Support Stiffness and Two-span Continuity Effects on Static Deflection and Fundamental Natural Frequency of Mass Timber Floors

by

Ning Kang

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Structural Engineering

Department of Civil and Environmental Engineering

University of Alberta

Abstract

Static deflection at mid-span under a concentrated load and fundamental natural frequency are two important vibration serviceability design parameters for mass timber panel (MTP) floor systems. Current design procedures for the two parameters tend to be conservative and over-simplified. To quantify the effects of rotational stiffness of connection, flexural stiffness of supporting beam, and two-span continuity on the two design parameters, an experiment program was conducted on 49 narrow MTP floor panel specimens at University of Alberta.

Six analytical expressions are proposed to calculate the static deflection and fundamental natural frequency of two-span floor with restrained connections, one-span floor with restrained connections and supported by beams, and two-span floor with restrained connections and supported by beams. The analytical expressions were verified against the test results and a relatively good agreement was found. Sensitivity analyses were conducted to simplify the analytical expressions. Conclusions were drawn based on the comparisons and discussions. The recommended stiffnesses of supporting beams are proposed to reach satisfactory fundamental natural frequencies for both one-span and two-span floors with varies support conditions.

Acknowledgements

This research was financially supported by the Natural Sciences and Engineering Research Council to NSERC Industrial Research Chair in Engineered Wood and Building Systems at the University of Alberta. A special gratitude towards Nordic Structures and Rothoblaas for providing the test materials.

A number of individuals need to be thanked for their support during my study. First and foremost, I wish to express my sincere gratitude to my supervisor, Dr. Ying-Hei Chui for his extraordinary guidance, continuous support, and substantial encouragement throughout this study.

I would also like to appreciate the patience and hard work of Terry Sopkow, Cameron West and Greg Miller and their instruction on the use of instruments in the lab. Thank you, Linlin Ma, Jialin Li, Sigong Zhang, Jan Niederwestberg and the intern student Ahmad Shazryz Azni for volunteering their time and donating their muscle in the lab.

Many thanks to my dear colleagues, especially Sigong Zhang, from the ARTS group at the University of Alberta for sharing their brilliance and providing enthusiastic support to me. I am also grateful to my best friends and colleagues Lidan Tang, Jialin Li, and Lin Zheng for their encouragement and friendship.

Finally, my deepest appreciation goes to my parents Shan Kang and Suhua Zhou, my husband Yanheng Li and my dogs Pepper and Luna, for their endless love and unconditional support.

Table of Contents

List of Tables

List of Figures

Chapter 1 - Introduction

1.1 Problem Statement

As a widely used construction material, wood has a history of more than 10000 years (Abeysekera et al. 2018). Wood structures benefit from wood's intrinsic advantages, including high thermal and sound insulation. With the development of mass timber products, wood has created more possibilities for reaching larger building heights and spans. Mass timber products become a possible substitution for concrete, steel and masonry for structural members. However, due to their long-span, low damping ratio and lightweight nature, vibration serviceability of mass timber panel (MTP) floors is a critical serviceability limit state design check. Often vibration serviceability governs the MTP floor design (Weckendorf et al. 2016).

In order to address the floor serviceability issue, many design standards focus on finding the empirical relationship between the vibration performance and design criteria, such as mid-span deflection under uniformly distributed load, maximum velocity from a unit impulse, acceleration and fundamental natural frequency (CCBFC 2010; CCMC 1997; Ohlsson 1988; Smith and Chui 1988; Hu and Gagnon 2012). The dynamical properties of floors, such as maximum velocity from a unit impulse and acceleration, normally require experimental method to obtain and they are not easy to compute accurately for timber floor systems. In contrast, deflection under 1 Kilonewton (kN) and fundamental natural frequency can be determined by equations, which is easy to account for properly in the normal design practice (Hu and Chui 2004). This research focused on improving the prediction accuracy of these two design criterion parameters.

In calculations of these two parameters, the simply supported and one-span assumptions are usually made in currant design standards. However, in practice the presence of multi-span floor and use of beam support are very common. In addition, the actual floor connection is more rigid than simple support due to the use of self-tapping screws and load transferred from upper storeys. Accounting for the effects of multi-span, beam support, and torsional stiffness of connection in parameter calculations can increase their accuracy, leading to more precise vibration performance prediction of MTP floors.

