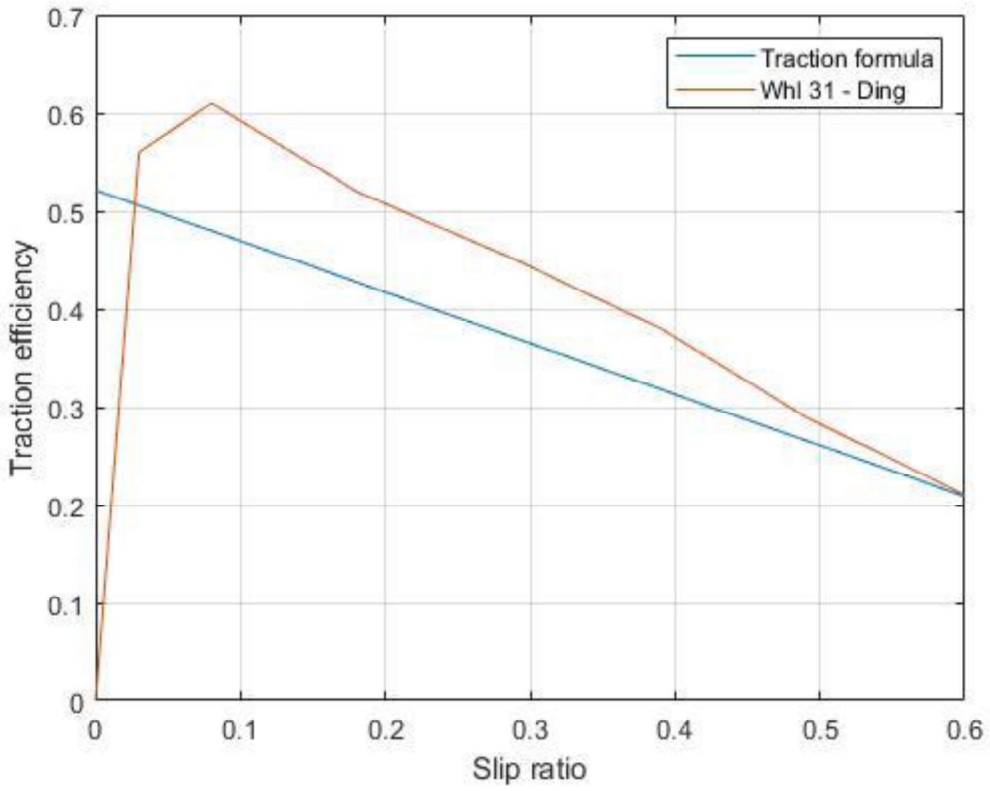

Figure 3.1 Comparison of Slip ratio vs Traction efficiency

3.3 Deterministic Optimization

Size optimization methods can be used for a given task to redesign the wheel grousers. This is done by varying the geometric wheel parameters into a feasible design space. The objective was to minimize traction efficiency subjected to a constraint on the efficiency. The closed-form traction efficiency equation is used to find the feasible region and the optimum point. The first case, the angle between two successive grousers and the height of the height of the grousers were optimized and the second case, the angle between two successive grousers, the height of the grousers, and the normalized sinkage were optimized. The three algorithms used were SQP, Interior-point, and Active-set and compared by graphing. The starting point for those three algorithms were (0,0,0) and the midpoint between the lower and upper bounds.

3.4 Reliability Analysis

For the First Order Reliability Method (FORM) [17], the performance function g(U) is expanded at the point that has the highest contribution to the probability of integration. It is preferable to expand the function at the point that has the highest value of the integrand, which is the highest probability density. The point that has the highest probability density on the performance g(U) = 0 is termed the Most Probable Point (MPP) and the function will be approximated at the MPP, which is expressed as

$$\begin{cases} min\|u\| \\ subject\ to\ g(u) = 0 \end{cases} \tag{3.2}$$

where $\|\cdot\|$ stands for the norm of a vector. The MPP is denoted by $u^* = (u_1^*,\ u_2^*, \ldots\ u_n^*)$. The minimum distance $\beta = \|u^*\|$ from the limit state g(U) = 0 to the origin in U-space is called the reliability index. The most used MPP search algorithm uses a recursive formula and is based on

the linearization of the performance function. The MPP kth iteration is u^k and is linearized at that point. The linearized function becomes

$$g(u) = g(u^k) + \nabla g(u^k)(u - u^k)^T \tag{3.3}$$

The next iteration is u^{k+1}line, which becomes

$$g(u^{k+1}) = g(u^k) + \nabla g(u^k)(u^{k+1} - u^k)^T = 0 \tag{3.4}$$

Where the reliability index becomes $u^k = -\beta^k a^k$ and the next iteration reliability index becomes $u^{k+1} = -\beta^{k+1} a^k$. By substituting Eq. 3.2 and Eq. 3.3 it becomes

$$g(u^k) + \nabla g(u^k)(a^k)^T(\beta^k - \beta^{k+1}) = \nabla g(u^k) + \|\nabla g(u^k)\|(\beta^k - \beta^{k+1}) = 0 \tag{3.5}$$

By rearranging

$$\beta^{k+1} = \beta^k + \frac{g(u^k)}{\|\nabla g(u^k)\|} \tag{3.6}$$

Therefore, the updated point becomes

$$u^{k+1} = -a^k \left\{ \beta^k + \frac{g(u^k)}{\|\nabla g(u^k)\|} \right\} \tag{3.7}$$

The MPP vector u^* is perpendicular to the curve g(U) = 0. Therefore, the direction of the gradient can be expressed as $u^* = -\beta a$. Hence, the probability of function is evaluated by

$$p_f = P\{L(U) = 0\} = \Phi(au^{*T}) = \Phi(-\beta aa^T) = \Phi(-\beta) \tag{3.8}$$

where $aa^T = \sum_{i=1}^{n} \alpha_i^2 = 1$

Therefore, the reliability is found as

$$R = 1 - p_f = 1 - \Phi(-\beta) = \Phi(\beta) \tag{3.9}$$

For the Second Order Reliability Method (SORM) [17], the approximation is given by

$$g(U) = q(U) = g(u^*) + \nabla(u^*)(U - u^*)^T + \frac{1}{2}(U - u^*)H(u^*)(U - u^*)^T \tag{3.10}$$

where $H(u^*)$ is the Hessian matrix at the MPP,

$$H(u^*) = \begin{bmatrix} \frac{\partial^2 g}{\partial U_1^2} & \frac{\partial^2 g}{\partial U_1 U_2} & \frac{\partial^2 g}{\partial U_1 U_n} \\ \frac{\partial^2 g}{\partial U_2 U_1} & \frac{\partial^2 g}{\partial U_2^2} & \frac{\partial^2 g}{\partial U_2 U_n} \\ \frac{\partial^2 g}{\partial U_n U_1} & \frac{\partial^2 g}{\partial U_n U_2} & \frac{\partial^2 g}{\partial U_n^2} \end{bmatrix} \tag{3.11}$$

After the linear transformations are completed, the performance function is simplified as

$$q(U) = U_n - \left(\beta + \frac{1}{2} U'^T D U^T\right) \tag{3.12}$$

where D is a $(n-1) * (n-1)$ diagonal matrix whose elements are determined by the Hessian matrix and the U-space. When β is large enough, an asymptotic solution of the probability of failure is derived as

$$p_f = P\{g(X) < 0\} = \Phi(-\beta) \prod_{i=1}^{n-1} (1 + \beta k_i)^{1/2} \tag{3.13}$$

Which k_i denotes the i-th main curvature of the performance function g(U) at the MPP.

To validate the values of FORM and SORM, Monte-Carlo simulation (MCS) was conducted to predict the probability of different outcomes. MCS is used to estimate the possible outcome of an uncertain event. In FORM and SORM there are two to three uncertain variables. Once MCS was run, SORM and FORM were compared to each other. This paper uses reliability analysis for the rover case.

3.5 Reliability-Based Optimization

The reliability-based optimization is used to account for the randomness in structural geometry, material properties, etc. The purpose of a reliability-based optimization is to decrease the probability of failure through each iteration. Stochastic optimization considers the randomness or uncertainty in the data. The process for stochastic optimization is to use upper and lower bounds for each variable and the partial derivatives to find the optimum values for each uncertain variable. The process for reliability-based optimization is shown in Figure 3.2.

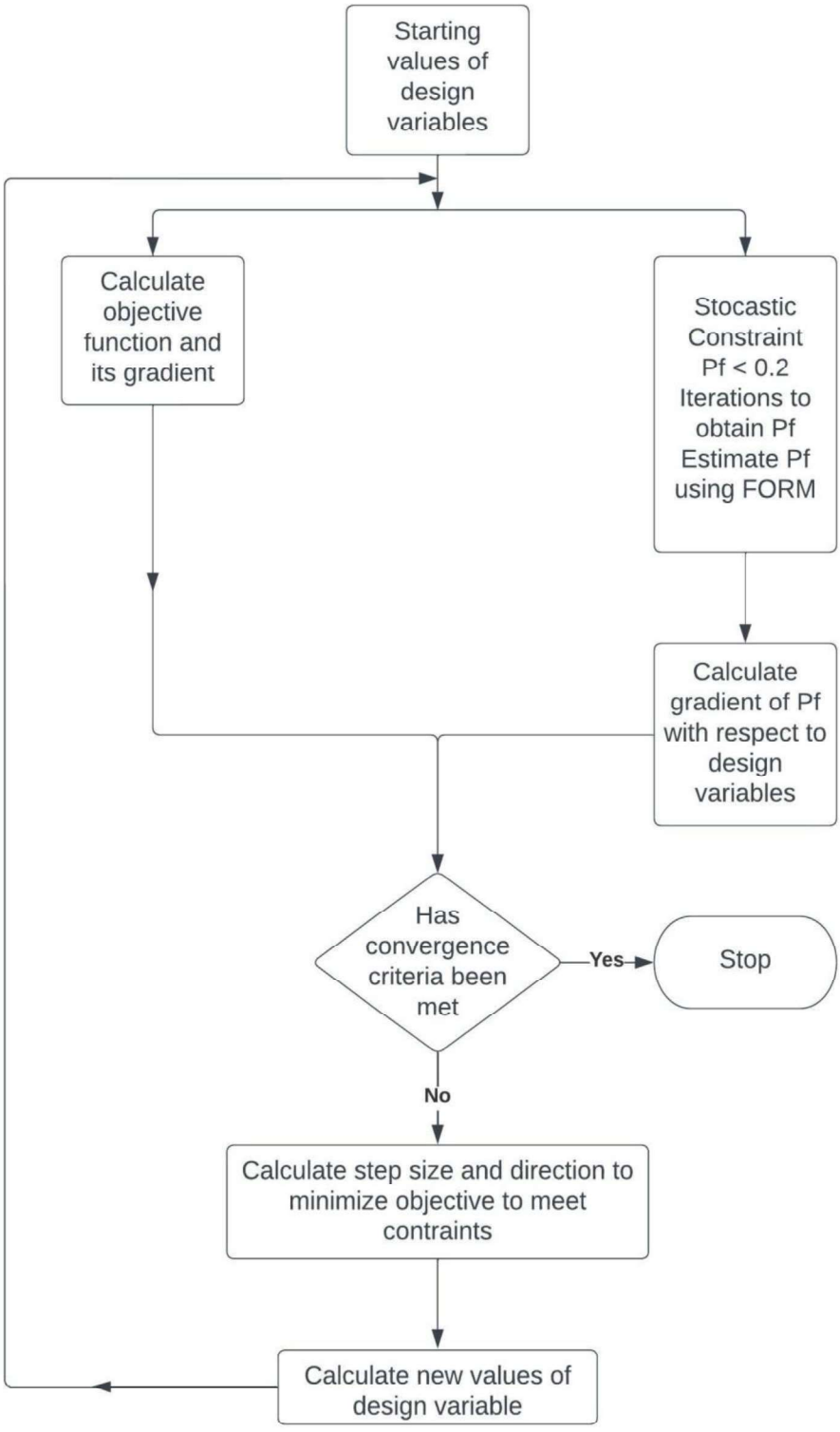

Figure 3.2 Flow chart

4 Deterministic Optimization

In this section, MATLAB was utilized to get the optimum design using deterministic optimization.