1.2 Research Objectives

There are two objectives in this thesis. The first objective is to conduct an experimental program to test the static deflection and fundamental natural frequency, which are known as the two criterion parameters of vibration serviceability design of one-way MTP floor panels with various support conditions to show the influence of restrained connection, two-span, and supporting beams. The test results provided from the experimental program are used to validate the analytical models proposed in this research.

The second, which is also the main objective is to develop the analytical models to predict the two parameters of one-way MTP floors, including the effect of connection restraint, supporting beam, and two-span structure.

1.3 Outline of Thesis

Chapter 2 presents a literature review of previous experimental programs and numerical simulations of static deflection and fundamental natural frequency of floors built with different construction materials and different configurations, involving semi-rigid and elastic connections. The test specimens used, and results and conclusions obtained by previous researchers are described and summarized. The current requirements in building code and design standards for floor vibration design of timber floors are also reviewed.

Chapter 3 summarizes the MTP floor panel tests conducted at University of Alberta. Details of test specimens, test set-up, test method, instrumentation and procedure are described. The test results in terms of the static deflection of floor panel under 1 kN and fundamental natural frequency are presented. The test results are also discussed.

Chapter 4 discusses the development of the analytical models for predicting the static deflection and fundamental natural frequency of two-span and one-way MTP floor with restrained connections. Sensitivity analyses were conducted on both models to generate data that led to simplification of the analytical models. These sensitivity analysis results are discussed along with the explanation of the equation simplification process. Both the original and simplified analytical models are verified against the test results presented in Chapter 3 using the circular reasoning fallacy. Some conclusions are drawn based on the comparison.

Chapter 5 develops multiple analytical models of one-way MTP floor supported by beams and having restrained connections, including both one-span and two-span structures. The analytical models can predict the static deflection and fundamental natural frequency based on the two parameters of simply support single-span beam with same configuration. Some models with extensive lengths are simplified according to the sensitivity analyses results. A comparison is made between the results from the experiment and simplified models. Some recommendations are made, and conclusions are drawn based on the comparison and discussion.

Chapter 2 - Literature Review

2.1 Introduction

The common floor vibration serviceability design parameters considered in existing design methods include deflection under 1kN load (d_{1kN}), fundamental natural frequency (f_1), peak velocity due to unit impulse (V_{peak}), and root-mean-square acceleration (a_{rms}) as shown in Table 2-1. Since deflection under 1kN and fundamental natural frequency can be determined accurately by simple equations, they are preferred for normal design practice (Hu and Chui 2004).

Table 2-1 Existing design methods in codes and literature

Parameters	Criteria	Reference
d_{UDL}	d_{UDL} < span/factor factor=360, 480…	UDL deflection method
d_{1kN}	for span < 3m: $d_{1kN} \leq 2$mm, for span ≥ 3m: $d_{1kN} \leq 8/span^{1.3}$	National Building Code of Canada (CCBFC 2010)
d_{1kN}	for span < 3m: $d_{1kN} \leq 2$mm, for 5.5m ≥ span ≥ 3m: $d_{1kN} \leq 8/span^{1.3}$ for 9.9m ≥ span ≥ 5.5m: $d_{1kN} \leq 2.55/span^{0.63}$ for span ≥ 9.9m: $d_{1kN} \leq 0.6$mm	Canadian Wood Council (CCMC 1997)
f_1, d_{1kN}, and V_{peak}	$f_1 > 8$ Hz $d_{1kN} < 1.5$ mm $V_{peak} < 100^{(f1 \cdot \xi - 1)}$	Ohlsson (1988)
f_1 and a_{rms}	$f_1 > 8$ Hz $a_{rms} \leq 0.45$ m/s^2	Smith and Chui (1988)
d_{1kN} and f_1	$f_1/ d_{1kN}^{\ 0.7} \geq 13.0$	Hu and Gagnon (2012)

In this chapter, the relevant research on the deflection and fundamental natural frequency of floor will be reviewed and discussed. The first and second parts focus on the effect of restrained connections and elastic supports on the two design parameters. The last part reviews the current methods to determine the two parameters according to building codes and design handbooks.

2.2 Review of Previous Research

2.2.1 Effect of Partially Restrained Connections on Deflection and Natural Frequency of Floor

Rathbun (1936) adopted the slope deflection and moment distribution methods in analyzing the frame with semi-rigid connections, and by assuming a linear moment-angle of rotation relationship proposed a semi-rigid connection factor Z, which is defined as:

$$Z = \frac{\theta}{M} \tag{2.1}$$

where θ is the angle of rotation and M is the moment at the connection. The semi-rigid connection factor was introduced to the matrix stiffness method by Monforton and Wu (1963). They modified the beam stiffness matrix by incorporating the effects of semi-rigid connections.

Rathbun (1936) conducted a test program consisting of 18 steel beam specimens to find their connection elastic resistance to moment. The test results show that utilizing the support from restrained connections of beams will increase the load-carrying capacity by up to 95%. It also means that for beams under the same load, considering the restrained connections will lead to a smaller deflection. Utilizing semi-rigid connections in design will reduce the beam deflections and moments which leads to lighter beams. The possible savings could reach as much as 20% when taking advantages of end restraint in design (Jones et al. 1983; SSRC 1934).

Elliott et al (2003) built 28 full scale precast concrete beam-to-column connections and conducted bending tests to find the ultimate flexural strength and rotational stiffness of connections. All precast connections tested are semi-rigid and having non-linear moment-rotation relationship. The initial rotational stiffness of the connections was assumed to be linear and defined as the tangent of moment-rotation diagrams at the intersection of test moment and beam-line. The moment of resistance of connections is 83% of ultimate test moments. In this sense, comparing assuming the simple support of beams, including the rotational stiffness of connections will obtain less beam deflection.

Hernandez and Chui (2012; 2014) conducted a series of laboratory tests to determine the rotational stiffness of 9 connection configurations with 5 different spans of cross laminated timber (CLT) floor. The test results contributed to the determination of the fixity factors of connections (Monforton and Wu 1963) which are dimensionless with values ranging from 0 to 1. The fixity factor is defined as:

$$r = \frac{1}{1 + 3\frac{EI}{RL}} \tag{2.2}$$

where EI and L are the flexural stiffness and length of floors and R is the rotational stiffness of the connection.

Zhang et al. (2019) utilized the fixity factors to compose an end restrained coefficient on deflection. The coefficient was added to the mid-span deflection of a simply supported beam to incorporate the effects of restrained connections. The CLT floor connections that were examined had fixity factors up to 0.2, which means the end restrained coefficient can go as low as 0.917. Therefore, considering the end support restraints will reduce mid-span deflection by up to 8.3%. Meanwhile, an end restrained coefficient on first natural frequency was also introduced. With a fixity factor of 0.2, the corresponding coefficient on the frequency is 1.13, which suggests that taking the restraint connections into account will raise the frequency by 13%.

Chui and Chan (1997) conducted a numerical study on the fundamental natural frequency of a three-storey steel frame with and without brace. The connection types considered include rigid, pinned and semi-rigid, which are welded and bolted specifically. The results show that adding bracing members can greatly increase the rigidity of the structure. In a braced structure, considering the connection stiffness will increase frequency slightly. Meanwhile, for unbraced structure, including connection stiffness will increase the frequency significantly. Therefore, economy should be gained in most cases.

Kawashima and Fujimoto (1984) built a small-scale L-type plane steel frame with a rotary spring as the semi-rigid connection. The natural frequencies of the frame were measured with different connection stiffnesses. The results show that natural frequencies increase as the connection stiffness increases. When the fixity factor of the semi-rigid connection reaches 0.307 and 0.470, the first natural frequency increases by 5.8% and 8.8% respectively.

Qi and Jiang (2010) conducted a numerical study based on the results from modal tests on a six-storey steel-concrete composite frame conducted by Hu et al. (2008). The semi-rigid connections obviously lower the frequencies of composite frame, compared to rigid connections. When the ratio of joint stiffness to beam linear stiffness is below 50, it is necessary to consider the effects of semi-rigid joint on natural frequencies of composite frames.

2.2.2 Effect of Elastic Supports on Deflection and Natural Frequency of Floor

Zhang et al. (2017) built a finite element model of a cantilever beam with four different connections, including clamped support and translationally and rotationally elastic supports with different stiffnesses. The translational (k_y) and rotational ($k_{\theta z}$) stiffnesses increase together. The value of longitudinal stiffness (k_x) is set to be 2×10^{19} N/mm to stimulate longitudinal rigid support. The deflections of the cantilever under a concentrated load applied at the free end are shown in Fig. 2-1 below. The node location is shown in Fig. 2-2. The first natural frequency of the cantilever beam under four constraint conditions is given in Table 2-2. It is obvious that when the connection has large translational and rotational stiffnesses, the deflection and frequency are almost the same as the clamped support. As the stiffnesses decrease, the deflection and frequency increase and decrease respectively.