4.1 Preliminary results for straight grousers

A literature review reveled that the sinkage of a wheel should be between 5% and 15% of the radius of the wheel so that it can be used as a safety factor for traction efficiency. This was done twice, once for two design variables and another for three design variables. For optimization, the design variables were the number of grousers and the height of the grousers. The codes used for this optimization are given in appendix A1.

For the two-design variable optimization, the sinkage was kept at a constant value of 10% of the radius. The formulation for this is given below:

Minimize:

$$\eta = -\frac{180K_u}{2Tx_1\pi}\left(\sqrt{\frac{w^3}{b}}\right)\left(\sqrt{(1+x_2)^2-(1-0.1)^2}-\sqrt{1-(1-0.1)^2}\right) \tag{4.5}$$

Subjected to:

$$57 \leq x_1 \leq 360 \tag{4.6}$$

$$0.1 \leq x_2 \leq 0.157 \tag{4.7}$$

Here x_1 is the angle between grousers and x_2 is the height of the grousers.

The graphical solution for the feasible region is given in Figure 4.1. The optimum point for the graphical solution can be seen in Figure 4.2. The starting point for each method was at the origin, which is x = (0,0). The results for the SQP, Active Set, and Interior Point, the algorithm can be

seen in Figure 4.3. The codes used for this optimization are given in appendix A2. All algorithms converge to the same optimum point.

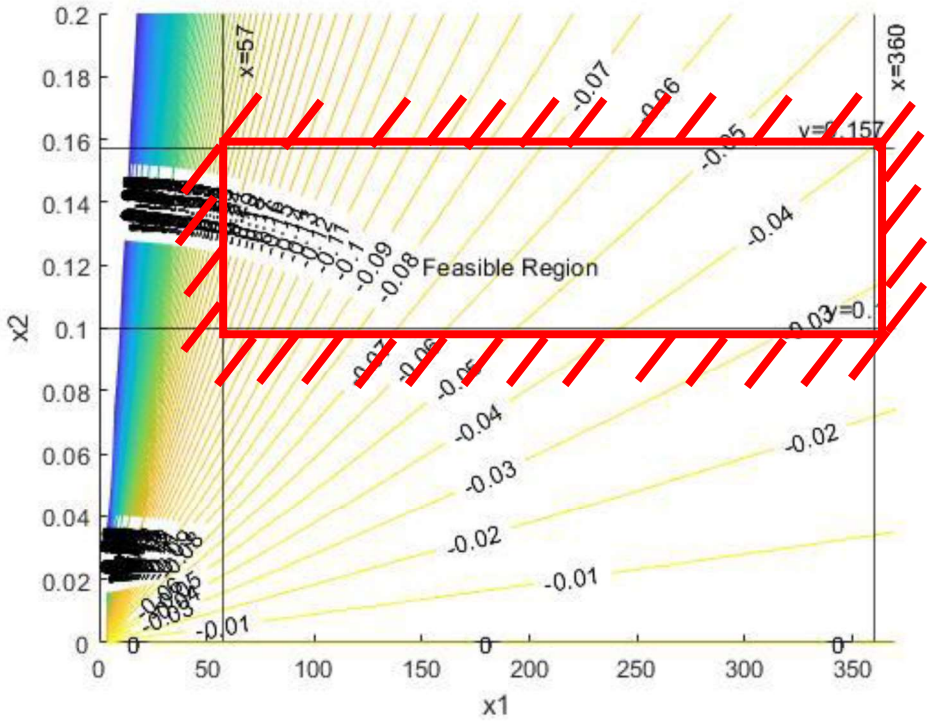

Figure 4.1 Graphical solution: feasible region

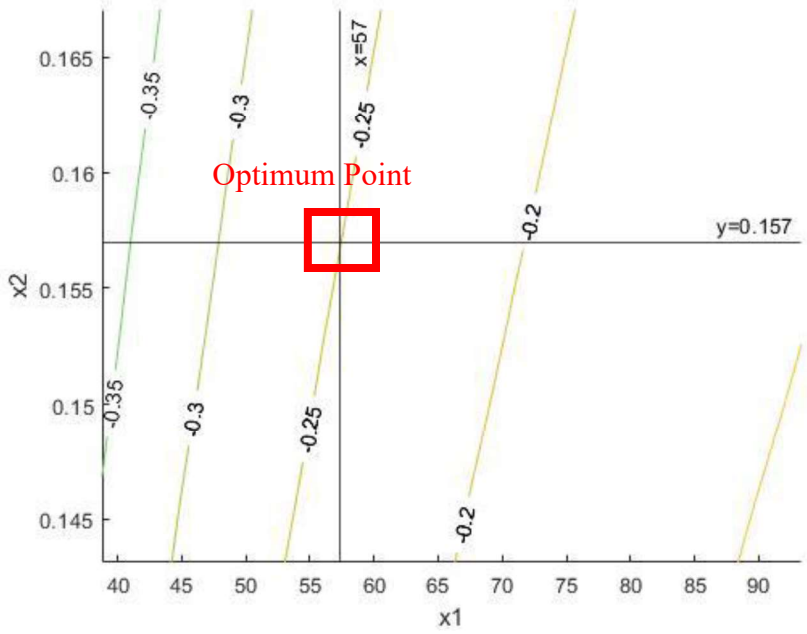

Figure 4.2 Graphical solution: optimum point

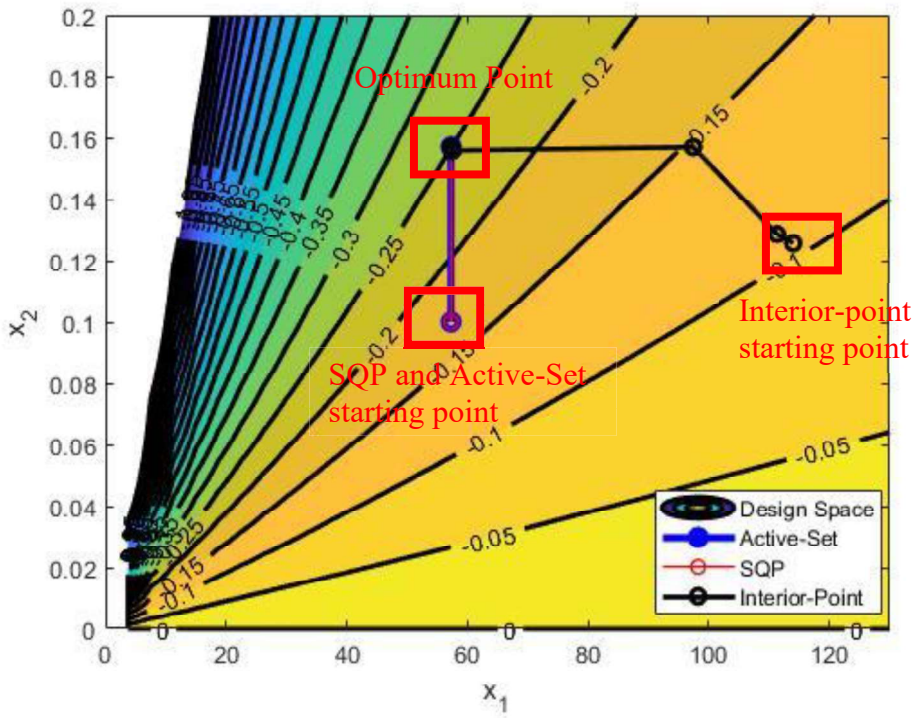

Figure 4.3 Comparison of Active-Set, SQP, and Interior-Point

SQP and Active Set had the same starting position. For Interior Point the starting point was changed to be the midpoint between the upper bound and lower bound constraints. Hence, the starting point was (209, 0.1285). The MATLAB code ran again, and the results are seen in Figure 4.4. Table 4.1 shows the comparison between them with a starting point of (0,0), and Table 4.2 shows the comparison with a starting point of (209, 0.1285).

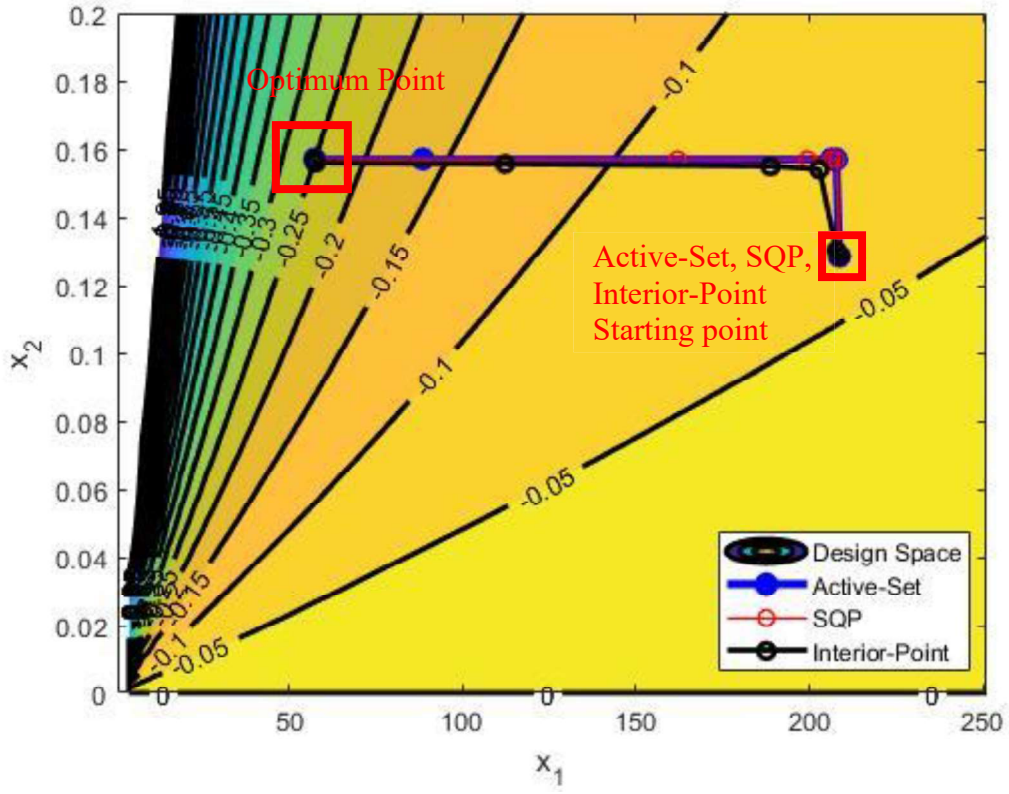

Figure 4.4 Comparison of methods with a starting point of (209, 0.1285)

Table 4.1 Summary of convergences values for starting point (0,0)

Algorithm	**Convergence values**			**Number of Iterations**
	$-\eta$	**$\emptyset$ (rad)**	**$\hat{h}$**	
MATLAB Graphical Solution	-0.2504	1	0.157	----
MATLAB fmincon: Interior Point	-0.254172	1	0.157	15
MATLAB fmincon: SQP	-0.2504180	1	0.157	4
MATLAB fmincon: Active Set	-0.250418	1	0.157	2

Table 4.2 Summary of convergences values for starting point *(3.6416, 0.1285)*

Algorithm	**Convergence values**			**Number of iterations**
	$-\eta$	**$\emptyset$ (rad)**	**$\hat{h}$**	
MATLAB Graphical Solution	-0.2504	1	0.157	----
MATLAB fmincon: SQP	-0.2504180	1	0.157	7
MATLAB fmincon: Active Set	-0.250418	1	0.157	5
MATLAB fmincon: Interior Point	-0.2504172	1	0.157	18

After the optimization of the two design variables, the next step was to use all three of the key parameters. The constraints for sinkage were still between 5% and 15% of the radius. The formulation is shown below:

Minimize:

$$\eta = -\frac{180K_u}{2Tx_1\pi}\left(\sqrt{\frac{w^3}{b}}\right)\left(\sqrt{(1+x_2)^2-(1-x_3)^2}-\sqrt{1-(1-x_3)^2}\right) \tag{4.8}$$

Subjected to:

$$57 \leq x_1 \leq 360 \tag{4.9}$$

$$0.1 \leq x_2 \leq 0.157 \tag{4.10}$$

$$0.05 \leq x_3 \leq 0.15 \tag{4.11}$$

Here x_1 is the angle between two grousers in degrees, x_2 is the height of the grousers, and x_3 is the normalized sinkage of the wheel.