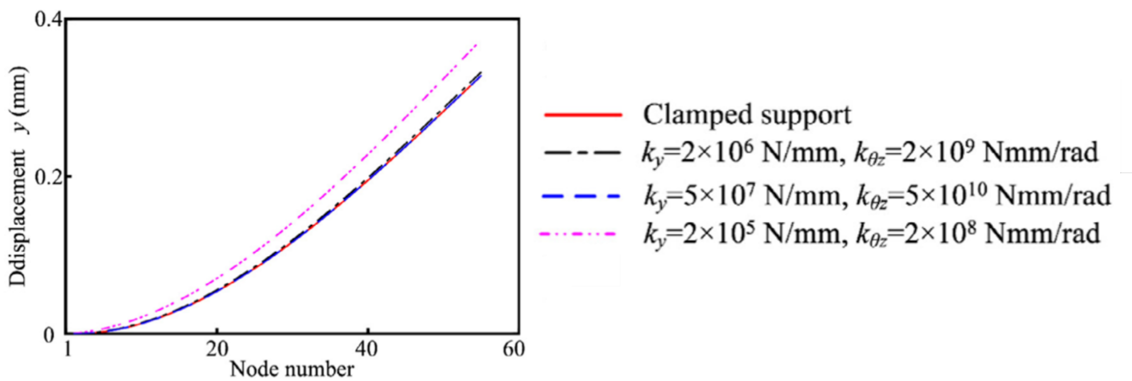


Figure 2-1 Transverse displacement under four different constraints (Zhang et al. 2017)

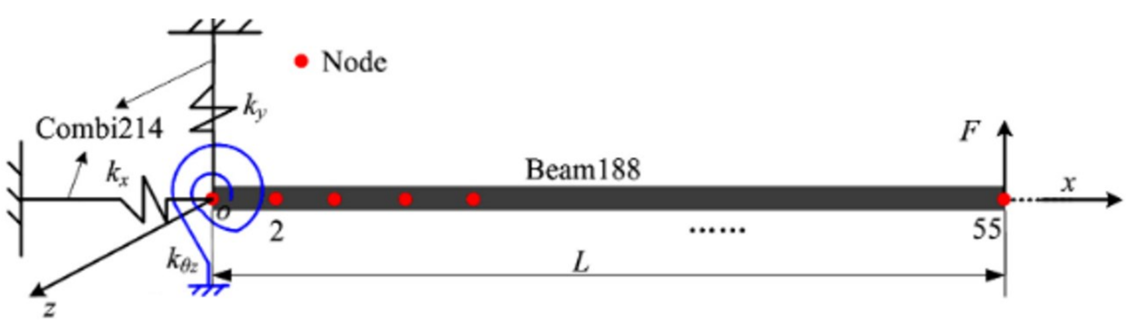


Figure 2-2 The node location of an elastic-support cantilever beam (Zhang et al. 2017)

Table 2-2 First natural frequency under four constraint conditions (Zhang et al. 2017)

Constraint Conditions	First Natural Frequency (Hz)
Clamped support	183.52
$k_y = 5 \times 10^7$N/mm, $k_{\theta z} = 5 \times 10^{10}$ N mm/rad	183.45
$k_y = 2 \times 10^6$N/mm, $k_{\theta z} = 2 \times 10^9$ N mm/rad	181.83
$k_y = 2 \times 10^5$N/mm, $k_{\theta z} = 2 \times 10^8$ N mm/rad	168.41

Chui and Smith (1990) investigated the effect of support flexibility on fundamental natural frequency of a Timoshenko beam. The beam was partially clamped at both ends. The partially clamp support was assumed to be a combination of translational stiffness, K_H, and rotational stiffness, K_T (Clough and Penzien 1975; Jacob and Ayre 1958). Figure 2-3 shows the variation of the ratio between fundamental natural frequency (f_T) of a Timoshenko Beam on flexible support and that of an identical clamped Bernoulli-Euler beam (f_E). It can be observed that the first natural frequency of a partially clamped wooded beam is sensitive to translational stiffness (K_H) regardless of slenderness of the beam (L/r) except for small rotational stiffness (K_T). The frequency is especially sensitive to change in translational stiffness within the range of 10^4 to 10^6 N/m.

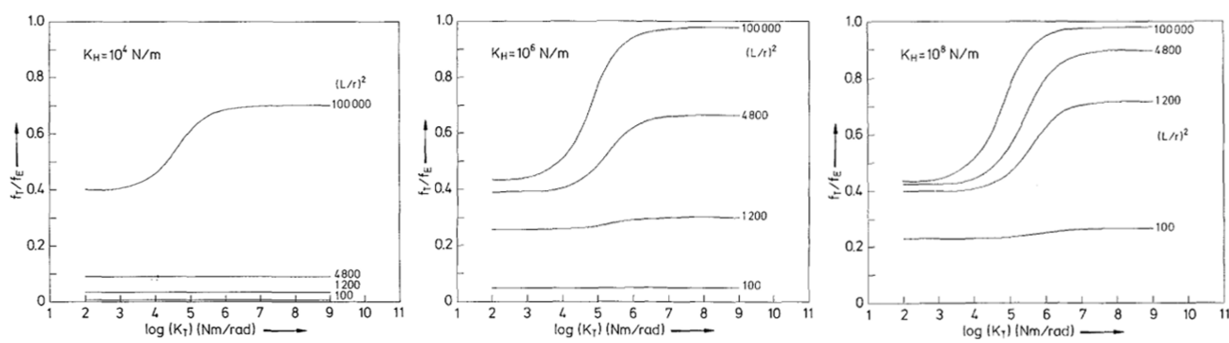


Figure 2-3 Variation of (f_T/f_E) with K_T under three different K_H (Chui and Smith 1990)

Brancheriau and Bailleres (2002) discussed Bordonné's and Goen's theoretical models which determine the effects of elastic supports on natural frequencies of wooden beams. The results show that the elastic support has very little effect on fundamental natural frequency for wooden beams with high density (1200 kg/m^3) and low length-to-depth ratio (L/d=10). However, the mean oven-dry densities of lumber, glulam, and CLT are below 500 kg/m^3 (CSA 2019).

2.2.3 Building Codes and Design Handbooks

The equations for design criterion of vibration serviceability of CLT floors, including deflection under 1kN and fundamental natural frequency are given in the Canadian CLT Handbook (2019). The equations are based on the work done by Hu and Gagnon (2012). They were first published in the Update 1 of the CSA Standard O86-14 (CSA 2016) and in the 2019 Edition of CSA Standard O86-19 (CSA 2019). The effect of restraint connections on the design criterion parameters was neglected for simplicity.

Hu and Gagnon (2012) conducted laboratory studies and subjective evaluations on CLT floors with various floor thicknesses, floor spans, joints, and support conditions. A subjective evaluation was conducted to identify the maximum acceptable annoying vibration level. Laboratory tests were conducted to determine the deflections under 1kN concentrated static load, natural frequencies, damping ratios, and vibration modes of the CLT floor specimens. The test results showed that CLT stiffness in the major strength direction and mass affect vibration performance deeply. A composite design criterion consisting of fundamental natural frequency and 1 kN static deflection was proposed for CLT floors.

The design criterion has the following general form (Hu at al. 2016):

$$\frac{f}{d_{1kN}^{x}} \geq y \tag{2.3}$$

where x and y are constants to be determined from floor response measurements and subjective evaluation data.

For an acceptable CLT floor, the following criterion must be met (Hu and Gagnon 2012):

$$\frac{f}{d^{0.7}} \geq 13.0 \qquad or \qquad d \leq \frac{f^{1.43}}{39.0} \tag{2.4}$$

One of the design parameters, the deflection under 1kN, is determined using (Hu and Gagnon 2012):

$$d = \frac{1000PL^3}{48(EI)_{app}} \tag{2.5}$$

where:

d = static deflection at mid-span under a 1 kN concentrated load of a 1 m wide simply supported CLT panel (mm)

P =1000 (N)

$(EI)_{app}$ = $0.9(EI)_{eff}$

where:

$(EI)_{eff}$ = effective bending stiffness in the major strength direction of a 1 m wide CLT floor panel (Nm^2)

Another parameter, the fundamental natural frequency of a CLT floor, can be calculated using:

$$f = \frac{3.142}{2L^2}\sqrt{\frac{(EI)_{app}}{\rho A}} \tag{2.6}$$

where:

f = fundamental natural frequency of a 1 m simply supported CLT panel (Hz)

L = span of CLT floor (m)