The above formulation was then put into MATLAB to find the optimum solution. The codes used for this optimization are given in appendix A3. The starting point for each method was at the origin, which is x = (0,0,0). After running SQP, Active Set, and Interior Point, they were all put into a graph for comparison which can be seen in Figure 4.5.

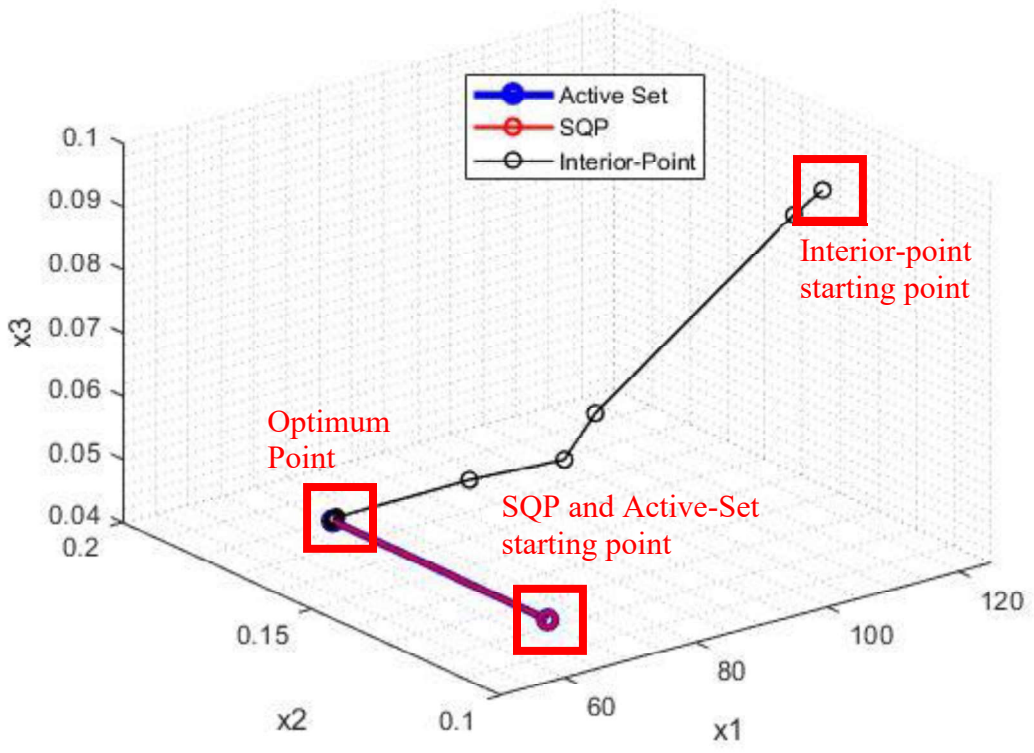

Figure 4.5 Comparison of methods with an initial starting point

SQP and Active Set had the same starting position. For Interior Point was changed to be the midpoint between the upper bound and lower bound constraints. Hence, the starting point ends up being (139, 0.085667, 0.0667). The MATLAB code ran again, and the results can be seen in Figure 4.6. Table 4.3 shows the comparison between them with a starting point of (0,0,0), and *Table 4.4* shows the comparison with a starting point of (139, 0.085667, 0.0667).

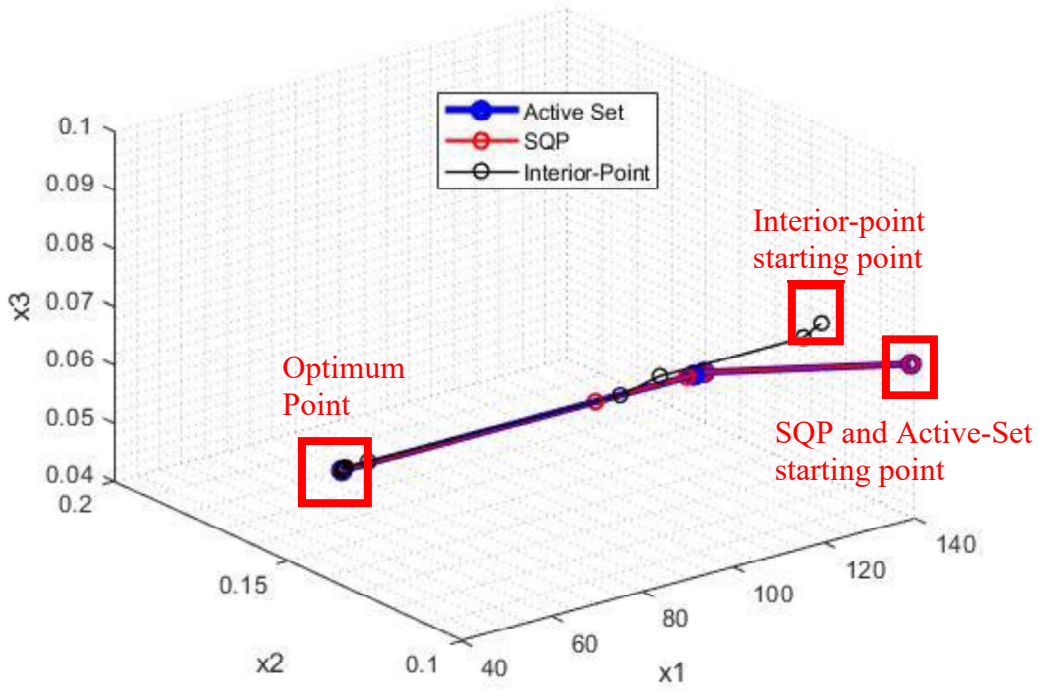

Figure 4.6 Comparison of methods with a starting point of (139, 0.085667, 0.0667)

Table 4.3 Summary of convergences values for starting point (0,0,0)

Algorithm	**Convergence values**				**Number of Iterations**
	$-\eta$	**$\emptyset$ (rad)**	**$\hat{h}$**	**$\hat{z}$**	
MATLAB fmincon: Interior Point	-0.2994123	1	0.157	0.05	21
MATLAB fmincon: SQP	-0.2994135	1	0.157	0.05	4
MATLAB mincon: Active Set	-0.299414	1	0.157	0.05	2

Table 4.4 Summary of convergences values for starting point (2.427, 0.085667, 0.0667)

Algorithm	**Convergence values**				**Number of Iterations**
	$-\eta$	**$\emptyset$ (rad)**	**$\hat{h}$**	**$\hat{z}$**	
MATLAB fmincon: Interior Point	-0.2994135	1	0.157	0.05	24
MATLAB fmincon: SQP	-0.2994135	1	0.157	0.05	6
MATLAB mincon: Active Set	-0.299414	1	0.157	0.05	4

It was realized that sinkage was not a good parameter to optimize because sinkage cannot be a variable to design for. Therefore, the design parameters will be chosen carefully for the upcoming case studies.

5 Reliability Analysis

Some parameters in the simulation of rover wheels may have uncertainties for example, soil parameter, K_u, due to different soil types, torque, T, due to signal processing error, weight of the wheel, w, due to manufacturing and the amount of sensors on the wheel, the width of the wheel, b, due to manufacturing and tolerancing The purpose of using reliability analysis method, is to understand the effects of the uncertainty on traction efficiency.

First-Order Reliability Method (FORM) will be applied to find the reliability of the traction with two and three uncertainties. The limit state function is the difference between traction efficiency and η_0, where η_0 is the lower limit of the expected efficiency is given by g, Eq 5.1.

$$g = \frac{180K_u}{2T\emptyset\pi}\left(\sqrt{\frac{w^3}{b}}\right)\left(\sqrt{\left(1+\hat{h}\right)^2-(1-\hat{z})^2}-\sqrt{1-(1-\hat{z})^2}\right)-\eta_0 \tag{5.1}$$

The probability of failure is defined as the probability of the allowable tractive efficiency less than η_0.

$$p_f = P\{g < 0\} \tag{5.2}$$

In general, a failure event for a rover wheel could be that the wheel does not have enough force to pull the rover out if the wheel gets stuck. A few more events could be the wheel sinking into the soil more than it should. If the rover has solar panels, like Spirit, the sand could cover the solar panels so that enough light cannot be absorbed this causes the wheels to lose torque due to the lack of power being produced. We define that a failure event occurs if the efficiency drops below 0.2.

After using FORM, the same Most Probably Point (MPP) is used for SORM to find the probability of failure. To compare the accuracy of the results between FORM and SORM, Monte Carlo Simulation (MCS) is used.

5.1 Two Uncertainties

The first two uncertainty variables are the empiric soil parameter, Ku, and the weight of the wheel, w. The fixed values for torque, width of the wheel, and the angle between two grousers are taken from Table 2.2 and Table 5.1 shows the mean and standard deviation of the uncertain parameters with the normal distribution. The codes used for this optimization are given in appendix A4.

Table 5.1 Normal distribution for two uncertainties

Variable	Mean Value	Standard Deviation	Possible reason for uncertainty
$\boldsymbol{K_u}$	0.3	0.03	Different soil types
w	8.155	1	The number of sensors on the wheels, manufacturing, tolerancing

The limit function, g(s), is given below and the gradient of the transformed function, ∇g(s), is found. The values of ∅ and h are retrieved from Table 4.2.

$$g(s) = \frac{180 s_1}{2T\emptyset\pi}\left(\sqrt{\frac{s_2^3}{b}}\right)\left(\sqrt{\left(1+\hat{h}\right)^2-(1-\hat{z})^2}-\sqrt{1-(1-\hat{z})^2}\right)-\eta_0 \tag{5.3}$$

The gradient of the limit function is with respect to s_1, s_2. The starting point of the MPP is set to $u^0 = (0,0)$. As a result, the search determined there were a total of 2 iterations. Table 5.2 shows the MPP search history and Figure 5.1 shows those results graphed until convergence. The transformed standard variable of (U_x, U_y) is a new point which will be used in the next iteration.

Table 5.2 Most Probably Point search history for two uncertainties

Iteration	**β**	**g**	**∇g**	**(U_x, U_y)**
1	-0.4492	-2	(70.8888, 3.9117)	(0.2146,0.3946)
2	-0.8760	-2	(76.0962, 4.0912)	(0.4269, 0.7650)

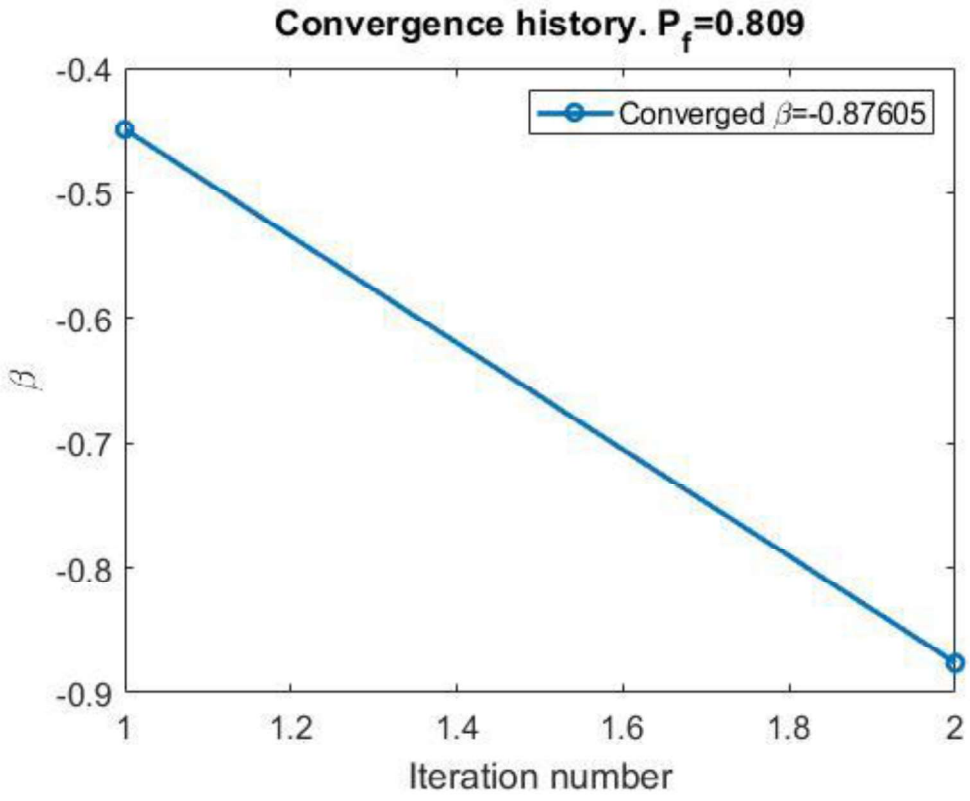

Figure 5.1 Convergence history for two uncertainties

The MPP has found that at $u^* = (0.4269, 0.7650)$, and the reliability index is $\beta = -0.8760$. The probability of failure is

$$p_f = \Phi(0.8760) = 0.8095$$

The reliability is

$$R = 1 - p_f = 1 - 0.8095 = 0.19050$$

Therefore, the tractive efficiency with the given inputs from Table 5.1 is about 19% reliable. After using FORM, the second derivative was found for each variable. The second derivative was used in SORM as elaborated in section 3.4. The Monte Carlo simulation (MCS) was performed with 100 samples for each uncertain parameter. The results of the probability are compared in Table 5.3.

Table 5.3 Probability of failure from different methods for two uncertainties

Method	**FORM**	**SORM**	**MCS**
$\boldsymbol{P_f}$	0.8095	0.81354	0.81495
Time (s)	0.1135	0.1205	8.389

5.2 Three Uncertainties

The three uncertainty variables considered are the empiric soil parameter, Ku, the weight of the wheel, w, and the width of the wheel, b. The fixed value for torque is taken from Table 2.2, and Table 5.4 which shows the mean and standard deviation with the normal distribution. The codes used for this optimization are given in appendix A5.

Table 5.4 Normal distribution for three uncertainties

Variable	Mean Value	Standard Deviation	Possible reason for uncertainty
$\boldsymbol{K_u}$	0.3	0.03	Different soil types
w	8.155	1	The number of sensors on the wheels, manufacturing tolerance
b	0.165	0.0165	Manufacturing tolerance

The limit function, g(s) is given below and the gradient of the transformed function, ∇g(s), is found. The values of $\emptyset$, $\hat{h}$, and $\hat{z}$ are retrieved from Table 4.4.

$$g(s) = \frac{180 s_1}{2T\emptyset\pi}\left(\sqrt{\frac{s_2^3}{s_3}}\right)\left(\sqrt{\left(1+\hat{h}\right)^2 - (1-\hat{z})^2} - \sqrt{1-(1-\hat{z})^2}\right) - \eta_0 \tag{5.4}$$

The gradient of the transformed function is with respect to s_1, s_2, s_3. The starting point of the MPP is set to $u^0 = (0,0,0)$. As a result, the search determined there were a total of 6 iterations. However, the solution for iterations 5 and 6 is similar to iteration 4. Table 5.5 shows the MPP

search history and Figure 5.2 shows those results graphed until convergence. The transformed standard variable of (U_x, U_y, U_z) is a new point which will be used in the next iteration.

Table 5.5 Most Probable Point search history for three uncertainties

Iteration	**β**	**g**	**∇g**	**(U_x, U_y, U_z)**
1	0.9865	0.5392	(8.4640, 0.4670, -7.6945)	(-0.4583, -0.8430, 0.2292)
2	1.0566	0.0339	(7.1051, 0.4172, -6.0251)	(-0.4702, -0.9204, 0.2193)
3	1.0568	0.0001	(6.9960, 0.4147, -5.9309)	(-0.4670, -0.9227, 0.2177)
4	1.0568	0.0000	(6.9932, 0.4148, -5.9315)	(-0.4667, -0.9228, 0.2177)
5	1.0568	0.0000	(6.9931, 0.4148, -5.9315)	(-0.4667, -0.9228, 0.2177)
6	1.0568	0.0000	(6.9930, 0.4148, -5.9315)	(-0.4667, -0.9228, 0.2177)

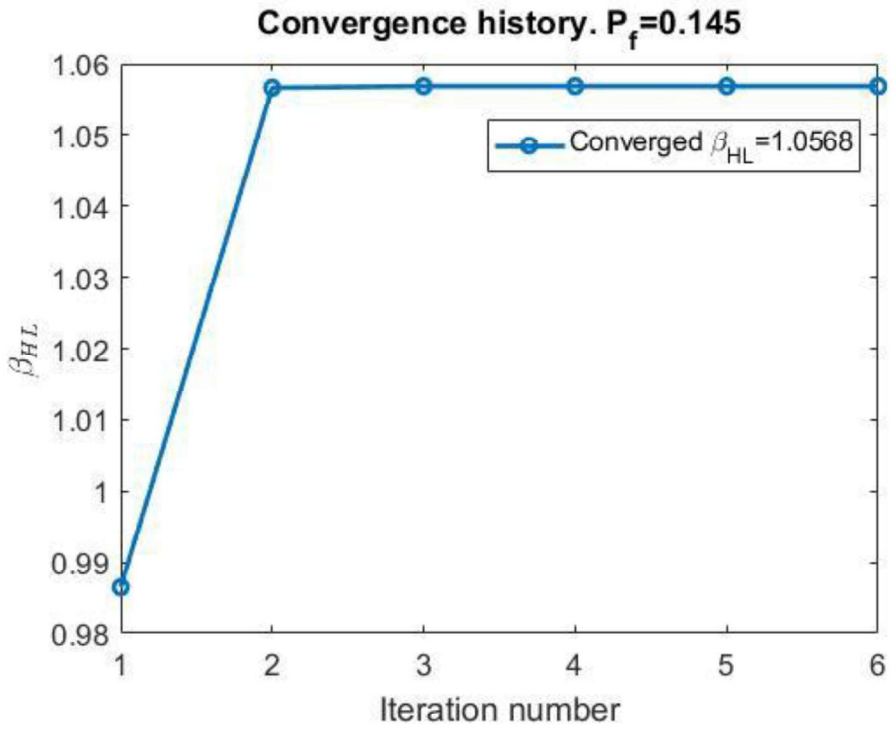

Figure 5.2 Convergence history for three uncertainties

The MPP has found that at $u^* = (-0.4667, -0.9228, 0.2177)$, and the reliability index is $\beta = 1.0568$. The probability of failure is

$$p_f = \Phi(-1.0568) = 0.14530$$

The reliability is

$$R = 1 - p_f = 1 - 0.14530 = 0.85470$$

Therefore, the tractive efficiency with the given inputs from Table 5.4 is about 85% reliable. After using FORM, the second derivative was found for each of the three variables. The second derivative was used in SORM. The results of the probability are compared in Table 5.6 and it is seen that traction efficiency goes below 0.2 about 15% of the time due to the uncertainties in empiric soil parameter, the weight of the wheel, and the width of the wheel. The time needed for FORM and SORM to calculate is only a fraction when compared to MCS. For MCS 100 samples were taken for each uncertain parameter.

Table 5.6 Probability of failure from different methods for three uncertainties

Method	**FORM**	**SORM**	**MCS**
$\boldsymbol{P_f}$	0.14530	0.14136	0.15075
Time (s)	0.1206	0.1262	7.2849

6 Reliability-Based Optimization

After understanding how the certain parameters affect traction efficiency, the goal of chapter 6 is to design a rover wheel while considering the uncertain parameters.

After obtaining a closed-form equation for traction efficiency and running the reliability analysis to understand how the uncertainties affect traction efficiency, the key parameters grouser design was chosen as the width of the wheel, the radius of the wheel, and the normalized height of the grouser, as shown in Figure 6.1.

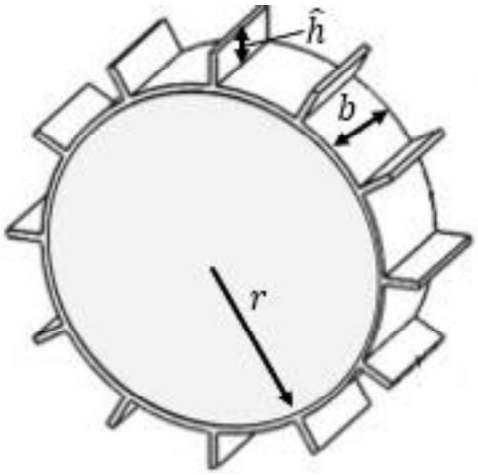

Figure 6.1 Design parameters

Volume of the rover wheels, given in Eq 6.1, was minimized subjected to a constraint on the probability of failure (in case of reliability optimization) or a constraint on the value of efficiency (in case of deterministic optimization).

$$V = \pi br^2 + 360br\hat{h}t \tag{6.1}$$

The deterministic optimization has b, r, and $\hat{h}$ as the design variables and has an upper and lower bound limit. The thickness of the wheel is obtained from Table 2.2. Table 6.1, shows the initial design variables and their initial values with upper and lower bounds. The optimization problem is defined in Eq 6.2-6.6.

Table 6.1 Initial design parameters

Initial design parameters	**Initial points**	**Deterministic optimization**	**Reliability-based optimization**
		(Lower, Upper) bound	**(Lower, Upper) bound**
Normalized height of grouser, $\hat{h}$	0.55	(0.1, 1)	(0.1, 1)
Radius of the wheel, r (m)	0.175	(0.15, 0.2)	(0.15, 0.2)
Width of the wheel, b (m)	0.165	(0.112, 0.218)	(0.132, 0.198)

For reliability-based optimization, three design parameters that are considered to be uncertain. The two parameters that are not design variables are the empiric soil parameter, K_u and torque, T. The design variable and is considered uncertain is the width of the wheel, b. The uncertain empiric soil parameter mimics the changes with different locations; the uncertain torque is related to power variability as well as signal processing error, sensor or actuator noise; and the uncertain width of the wheel is related to manufacturing tolerances for the wheel. Table 6.2 shows the uncertain parameters with their mean value and standard deviation for reliability-based optimization

Table 6.2 Uncertain variables

Uncertain variables	Mean value	Standard deviation	Possible reason for uncertainty
Soil parameter, K_u	0.3	0.03	Change in soil in varies locations
Torque, T (Nm)	60	6	Power variability, signal processing error, sensor or actuator noise
Width of the wheel, b (m)	0.165	0.0165	Manufacturing tolerances

The deterministic optimizations upper and lower bound limits are not as tight as the reliability-based optimization because the bounds for the reliability-based optimization are on the mean values. The uncertain variable that has the tighter bounds is the width of the wheel. The goal is to minimize the volume. The optimization problem is defined in Eq 6.7-6.11. For the deterministic optimization the constraint is that the traction efficiency is greater than 0.5 as seen in Eq 6.2 and the reliability-based optimization has a constraint for the probability of failure to be less than 10% as seen in Eq 6.7. The purpose of minimizing the volume of the wheel is to keep the wheel small in size because it is better for handling and maneuvering on the terrain. Figure 6.2 and Figure 6.3 displays the iteration history for deterministic optimization and reliability-based optimization respectively. Figure 3.2 shows a flow chart on how the iterations are carried out. Table 6.3 shows the results by comparing deterministic optimization and reliability-based optimization. The codes for this reliability-based optimization are given in appendix A6, appendix A7, appendix A8, and appendix A9.

Deterministic optimization

Minimize

$$\mathrm{V} = \pi x_1 x_2^2 + 360 x_1 x_2 x_3 t \tag{6.2}$$

Subjected to

$$\eta > 0.5 \tag{6.3}$$

and

$$0.112 < x_1 < 0.218 \tag{6.4}$$

$$0.15 < x_2 < 0.2 \tag{6.5}$$

$$0.1 < x_3 < 1 \tag{6.6}$$

where η is calculated as in Eq 4.5-4.7 and Eq 4.8-4.11. Here x_1 is width of the wheel, b, x_2 is the radius of the wheel, r, and x_3 is the normalized height of the wheel.