ρ = mass density (kg/m^3)

A = cross-section area of 1 m wide CLT panel (m^2)

For CLT floor supported on beams, Hu and Gagnon (2012) introduced a supporting beam stiffness requirement to ensure that the floor vibration is acceptable. The equation was initially proposed by Hu (2018). The general vibration-controlled criterion for supporting beam with constants x and y:

$$(d_{1\,kN})_{beam} \leq \frac{x}{{l_{beam}}^{y}} = \frac{157.63}{{l_{beam}}^{3.55}} \tag{2.7}$$

where:

$(d_{1\,kN})_{beam}$ = beam deflection under 1 kN point load (mm)

l_{beam} = clear span of supporting beam (m)

x = 157.63, identified using beam test results

y = 3.55, identified using beam test results

The supporting beam deflection under 1 kN can be calculated using:

$$(d_{1\,kN})_{beam} = F_{span} \frac{1000pl_{beam}{}^{3}}{48(EI)_{beam}} \tag{2.8}$$

where:

p = point load of 1000 (N)

$(EI)_{beam}$ = apparent bending stiffness of supporting beam (Nm^2)

F_{span} = 1.0 for simple span beam and ≈ 0.7 for a multi-span continuous beam

The supporting beam stiffness requirement was obtained by substituting Eq. (2.8) into (2.7):

$$(EI)_{beam} \geq F_{span} 132.17 l_{beam}{}^{6.55} \tag{2.9}$$

Due to the improved vibration performance provided by intermediate support continuity, the vibration-controlled span for multiple-span CLT floors can be increased by up to 20%, but this simple rule only applies to floor span that does not exceed 8 m (Karacabeyli and Gagnon 2019).

2.3 Summary

This chapter discusses building code requirements on design parameters of vibration performance of timber floors and the previous research conducted to investigate the effects of restrained connections, continuous span, and elastic supports on the static point load deflection and fundamental natural frequency of a beam member.

The review of the studies has covered the structures built of steel, concrete, wood, and composite material. Irrespective of building materials and structure configurations, connection rotational stiffnesses at supports can influence beam deflection and fundamental natural frequency, compared with simply supported beam. Meanwhile, simple beams on elastic supports can increased deflection and reduced natural frequencies compared with the same beam in rigid foundation. Under some special circumstances, such as translational stiffness over a sensitive range, beam with large density and high length-to-depth ratios, the effect of elastic supports can be neglected for simplicity.

The design parameters defined by CSA O86-14 and CSA O86-19 Standards were based on the single-span simple support conditions, neglecting rotational stiffness of connections. In reality, the actual support conditions are never simple. Modified equations are desirable to allow design engineers to properly account for the end support condition in real MTP floor systems, leading to more economical MTP floor designs. This project focused on quantifying the effects of restrained connection, two-span, and elastic support on the two vibration design parameters, namely static deflection under point load at mid-span and fundamental natural frequency of MTP floors, and the development of analytical models to account for these effects.

Chapter 3 - Experimental Program

3.1 Introduction

As discussed in the previous chapter, static mid-span deflection and fundamental natural frequency, the two design parameters for MTP floor vibration performance, can be influenced by the boundary condition, supporting elements and double span arrangement. To investigate the effects of each factor on the two parameters, a laboratory test program was conducted at the University of Alberta. A major goal of this test program is to generate test data to validate analytical models that were developed to predict the two design parameters which will be presented in the next chapter.

The testing program was performed on narrow mass timber floor panels. It covered two types of floor systems of interest, namely two-span floors and floors supported on beams. In this chapter, a description of the test specimens and test setup is presented. This is followed by test methods and procedures, and finally a discussion of the test results in the form of mid-span deflection and fundamental natural frequency.

The specific objectives of the test program are to:

1. Provide test data to validate analytical models for static deflection and fundamental natural frequency in the next chapter.
2. Explore effects of fixity of the middle support on the static mid-span deflection and natural frequency of two-span CLT and glued laminated timber (GLT) floors supported on walls (rigid foundation).
3. Explore effects of fixity of the first and third supports on the static mid-span deflection and natural frequency of two-span CLT and GLT floors supported on walls.
4. Explore effects of rotational restraint in supports on static deflection and natural frequency of one-span and beam-supported floors (flexible foundation).
5. Explore effects of supporting beam stiffness on the static mid-span deflection and natural frequency of one-span and two-span GLT floors supported on beams.
6. Explore effects of span ratio of beam on the static mid-span deflection and natural frequency of two-span GLT floors supported on beams.