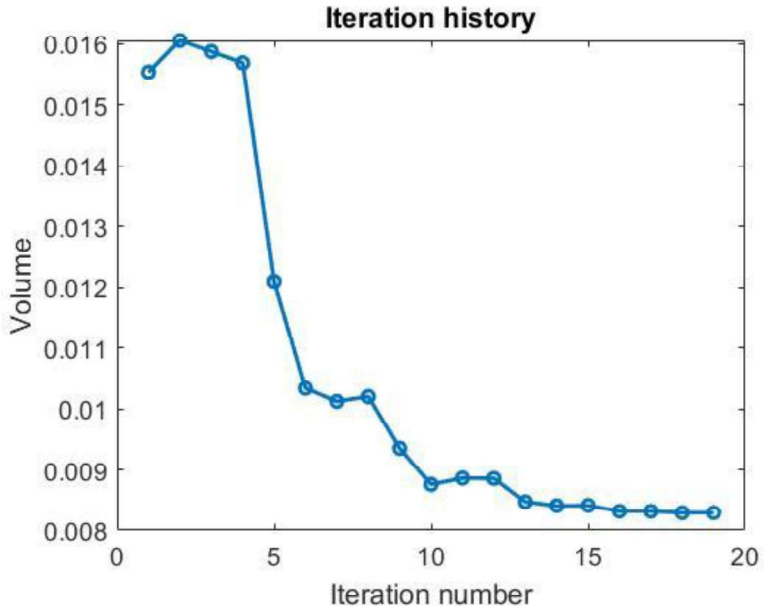

Figure 6.2 Iteration history for deterministic optimization

Reliability-based optimization

Minimize

$$V = \pi x_1 x_2^2 + 360 x_1 x_2 x_3 t \tag{6.7}$$

Subjected to

$$P_f < 0.1 \tag{6.8}$$

and

$$0.132 < x_1 < 0.198 \tag{6.9}$$

$$0.15 < x_2 < 0.2 \tag{6.10}$$

$$0.1 < x_3 < 1 \tag{6.11}$$

where P_f is obtained from FORM as explained in section 3.4 and calculated in Eq 5.2. Here x_1 is width of the wheel, b, x_2 is the radius of the wheel, r, and x_3 is the normalized height of the wheel.

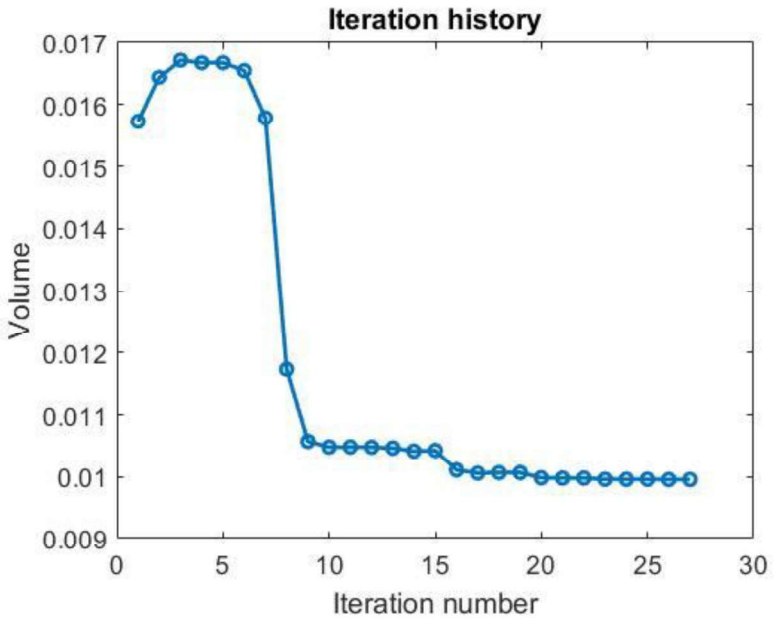

Figure 6.3 Iteration history for reliability-based optimization

Table 6.3 Comparison between deterministic and reliability-based optimization

Parameters	Deterministic Optimization	Reliability-Based Optimization
$\boldsymbol{\eta}$	0.5001	0.6098
b (m)	0.1120	0.1320
r (m)	0.1500	0.1500
$\widehat{\boldsymbol{h}}$	0.3738	0.5265
V ($\boldsymbol{m^3}$)	0.0083	0.01
$\boldsymbol{P_f}$	----	7.4455E-5
Time (sec)	0.571939	0.8157

In conclusion, the reliability-based optimization results in a larger $\hat{h}$ and b but it ensures a lower probability of failure occurrence. The width of the wheel and the radius of the wheel lean more towards the lower bounds, while the normalized height of the grousers lean towards the middle of the bounds. Also, the reliability-based optimization takes about 0.8 seconds while the deterministic optimization takes about 0.6 seconds. Again, the normalized height of the grouser, $\hat{h}$, was a bit difficult to get an optimal value because as the height increases, wheel slip decreases, and surface contact increases. However, if the grouser height is too tall then wheel sinkage increases.

7 Conclusion and Future Work

When rovers are sent to the Moon or Mars, multiple factors could have a negative effect on the wheel such as the amount of torque needed to let the rover keep moving and the size of the regolith. In different areas of the planet the material properties of the terrain changes. To this end, reliability-based optimization was performed to ensure that the uncertainties in certain important parameters of the simulation (e.g., the soil parameter) are considered while optimizing the grouser geometries. After running two cases it was seen that sinkage was not a good parameter to be optimized. Sinkage is just a value that is a result of the design parameters. Therefore, sinkage was set to the desired value of 10% of the radius of the wheel, and optimum values were found at that desired value of sinkage.

7.1 Conclusion

Novelty of the presented research on rover wheels is in two areas: (a) obtaining a closed-form equation for rover traction efficiency that can be used for reliability analysis and conceptual design, and (b) reliability-based optimization of rover wheels with more than 3 uncertain parameters (one of which is also a design variable) and with the use of a stochastic constraint.

Design of rover wheels relies on models based either experimentation or use of computationally expensive analysis such as the discrete element method (DEM). As an alternative to these methods, we propose to use a simple closed-form equation for rover traction efficiency. This equation was obtained by combining two separate equations: one for the efficiency based on the applied torque and slip ratio, and the other related to the slip ratio and the angle between the wheel grousers. This equation was verified for its accuracy by comparing against previous studies.

Further, we calculate the effect of uncertainty in the rover traction efficiency arising due to different sources such as soil types (which changes the empiric soil parameter value), sensor and actuator noise (which affects the applied torque), and manufacturing tolerances (which affect the

value of the weight of the wheel). This uncertainty in the traction efficiency was obtained using the first and second order reliability methods (FORM and SORM) and was verified using the Monte-Carlo simulations (MCS). It was seen that the FORM and SORM methods can estimate the probabilities of failure reasonably well and much efficiently than MCS.

Finally, we set up a reliability-based optimization to minimize the volume of the rover wheel while constraining the probability of failure. In essence, the optimization problem included a stochastic constraint on the value of the traction efficiency. To make the problem realistic, we choose 3 design variables out of which 1 design variable (weight of the wheel) was considered uncertain (to simulate the manufacturing tolerance errors). Also, there were 2 other parameters which were considered to be uncertain but were not design variables.

The reliability-based optimization was compared with a deterministic optimization wherein the constraint was set on a value of the efficiency. It was found that while the reliability-based optimization resulted in a larger volume (that is higher heights of the grousers), it did ensure that the probability of failure is within the stipulated bounds.

7.2 Future Work

The current approach of reliability-based optimization of rover wheels, although efficient, relies on a simple equation of traction efficiency. The same optimization formulation can be extended to use DEM instead of the simple equation. This can be done using tools such as EDEM and LIGGGHTS.

8 REFERENCES

[1] Favaedi, Yalda. Prediction of tractive response for flexible wheels with application to planetary exploration rovers. Diss. University of Surrey, 2010

[2] NASA. (2019, July 16). In depth. NASA. Retrieved July 9, 2022, from https://solarsystem.nasa.gov/missions/spirit/in-depth

[3] NASA. (2019, July 16). In depth. NASA. Retrieved July 9, 2022, from https://solarsystem.nasa.gov/missions/spirit/in-depth

[4] NASA. (2021, February 11). In depth. NASA. Retrieved July 9, 2022, from https://solarsystem.nasa.gov/missions/curiosity-msl/in-depth/

[5] NASA. (2021, September 29). In depth. NASA. Retrieved July 9, 2022, from https://solarsystem.nasa.gov/missions/mars-2020-rover/in-depth/

[6] Kulkarni, Mandar., and Canfield, Robert. "Reliability Based Structural Design using Continuum Sensitivity Analysis." 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. 2015.

[7] Zhang, Junfu., and Xiaoping, Du. "A second-order reliability method with first-order efficiency." (2010): 101006.

[8] Inotsume, Hiroaki., Moreland, Scott., Skonieczny, Krzysztof., and Wettergreen, David. "Parametric study and design guidelines for rigid wheels for planetary rovers." Journal of Terramechanics 85 (2019): 39-57.

[9] Skonieczny, Krzysztof., Moreland, Scott., and Wettergreen, David. "A grouser spacing equation for determining appropriate geometry of planetary rover wheels." 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2012.

[10] Buchele, Felix. and Lichtenheldt, Roy., "Multi-parameter rover wheel and grouser optimization for deployment in Phobos mili-g environment," in Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), 2020.

[11] Structural Engineering Forum of India. (n.d.). Retrieved July 27, 2022, from https://www.sefindia.org/forum/files/appc_soil_properties_718.pdf

[12] Smith, William., and Peng, Huei. "Modeling of wheel–soil interaction over rough terrain using the discrete element method." Journal of Terramechanics 50.5-6 (2013): 277-287.

[13] Cui, Da., Wang, Guoqiang., Lu, Yanpeng., and Sun, Kangkang. "Reliability design and optimization of the planetary gear by a GA based on the DEM and Kriging model." Reliability Engineering & System Safety 203 (2020): 107074.

[14] Peynot, Thierry., Lui, Sin-Ting., McAllister, Rowan., Fitch, Robert., and Sukkarieh, Salah. "Learned stochastic mobility prediction for planning with control uncertainty on unstructured terrain." Journal of Field Robotics 31.6 (2014): 969-995.

[15] Song, HyunSeok., Jung, DoHyun., and Park, BooHee. "A case study on reliability assessment for LRV& LRW." 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE, 2017.

[16] Ding, Liang., Gao, Haibo., Deng, Zongquan., Nagatani, Keiji., and Yoshida, Kazuya. "Experimental study and analysis on driving wheels' performance for planetary exploration rovers moving in deformable soil." Journal of Terramechanics 48.1 (2011): 27-45.

[17] Haldar, Achintya., and Mahadevan, Sankaran. "First-order and second-order reliability methods." Probabilistic structural mechanics handbook. Springer, Boston, MA, 1995. 27-52.

9 APPENDIX

9.1 A1 – Graphical solution

```
clc;
clear all;
close all;

%Assuming Constants from Ding wheel from article
r = 0.157; %m Radius
D = r+r; %m Diameter
W = 8.155; %Kg Weight of wheel only
b = 0.165; %m
K = 0.3; %Empiric soil parameter
T = 10; %N/m Torque

f1 = xline(57.2958, 'k','x=57');
f2 = yline(0.1, 'k','y=0.1');
f3 = xline(360, 'k','x=360');
f4 = yline(0.157,'k','y=0.157');
hold on
x1 = linspace(0,370);
x2 = linspace(0,0.2);
[X1,X2] = meshgrid(x1,x2);
F = -(((K*sqrt(W^3/(b*D^2)))/T)*r*(180./(pi*X1)).*(sqrt((1+X2).^2-(1-(0.1))^2)-
sqrt(1-(1-(0.1))^2)));
levels = -1:0.05:1;
contour(X1,X2,F,levels,'showtext','on');
hold on
[C,h] = contour(X1,X2,F,[-0.2,-0.25,-0.3]);
clabel(C,h)
xlabel('x1')
ylabel('x2')
%title('Feasible Region using Graphical Solution')

%Feasible Region
   text(150,0.12,'Feasible Region')
```

9.2 A2 - Algorithm comparison (2 design variables)

```
function [history,searchdir] = runfmincon

% Set up shared variables with OUTFUN
format long
history.x = [];
history.fval = [];
searchdir = [];

% call optimization
%x0 = [0,0]; %Initial point
%lb = [0.27,0.1413]; %-std
%ub = [0.303,0.1727]; % +std
lb = [57.2958,0.1]; %-std
ub = [360,0.157]; % +std
x0 = (lb+ub)/2
```

```
%x0 = [0,0];
options = optimoptions(@fmincon,'OutputFcn',@outfun,...
    'Display','iter','Algorithm','active-set');%,'FiniteDifferenceType', 'central');
xsol = fmincon(@objfun,x0,[],[],[],[],lb,ub,@confun,options);

 function stop = outfun(x,optimValues,state)
     stop = false;

     switch state
         case 'init'
             hold on
         case 'iter'
         % Concatenate current point and objective function
         % value with history. x must be a row vector.
           history.fval = [history.fval; optimValues.fval];
           history.x = [history.x; x];
         % Concatenate current search direction with
         % searchdir.
           %searchdir = [searchdir;...
                       %  optimValues.searchdirection'];
%            plot(x(1),x(2),'o');
         % Label points with iteration number and add title.
         % Add .15 to x(1) to separate label from plotted 'o'
%            text(x(1)+0.0025,x(2),...
%                 num2str(optimValues.iteration));
%            title('Sequence of Points Computed by fmincon');
         case 'done'
             hold off
         otherwise
     end
 end

 function f = objfun(x)
    % f = -
(((x(1)*sqrt(8.155^3/(0.165*0.314^2)))/45)*((0.157)/(18*(pi/180)))*(sqrt((1+x(2))^2-
(1-(0.1*0.157))^2)-sqrt(1-(1-(0.1*0.157))^2)));
     f = -
(((0.3*sqrt(8.155^3/(0.165*0.314^2)))/10)*0.157*(180/(pi*x(1)))*(sqrt((1+x(2))^2-(1-
(0.1))^2)-sqrt(1-(1-(0.1))^2)));
 end

 function [c, ceq] = confun(x)
     % Nonlinear inequality constraints
%      c = [1.5 + x(1)*x(2) - x(1) - x(2);
%          -x(1)*x(2) - 10];
     c = [];
     % Nonlinear equality constraints
     ceq = [];
 end
end



x1AS = ([1;1])*(180/pi);
x2AS = [0.1;0.157];
```

```
x1SQP = ([1;1;1;1])*(180/pi);
x2SQP = [0.1;0.15699;0.157;0.157];
x1IP =
([1.99;1.943;1.7005;1.0040;1.0056;1.0027;1.0040;1.0040;1.0008;1.0008;1.0008;1.0002;1.
0002;1;1])*(180/pi);
x2IP =
[0.1257;0.1287;0.1569;0.1558;0.1561;0.1563;0.1563;0.1563;0.1568;0.1569;0.1569;0.157;0
.157;0.157;0.157];

figure(2)
levels = -1:0.05:1;
[M,c] = contourf(X1,X2,F,levels,'ShowText','on');
hold on
plot(x1AS,x2AS,'bo-','LineWidth',3)
hold on
plot(x1SQP,x2SQP,'ro-','LineWidth',1)
hold on
plot(x1IP,x2IP,'ko-','LineWidth',2)
c.LineWidth = 2;
xlabel('x_1');
ylabel('x_2');
xlim([0,130]);
ylim([0,0.2])
%title('Comparison of Algoirithms using FIMCON');
legend('Design Space','Active-Set','SQP','Interior-Point')
```

9.3 A3 – Algorithm comparison (3 design variables)

```
function [history,searchdir] = runfmincon

% Set up shared variables with OUTFUN
history.x = [];
history.fval = [];
searchdir = [];

% call optimization
%x0 = [0,0,0];
lb = [57.2958,0.1,0.05];
ub = [360,0.157,0.15];
x0 = (lb+ub)/3;
options = optimoptions(@fmincon,'OutputFcn',@outfun,...
    'Display','iter','Algorithm','active-set')%,'FiniteDifferenceType', 'central');
xsol = fmincon(@objfun,x0,[],[],[],[],lb,ub,@confun,options);

 function stop = outfun(x,optimValues,state)
     stop = false;

     switch state
         case 'init'
             hold on
         case 'iter'
         % Concatenate current point and objective function
         % value with history. x must be a row vector.
           history.fval = [history.fval; optimValues.fval];
```

```
          history.x = [history.x; x];
        % Concatenate current search direction with
        % searchdir.
          %searchdir = [searchdir;...
                      %  optimValues.searchdirection'];
%           plot(x(1),x(2),'o');
        % Label points with iteration number and add title.
        % Add .15 to x(1) to separate label from plotted 'o'
%           text(x(1)+0.0025,x(2),...
%                num2str(optimValues.iteration));
%           title('Sequence of Points Computed by fmincon');
        case 'done'
            hold off
        otherwise
    end
 end

 function f = objfun(x)
    f = -
(((0.3*sqrt(8.155^3/(0.165*0.314^2)))/10)*0.157*(180/pi*x(1))*(sqrt((1+x(2))^2-(1-
x(3))^2)-sqrt(1-(1-x(3))^2)));
 end

 function [c, ceq] = confun(x)
    % Nonlinear inequality constraints
%      c = [1.5 + x(1)*x(2) - x(1) - x(2);
%          -x(1)*x(2) - 10];
    c = [];
    % Nonlinear equality constraints
    ceq = [];
 end
end


%Assuming Constants from Ding wheel from article
r = 0.157; %m Radius
D = r+r; %m Diameter
W = 8.155; %Kg Weight of wheel only
b = 0.165; %m
K = 0.3; %Empiric soil parameter
T = 10; %N/m Torque

%Constraints
%x0 = [0,0,0];
lb = [(1*(180/pi)),0.1,0.05];
ub = [360,0.157,0.15];
x0 = (lb+ub)/3;

%3D Graph
x1AS = ([1;1])*(180/pi);
x2AS = [0.1;0.157];
x3AS = [0.05;0.05];
x1SQP = ([1;1;1;1])*(180/pi);
x2SQP = [0.1;0.157;0.157;0.157];
```

```
x3SQP = [0.05;0.05;0.05;0.05];
x1IP =
([1.99;1.9436;1.6983;1.6111;1.3572;1.0018;1.0047;1.0024;1.0041;1.0034;1.0034;1.0034;1
.0034;1.0007;1.0007;1.0007;1.0001;1.0001;1;1;1])*(180/pi);
x2IP =
[0.1257;0.1287;0.1569;0.1564;0.1563;0.1565;0.1565;0.1564;0.1564;0.1564;0.1564;0.1563;
0.1563;0.1569;0.1569;0.1569;0.157;0.157;0.157;0.157;0.157];
x3IP =
[0.095;0.091;0.0558;0.05;0.0509;0.0506;0.0506;0.0506;0.0506;0.0507;0.0507;0.0507;0.05
07;0.0502;0.0501;0.0501;0.05;0.05;0.05;0.05;0.05];

figure
plot3(x1AS,x2AS,x3AS,'bo-','LineWidth',3)
hold on
plot3(x1SQP,x2SQP,x3SQP,'ro-','LineWidth',1.5)
hold on
plot3(x1IP,x2IP,x3IP,'ko-','LineWidth',1)
grid minor
xlabel('x1')
ylabel('x2')
zlabel('x3')
%title('Comparison of Algorithms using FIMCON');
legend('Active Set','SQP','Interior-Point')
xlim([50,125])
ylim([0.1,0.2])
zlim([0.04,0.1])
```

9.4 A4 - Reliability Analysis (2 uncertainties)

```
%% Set mean and standard deviation values
close all;
clear all;
clc;

n_MC = 1e6;
mKu = 0.3; % x1
mw = 8.155;
%mT = 45; % x2 [N]
%mz = 0.125; % x3 [m]
%mh = 0.125; % x4 [m]
%mphi = 20*(pi/180); % x5 [radians]
sKu = mKu/10;
sw = 1;
%sT = 0.5;
%sz = 0.1;
%sh = 0.1;
%sphi = 10*(pi/180);
%sb = 0.2;
eta0 = 0.2;
D = 0.314; % Diameter of wheel
b = 0.165; % width of wheel
%w = 8.155; % Weight of wheel
r = 0.157; % radius of wheel
T = 10; %45; % Torque on wheel
```

```
%h = 0.157; % height of grouser is 10% the radius
z = 0.1; %Zhat
%phi = 1; %*(pi/180); %angle between 2 grousers is 18 degrees which is 20 grousers
phi = 1;
h = 0.157; %hhat
%% Define limit state function g

%Finding the partial derivatives for each variable
%{
%Using formula from Graphical solution
syms x1 x2
g_fun0 = ((x1)/(2*T*phi))*(sqrt(x2^2/b))*(sqrt((1+h)^2-(1-z)^2)-sqrt(1-(1-z)^2))-eta0
dgdx1_fun = vpa(diff(g_fun0,x1))
dgdx2_fun = vpa(diff(g_fun0,x2))
d2gdx21_fun = vpa(diff(dgdx1_fun,x1))
d2gdx22_fun = vpa(diff(dgdx2_fun,x2))
%}

g_fun = @(x) ((180*x(1))/(2*T*phi*pi))*(sqrt(x(2)^2/b))*(sqrt((1+h)^2-(1-z)^2)-
sqrt(1-(1-z)^2))-eta0;
dgdx_fun = @(x)
[0.834204725931117074317847463292 33*(6.06060606060606060606060606061*x(2)^2)^(1/2);

(5.055786217764345904956651292 6808*x(1)*x(2))/(6.06060606060606060606060606061*x(2)
^2)^(1/2)];
d2gdx2_fun = @(x) [0.0;

(5.055786217764345904956651292 6808*x(1))/(6.06060606060606060606060606061*x(2)^2)^(
1/2) -
(30.641128592511187302767583592005*x(1)*x(2)^2)/(6.06060606060606060606060606061*x(
2)^2)^(3/2)];
%}
%{
g_fun = @(x) ((x(1))/(2*T*phi))*(sqrt(x(2)^2/b))*(sqrt((1+h)^2-(1-z)^2)-sqrt(1-(1-
z)^2))-eta0;
dgdx_fun = @(x)
[0.014559603944992362543261863503 36*(6.06060606060606060606060606061*x(2)^2)^(1/2);

(0.088240023909044621474314324262789*x(1)*x(2))/(6.06060606060606060606060606061*x(
2)^2)^(1/2)];
d2gdx2_fun = @(x) [0.0;

(0.088240023909044621474314324262789*x(1))/(6.06060606060606060606060606061*x(2)^2)
^(1/2) -
(0.534788023691179524086753480380 54*x(1)*x(2)^2)/(6.06060606060606060606060606061*x
(2)^2)^(3/2)];
%}

%% Calculate Pf by FORM
%
%
tic;
m = [mKu;mw];
s = [sKu;sw];
tol = 1e-10;
```

```
er = 1;
x = m;
u = [0;0];
beta_old = 1;
iter = 0;
disp('  iter   g    beta    x(1)    x(2)    er');
beta_hist = [];
g_hist = [];
while er>tol
    iter = iter + 1;
    % (a)
    g = g_fun(x);
    dgdx = dgdx_fun(x)
    m_g = real(g - sum(dgdx.*s.*u));
    s_g = real(sqrt( sum( (dgdx.*s).^2 ) ));
    % (b)
    beta = m_g/s_g;
    alpha = -dgdx.*s/s_g;
    % (c)
    x = m + beta*s.*alpha;
    u = (x - m)./s
    er = abs(beta-beta_old)/beta_old;
    beta_old = beta;
    fprintf('%6.0f %6.4f %6.4f %6.4f %6.4f %6.4f\n',...
        iter,g,beta,x(1),x(2),er);
    beta_hist = [beta_hist,beta];
    g_hist = [g_hist,g];
end
beta_FORM = beta;
x_FORM = x;
Pf_FORM = normcdf(-beta_FORM);
R = 1 - Pf_FORM;
time_FORM = toc;
%%
%
% Plot convergence history for beta

figure()
set(gcf,'defaultlinelinewidth',2,'defaultaxesfontsize',13)
plot(beta_hist,'-o');
xlabel('Iteration number');ylabel('$$\beta$$','Interpreter','latex');
title(['Convergence history. P_f=',num2str(Pf_FORM,3)]);
legend(['Converged \beta=',num2str(beta_FORM)]);
%% Calculate Pf using Breitung's method
 tic;
 dgdx = dgdx_fun(x_FORM);
 d2gdx2 = d2gdx2_fun(x_FORM);
 dgdu = dgdx.*s;
 d2gdu2 = d2gdx2.*[s(1)^2,s(1)*s(2); s(2)*s(1), s(2)^2];
 absl_grad_g = norm(dgdu,2);
 B = (1/absl_grad_g)*d2gdu2;
 H0 = (1/absl_grad_g)*[-dgdu(1), -dgdu(2) ;0, 1];
 [H1,H2] = qr(H0');
 H(1,:) = H1(:,2);
 H(2,:) = H1(:,1);
```

```
 HBHt = H*B*H';
 k1 = HBHt(1,1);
 Pf_Breitung = normcdf(-beta_FORM)*(1 + k1*beta_FORM)^(-1/2);
 time_Breitung = toc;
%% Calculate Pf using Tvedt's method
 toc;
 A1 = Pf_Breitung;
 A2 = (beta_FORM*normcdf(-beta_FORM) - normpdf(-beta_FORM))*...
      ( (1 + beta_FORM*k1)^(-1/2) - ...
        (1 + (beta_FORM + 1)*k1)^(-1/2) );
 A3 =  (beta_FORM + 1)*(beta_FORM*normcdf(-beta_FORM) - normpdf(beta_FORM))*...
     ( (1 + beta_FORM*k1)^(-1/2) - ...
       real((1 + (beta_FORM + 1i)*k1)^(-1/2)));
 Pf_Tvedt = A1 + A2 + A3;
 time_Tvedt = toc;
%% Monte Carlo Simulations

tic;
%U_FORM = (x_FORM - m)./s;
%ghat_SORM = @(U) g_fun(x_FORM) + dgdu'*(U-U_FORM) + 1/2*(U-U_FORM)'*d2gdu2*(U-
U_FORM);
g_MC = zeros(n_MC,1);
for i = 1:n_MC
    x_rand = normrnd(m,s);
    g_MC(i) = g_fun(x_rand);
end
time_MC = toc;
Pf_MC = sum(g_MC<0.0)/n_MC;

% tic;
% U_FORM = (x_FORM - m)./s;
% ghat_SORM = @(U) g_fun(x_FORM) + dgdu'*(U-U_FORM) + 1/2*(U-U_FORM)'*d2gdu2*(U-
U_FORM);
% n_MC = 1e7;
% g_SORM = zeros(n_MC,1);
% for i = 1:n_MC
%     x_rand = normrnd(m,s);
%     g_SORM(i) = ghat_SORM((x_rand-m)./s);
% end
% time_MC_SORM = toc;
% Pf_MC_SORM = sum(g_SORM<0)/n_MC;
% %% Print Results
% %
% % Since the probability is of the order of 10^(-5), i.e. 1 in 100,000, a large
number of Monte Carlo simulations have to be done to get correct result.
% %

fprintf('Sr.No.  Method         Sampling          Pf      R          Time (s)\n');
fprintf('1       FORM           --       %10.4E   %10.4E
%10.4f\n',Pf_FORM,R,time_FORM);
fprintf('2       SORM-Breitung  --       %10.4E    --
%10.4f\n',Pf_Breitung,time_Breitung);
fprintf('3       SORM-Tvedt     --       %10.4E    --
%10.4f\n',Pf_Tvedt,time_Tvedt);
```

```
    fprintf('4       FORM          MCS      %10.4E     --
%10.4f\n',Pf_MC,time_MC);
```