3.2 Test Specimens

In the experimental program, six narrow floors were built in total. Three floors were built of CLT panels and the other three were built with GLT panels. All six floors have the same length, 6096 mm (20 ft). The three CLT floors have the same width of 800 mm but different thicknesses. The detailed layer combination information of the CLT panel is listed in Table 3-1. The three GLT floors are 600 mm wide and of thicknesses of 130 mm, 175 mm, and 215 mm respectively.

Table 3-1 CLT Panel Layups

Panel Thickness (mm)	Number of Plys	Ply Combination (mm)
105	3	35L-35T-35L
175	5	35L-35T-35L-35T-35L
244	7	35L-35T-35L-35T-35L-35T-35L

* L=longitudinal, T=transversal

The supporting elements of the floor panels were walls and beams. A wall support had a height of 305 mm (1 ft) and was made up of the same MTP product with the same width as the floor specimen. A supporting beam was made by combining either 2 or 3 pieces of 38mm x 140mm (2x6) spruce-pine-fir (SPF) lumber. The length of the 2x6 lumber beam was 1829 mm. The lumber pieces were oriented flatwise. The 2-ply and 3-ply layups provided beams with two different bending stiffnesses.

In addition to the two-span and beam support floor configurations, three boundary conditions were considered in the experimental program. The first boundary condition was simple support which was simply letting the floor panel rest on the supporting walls or beams. The second boundary condition was achieved by fastening the ends of the panel to the supporting wall or beam using self-tapping screws. Since the six floor panels adopted in the experiment varied in thickness, a specific length of the self-tapping screw was selected for each panel as shown in Table 3-2. The diameter of the self-tapping screws was 8 mm. Two fastener spacings to accommodate 3 and 5 STS were tested for each panel and the detailed layout is shown in Figure 3-1. The third boundary condition was achieved by applying a top load to the ends of the panel to simulate the load transferring from the upper storey of a building and the load was applied along the width of floor panel. The magnitude of the applied load was varied from 5.25 kN to 49.46 kN.

Table 3-2 Self-tapping screws used for the MTP floor panels

Panel Type	Panel Thickness (mm)	Screw model	Screw diameter (mm)	Screw length (mm)
CLT	105	HBS8240	8	240
CLT	175	HBS8280	8	280
CLT	244	HBS8340	8	340
GLT	130	HBS8160	8	160
GLT	175	HBS8280	8	280
GLT	215	HBS8340	8	340