9.5 A5 – Reliability Analysis (3 uncertainties)

```
%% Set mean and standard deviation values
close all;
clear all;
clc;

n_MC = 1e6;
mKu = 0.3; % x1
mw = 8.155;
%mT = 45; % x2 [N]
%mz = 0.125; % x3 [m]
%mh = 0.125; % x4 [m]
%mphi = 20*(pi/180); % x5 [radians]
sKu = mKu/10;
sw = 1;
%sT = 0.5;
%sz = 0.1;
%sh = 0.1;
%sphi = 10*(pi/180);
eta0 = 0.2;
D = 0.314; % Diameter of wheel
mb = 0.165; % width of wheel
sb = mb/10;
%w = 8.155; % Weight of wheel
r = 0.157; % radius of wheel
T = 10; % Torque on wheel
%h = 0.157;
%z = 0.05*r;
%phi = 1;%*(pi/180); %angle between 2 grousers is 18 degrees which is 20 grousers
phi = 1;
h = 0.156999809485354;
z = 0.05;

%% Define limit state function g

%{
syms x1 x2 x3
g_fun0 = (((x1*sqrt(x2^3/(x3*D^2)))/T)*r*(6.28/(phi))*(sqrt((1+h)^2-(1-(z*r))^2)-
sqrt(1-(1-(z*r))^2))) - eta0;
dgdx1_fun = vpa(diff(g_fun0,x1))
dgdx2_fun = vpa(diff(g_fun0,x2))
dgdx3_fun = vpa(diff(g_fun0,x3))
d2gdx21_fun = vpa(diff(dgdx1_fun,x1))
d2gdx22_fun = vpa(diff(dgdx2_fun,x2))
d2gdx23_fun = vpa(diff(dgdx3_fun,x3))
%}
```

```
g_fun = @(x) (((x(1)*sqrt(x(2)^3/(x(3)*D^2)))/T)*r*(6.28/(phi))*(sqrt((1+h)^2-(1-
(z*r))^2)-sqrt(1-(1-(z*r))^2))) - eta0;
dgdx_fun =
@(x)[0.046356543685876908811760443995809*((10.142399285975089950224440653537*x(2)^3)/
x(3))^(1/2);

(0.705249863369866536608472225681555*x(1)*x(2)^2)/(x(3)*((10.142399285975089950224440653537*x(2)^3)/x(3))^(1/2));
    -
(0.235083287789955512202824075227185*x(1)*x(2)^3)/(x(3)^2*((10.142399285975089950224440653537*x(2)^3)/x(3))^(1/2))];
d2gdx2_fun = @(x) [0.0;

(1.410499726739733073216944451363 1*x(1)*x(2))/(x(3)*((10.142399285975089950224440653537*x(2)^3)/x(3))^(1/2)) -
(10.729388566014846158418467639437*x(1)*x(2)^4)/(x(3)^2*((10.142399285975089950224440653537*x(2)^3)/x(3))^(3/2));

(0.470166575579911024405648150454 37*x(1)*x(2)^3)/(x(3)^3*((10.142399285975089950224440653537*x(2)^3)/x(3))^(1/2)) -
(1.192154285112760684268718626604 1*x(1)*x(2)^6)/(x(3)^4*((10.142399285975089950224440653537*x(2)^3)/x(3))^(3/2))];

%% Calculate Pf by FORM
tic;
m = [mKu;mw;mb];
s = [sKu;sw;sb];
tol = 1e-10;
er = 1;
x = m;
u = [0;0;0];
beta_old = 1;
iter = 0;
disp('  iter  g   beta   x(1)   x(2)    x(3)     er');
beta_hist = [];
g_hist = [];
while er>tol
    iter = iter + 1;
    % (a)
    g = g_fun(x);
    dgdx = dgdx_fun(x)
    m_g = real(g - sum(dgdx.*s.*u));
    s_g = real(sqrt( sum( (dgdx.*s).^2 ) ));
    % (b)
    beta = m_g/s_g;
    alpha = -dgdx.*s/s_g;
    % (c)
    x = m + beta*s.*alpha;
    u = (x - m)./s
    er = abs(beta-beta_old)/beta_old;
    beta_old = beta;
    fprintf('%6.0f %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f\n',...
        iter,g,beta,x(1),x(2),x(3),er);
    beta_hist = [beta_hist,beta];
    g_hist = [g_hist,g];
```

```
end
beta_FORM = beta;
x_FORM = x;
Pf_FORM = normcdf(-beta_FORM);
R = 1 - Pf_FORM;
time_FORM = toc;
%%
%
% Plot convergence history for beta

figure()
set(gcf,'defaultlinelinewidth',2,'defaultaxesfontsize',13)
plot(beta_hist,'-o');
xlabel('Iteration number');ylabel('$$\beta_{HL}$$','Interpreter','latex');
title(['Convergence history. P_f=',num2str(Pf_FORM,3)]);
legend(['Converged \beta_{HL}=',num2str(beta_FORM)]);
%% Calculate Pf using Breitung's method
tic;
dgdx = dgdx_fun(x_FORM);
d2gdx2 = d2gdx2_fun(x_FORM);
dgdu = dgdx.*s;
%d2gdu2 = d2gdx2.*[s(1)^2,s(1)*s(2); s(2)*s(1), s(2)^2];
d2gdu2 = d2gdx2.*[s(1)^2,s(1)*s(2),s(1)*s(3); s(2)*s(1),s(2)^2,s(2)*s(3);
s(3)*s(1),s(3)*s(2),s(3)^2];
%absl_grad_g = norm(dgdu,2);
absl_grad_g = norm(dgdu,3);
B = (1/absl_grad_g)*d2gdu2;
%H0 = (1/absl_grad_g)*[-dgdu(1), -dgdu(2) ;0, 1];
H0 = (1/absl_grad_g)*[-dgdu(1), -dgdu(2), -dgdu(3) ;0, 0, 1];
%[H1,H2] = qr(H0');
[H1,H2,H3] = qr(H0');
H(1,:) = H1(:,2);
H(2,:) = H1(:,3);
H(3,:) = H1 (:,1);
HBHt = H*B*H';
k1 = HBHt(1,1,1);
Pf_Breitung = normcdf(-beta_FORM)*(1 + k1*beta_FORM)^(-1/2);
time_Breitung = toc;

%% Calculate Pf using Tvedt's method
toc;
A1 = Pf_Breitung;
A2 = (beta_FORM*normcdf(-beta_FORM) - normpdf(-beta_FORM))*...
      ( (1 + beta_FORM*k1)^(-1/2) - ...
        (1 + (beta_FORM + 1)*k1)^(-1/2) );
A3 =  (beta_FORM + 1)*(beta_FORM*normcdf(-beta_FORM) - normpdf(beta_FORM))*...
     ( (1 + beta_FORM*k1)^(-1/2) - ...
       real((1 + (beta_FORM + 1i)*k1)^(-1/2)));
Pf_Tvedt = A1 + A2 + A3;
time_Tvedt = toc;

%% Monte Carlo Simulations

tic;
%U_FORM = (x_FORM - m)./s;
```

```
%ghat_SORM = @(U) g_fun(x_FORM) + dgdu'*(U-U_FORM) + 1/2*(U-U_FORM)'*d2gdu2*(U-
U_FORM);
g_MC = zeros(n_MC,1);
for i = 1:n_MC
    x_rand = normrnd(m,s);
    g_MC(i) = g_fun(x_rand);
end
time_MC = toc;
Pf_MC = sum(g_MC<0.0)/n_MC;

% tic;
% U_FORM = (x_FORM - m)./s;
% ghat_SORM = @(U) g_fun(x_FORM) + dgdu'*(U-U_FORM) + 1/2*(U-U_FORM)'*d2gdu2*(U-
U_FORM);
% n_MC = 1e7;
% g_SORM = zeros(n_MC,1);
% for i = 1:n_MC
%     x_rand = normrnd(m,s);
%     g_SORM(i) = ghat_SORM((x_rand-m)./s);
% end
% time_MC_SORM = toc;
% Pf_MC_SORM = sum(g_SORM<0)/n_MC;
% %% Print Results
% %
% % Since the probability is of the order of 10^(-5), i.e. 1 in 100,000, a large
number of Monte Carlo simulations have to be done to get correct result.
% %

fprintf('Sr.No.  Method         Sampling          Pf      R          Time (s)\n');
fprintf('1       FORM           --       %10.4E   %10.4E
%10.4f\n',Pf_FORM,R,time_FORM);
fprintf('2       SORM-Breitung  --       %10.4E    --
%10.4f\n',Pf_Breitung,time_Breitung);
fprintf('3       SORM-Tvedt     --       %10.4E    --
%10.4f\n',Pf_Tvedt,time_Tvedt);
fprintf('4       FORM           MCS      %10.4E    --
%10.4f\n',Pf_MC,time_MC);
```

9.6 A6 – Deterministic optimization

```
function [history,searchdir] =
runfmincon_Vol_Deterministic(lb,ub,x0,v_fun,eta0,Ku,T,w,zHat,phi)

% Set up shared variables with OUTFUN
history.x = [];
history.fval = [];
searchdir = [];

% Mimimize Volume
options = optimoptions(@fmincon,'OutputFcn',@outfun,...
    'Display','iter','Algorithm','interior-point');%,'FiniteDifferenceType',
'central');
%
nc = @(x) nonlin_con(x,eta0,Ku,T,w,zHat,phi);
```

```
%
xsol = fmincon(v_fun,x0,[],[],[],[],lb,ub,nc,options);

%{
% optFun = @(U) norm(U,2);
nc = @(x) nonlin_con(x,g_fun,dgdx_fun);
[x_opt,min_volume] = fmincon(optfun,x0,[],[],[],[],lb,ub,nc,options);
%}

    function stop = outfun(x,optimValues,state)
        stop = false;

        switch state
            case 'init'
                hold on
            case 'iter'
                % Concatenate current point and objective function
                % value with history. x must be a row vector.
                history.fval = [history.fval; optimValues.fval];
                history.x = [history.x; x];
                % Concatenate current search direction with
                % searchdir.
                %searchdir = [searchdir;...
                %  optimValues.searchdirection'];
                %            plot(x(1),x(2),'o');
                % Label points with iteration number and add title.
                % Add .15 to x(1) to separate label from plotted 'o'
                %            text(x(1)+0.0025,x(2),...
                %                 num2str(optimValues.iteration));
                %            title('Sequence of Points Computed by fmincon');
            case 'done'
                hold off
            otherwise
        end
    end
%{
    function f = objfun(x)
        %f = -
(((0.3*sqrt(8.155^3/(x(1)*(2*x(2))^2)))/10)*x(2)*((6.28)/(1))*(sqrt((1+x(3))^2-(1-
(0.1))^2)-sqrt(1-(1-(0.1))^2)));
        %f = (2*pi*r*b)+(360*r*b*h^hat)
        %Volume as the objective function
        f = (2*pi*x(2)*x(1)) + (360*x(2)*x(1)*x(3))
    end
%}
    function [c, ceq] = nonlin_con(x,eta0,Ku,T,w,zHat,phi)

        % Nonlinear equality constraints
        ceq = 0;
        % Nonlinear inequality constraints
        %Consider splitting into multiple terms
        c = eta0-geteta(x,Ku,T,w,zHat,phi);

    end
end
```

```
clear;
close all;
% clc;
%
tic
thickness = 0.002; % m
lb = [0.112,0.15,0.1]; %b (wheel width) and r (radius of wheel) and h^hat (height of
grousers)
ub = [0.218,0.2,1];
x0 = (lb+ub)/3;
eta0 = 0.2;
Ku = 0.3;
T = 10;
zHat = 0.1;
phi = 12;
w = 8.155;
%stnd = [0.02;0.05;0.05];

%the g_fun
% 2Pirb + 360rbh^hat
v_fun = @(x) (pi*x(2)^2*x(1)) + (30*x(2)*x(1)*x(3)*thickness);
%dvdx_fun = @(x) [2*pi*x(2) + 360*x(2)*x(3);
 %   2*pi*x(1) + 360*x(1)*x(3);
  %  360*x(1)*x(2)];

[history,searchdir] =
runfmincon_Vol_Deterministic(lb,ub,x0,v_fun,eta0,Ku,T,w,zHat,phi);%,dvdx_fun)

figure()
set(gcf,'defaultlinelinewidth',2,'defaultaxesfontsize',13)
plot(history.fval,'-o');
xlabel('Iteration number');ylabel('Volume');
title('Iteration history');

etafinal = geteta(history.x (end,:),Ku,T,w,zHat,phi)
b_final = history.x(end,1)
r_final = history.x(end,2)
h_hat_final = history.x(end,3)
%w_final = history.x(end,4)
volume_final = history.fval(end)
toc
```

9.7 A7 – Reliability-based optimization

runfmincon_Vol_ReliabilityBased.m [MATLAB]

```
function [history,searchdir] =
runfmincon_Vol_ReliabilityBased(lb,ub,x0,v_fun,mean_Ku,mean_T,stnd,g_fun_hHat,dgdx_fu
n_hHat,Pf_min)

% Set up shared variables with OUTFUN
```

```
history.x = [];
history.fval = [];
searchdir = [];

%dvdx_fun = @(x) [2*pi*x(2) + 360*x(2)*x(3);
%    2*pi*x(1) + 360*x(1)*x(3);
%    360*x(1)*x(2)];
%{
g_fun = @(x) geteta(x) - 2.0;
dgdx_fun = @(x) [(4770475938498913*pi*(19^(1/2)/10 - ((x(3) + 1)^2 -
81/100)^(1/2)))/(422212465065984000*x(1)^2*x(2)*(4770475938498913/(35184372088832*x(1
)*x(2)^2))^(1/2));
    (4770475938498913*pi*(19^(1/2)/10 - ((x(3) + 1)^2 -
81/100)^(1/2)))/(211106232532992000*x(1)*x(2)^2*(4770475938498913/(35184372088832*x(1
)*x(2)^2))^(1/2)) - (pi*(19^(1/2)/10 - ((x(3) + 1)^2 -
81/100)^(1/2))*(4770475938498913/(35184372088832*x(1)*x(2)^2))^(1/2))/6000;
    (x(2)*pi*(2*x(3) +
2)*(4770475938498913/(35184372088832*x(1)*x(2)^2))^(1/2))/(12000*((x(3) + 1)^2 -
81/100)^(1/2));];
%}
nc = @(x) nonlin_con(x,mean_Ku,mean_T,stnd,g_fun_hHat,dgdx_fun_hHat,Pf_min);

%x0 = [0,0];
options = optimoptions(@fmincon,'OutputFcn',@outfun,...
    'Display','iter','Algorithm','interior-point');%,'FiniteDifferenceType',
'central');

xsol = fmincon(v_fun,x0,[],[],[],[],lb,ub,nc,options);


%

    function stop = outfun(x,optimValues,state)
        stop = false;

        switch state
            case 'init'
                hold on
            case 'iter'
                % Concatenate current point and objective function
                % value with history. x must be a row vector.
                history.fval = [history.fval; optimValues.fval];
                history.x = [history.x; x];
                % Concatenate current search direction with
                % searchdir.
                %searchdir = [searchdir;...
                %  optimValues.searchdirection'];
                %            plot(x(1),x(2),'o');
                % Label points with iteration number and add title.
                % Add .15 to x(1) to separate label from plotted 'o'
                %            text(x(1)+0.0025,x(2),...
                %                 num2str(optimValues.iteration));
                %            title('Sequence of Points Computed by fmincon');
            case 'done'
                hold off
```

```
            otherwise
        end
    end
%
    function [c, ceq] =
nonlin_con(x,mean_Ku,mean_T,stnd,g_fun_hHat,dgdx_fun_hHat,Pf_min)

        % Nonlinear equality constraints
        ceq = 0;
        % Nonlinear inequality constraints
        m = [x(1);mean_Ku;mean_T];%x(4)];
        h_hat = x(3);
        g_fun = @(s) g_fun_hHat(s,h_hat);
        dgdx_fun = @(s) dgdx_fun_hHat(s,h_hat);
        c = GetPf(m,stnd,g_fun,dgdx_fun) - Pf_min;
    end
end



clear;
close all;
% clc;
tic
thickness = 0.002; % m
lb = [0.132,0.15,0.1]; %b (wheel width) and r (radius of wheel) and h^hat (height of
grousers)
ub = [0.198,0.2,1];
x0 = (lb+ub)/3;
eta0 = 0.2;
Pf_min = 0.1;
%
v_fun = @(x) (pi*x(2)^2*x(1)) + (30*x(2)*x(1)*x(3)*thickness);
% mean_b = 0.165: This is also a design variable
mean_Ku = 0.3;
mean_T = 10;
stnd = [0.0165; 0.03; 1.0]; % stnd_b, stnd_Ku, stnd_T
% h_hat = 0.1;
zHat = 0.1;
phi = 12;
w = 8.155;
g_fun_hHat = @(s,h_hat) s(2) / (2 * s(3) * phi * (pi/180)) * sqrt(w^3/s(1)) * ( sqrt(
(1+h_hat)^2 - (1-zHat)^2 ) - sqrt( 1 - (1-zHat)^2 ) ) - eta0;
%
dgdx_fun_hHat = @(s,h_hat) [-(1/2)* s(2) / (2 * s(3) * phi * (pi/180)) *
sqrt(w^3/s(1)^3) * ( sqrt( (1+h_hat)^2 - (1-zHat)^2 ) - sqrt( 1 - (1-zHat)^2 ) ) ;
                           1 / (2 * s(3) * phi * (pi/180)) * sqrt(w^3/s(1)) * ( sqrt(
(1+h_hat)^2 - (1-zHat)^2 ) - sqrt( 1 - (1-zHat)^2 ) ) ;
                      -1* s(2) / (2 * s(3)^2 * phi * (pi/180)) * sqrt(w^3/s(1)) * (
sqrt( (1+h_hat)^2 - (1-zHat)^2 ) - sqrt( 1 - (1-zHat)^2 ) )];

[history,searchdir] =
runfmincon_Vol_ReliabilityBased(lb,ub,x0,v_fun,mean_Ku,mean_T,stnd,g_fun_hHat,dgdx_fu
n_hHat,Pf_min);
```

```
figure()
set(gcf,'defaultlinelinewidth',2,'defaultaxesfontsize',13)
plot(history.fval,'-o');
xlabel('Iteration number');ylabel('Volume');
title('Iteration history');
%
b_final = history.x(end,1)
r_final = history.x(end,2)
h_hat_final = history.x(end,3)
%w_final = history.x(end,4)
eta_final = geteta(history.x (end,:),mean_Ku,mean_T,w,zHat,phi)
%
g_fun = @(s) g_fun_hHat(s,h_hat_final);
dgdx_fun = @(s) dgdx_fun_hHat(s,h_hat_final);
Pf_final = GetPf(history.x (end,:),stnd,g_fun,dgdx_fun)
volume_final = history.fval(end)
   toc
```

9.8 A8 – Traction efficiency equation

```
function eta = geteta (x,Ku,T,w,zHat,phi)
b = x(1);
% r = x(2) (not used)
h_hat = x(3);
%Constants
%Ku = 0.3;
%T = 10;
%zHat = 0.1;
%phi = 1;
%w = x(4);
%w = 8.155;

% x(1) = b, x(2) = r, x(3) = h^hat

eta = Ku / (2 * T * phi * (pi/180)) * sqrt(w^3/b) * ( sqrt( (1+h_hat)^2 - (1-zHat)^2
) - sqrt( 1 - (1-zHat)^2 ) ) ;

end
```

9.9 A9 – Probability of failure equation

```
function [Pf_FORM,GradPf_FORM] = GetPf(m,s,g_fun,dgdx_fun)
% m = [mKu;mw]; % Means
% s = [sKu;sw]; % Std deviations
tol = 1e-10;
er = 1;
m = m(:);
s = s(:);
x = m(:);
u = [0;0;0];
beta_old = 1;
iter = 0;
disp('  iter   g    beta    x(1)    x(2)    x(3) er');
beta_hist = [];
```

```
g_hist = [];
while er>tol
    iter = iter + 1;
    % (a)
    g = g_fun(x);
    dgdx = dgdx_fun(x);
    m_g = g - sum(dgdx.*s.*u);
    s_g = sqrt( sum( (dgdx.*s).^2 ) );
    % (b)
    beta = m_g/s_g;
    alpha = -dgdx.*s/s_g;
    % (c)
    x = m + beta*s.*alpha;
    u = (x - m)./s;
    er = abs(beta-beta_old)/beta_old;
    beta_old = beta;
    fprintf('%6.0f %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f\n',...
        iter,g,beta,x(1),x(2),x(3),er);
    beta_hist = [beta_hist,beta];
    g_hist = [g_hist,g];
end
beta_FORM = beta;
x_FORM = x;
Pf_FORM = normcdf(-beta_FORM);
%
if nargout  > 1
    dgdx_final = dgdx_fun(x_FORM);
    s_g = sqrt( sum( (dgdx_final.*s).^2 ) );
    dbeta_dmu = dgdx_final/s_g;
    GradPf_FORM = -normpdf(beta_FORM)*dbeta_dmu;
end
%
end
```