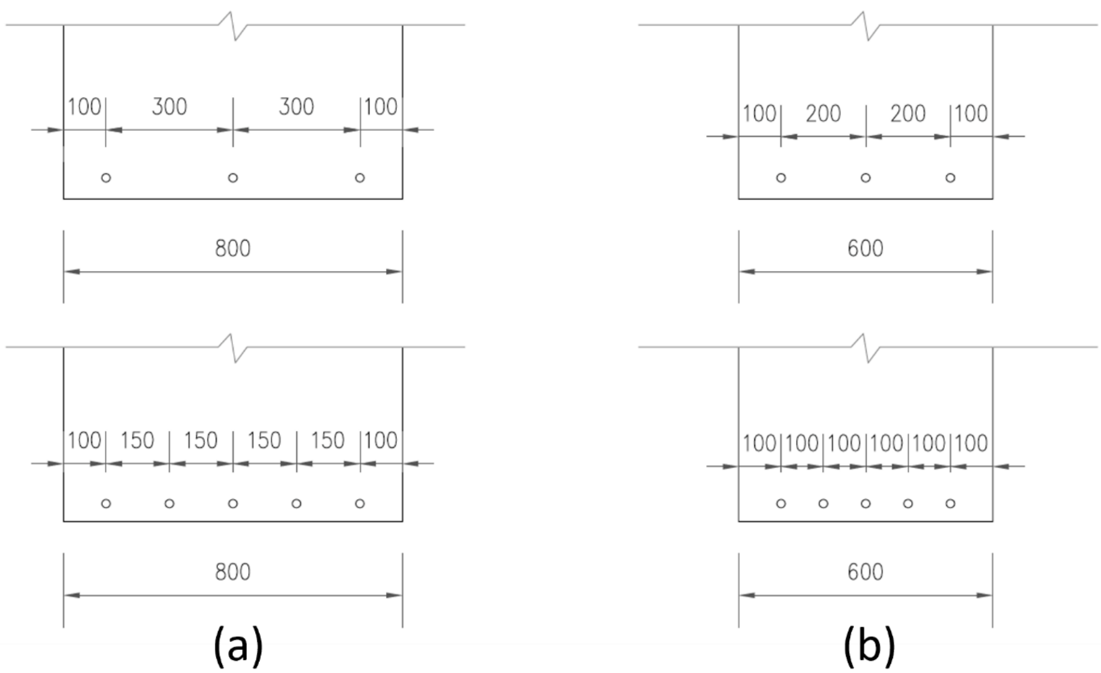


Figure 3-1 Self-tapping screw layout on (a) CLT floors and (b) GLT floors

3.3 Test Set-up

3.3.1 Two-span MTP Floor Test on Wall Support

The two-span MTP floor specimen required three walls located at the ends of the floor and one between them. As shown in Figure 3-2, the total length of the floor is 6096 mm (20 ft), and the intermediate wall separated the floor into 3658 mm (12 ft) and 2438 mm (8 ft) long spans of which the span length ratio is 3:2. Each wall had two triangular supports attached at both sides to improve the floor's stability by preventing rotation or tilt of supporting walls. These triangular supports did not lead to additional stiffness to the walls or floor panels.

For the test set-up that required the application of top load over the supported ends of the panel, a 152 mm (0.5 ft) high wall was placed on the top, and the top load was applied via this short wall. The load was generated by an ENERPAC hollow plunger hydraulic cylinder, as shown in Figure 3-2. The hydraulic actuator was reacted against an HSS section that was connected to the laboratory floor via two 22mm (7/8 in) threaded steel rods. The use of steel plates between the HSS and hydraulic cylinders prevented the local deformation of HSS. The purpose of HSS was to transfer the load generated by hydraulic cylinders to the floor evenly along the width. The same arrangement was used to apply a line load to the panel at the mid-span of the longer span to generate the static deflection under the load.

In the simple support test set-up, an HSS was placed over the supported end that could produce uplift when applying the load to the mid-span of the longer span. In that case, the HSS was simply in contact with the panel end, and there was no application of load. Since the weight of HSS was supported by the bolts underneath and not transferred to the panel, the boundary condition would remain as a simple support.

Alphanumeric I.D. was used to identify specimen configuration. It contains four variables that begin with MTP category and panel thickness in millimeter, such as “C105” and “G175”. After this, there are three variables separated by a dash. The three variables refer to the boundary condition in the sequence of end support of the longer span, intermediate support, and end support of the shorter span. If the condition is simple support, “S” is used in the I.D. If self-tapping screws are used, the code is either “3STS” or “5STS” to reflect the number of self-tapping screws used. When additional top load was applied over the end support, the numerical value of the load generated by hydraulic jacks in kN is included in the variable. For example, G130-S-5STS+29.97-S represents a two-span floor specimen built with 130 mm thick GLT panel; its first and third supports are simple boundary condition, and the intermediate support has 5 self-tapping screws with an additional 29.97 kN top load generated by hydraulic cylinders.

In addition, several single-span floors with the same kinds of connection as the two-span floor tests with a length of 3658 mm (12 ft) or 2438 mm (8 ft), which is the length of either span of the two-span floor, were also tested for reference purposes. The test configurations of two-span CLT and GLT floor and the single-span reference floor tests supported by walls are presented in Table 3-3, 3-4, and 3-5 respectively.

Table 3-3 Two-span and wall-supported CLT floor test configurations

Floor ID	Panel Type	Boundary Condition of End Supports	Boundary Condition of Middle Support
C105-S-S-S	CLT105	Simple support	Simple support
C105-S-3STS-S	CLT105	Simple support	3 STS
C105-S-5STS-S	CLT105	Simple support	5 STS
C105-S-5STS+10.32-S	CLT105	Simple support	5 STS + Jacks at 10.32 kN
C105-S-5STS+29.97-S	CLT105	Simple support	5 STS + Jacks at 29.97 kN
C105-S-5STS+49.62-S	CLT105	Simple support	5 STS + Jacks at 49.62 kN
C105-5.41-5STS-5.41	CLT105	Jacks at 5.41 kN	5 STS
C105-10.32-5STS-10.32	CLT105	Jacks at 10.32 kN	5 STS
C105-20.15-5STS-20.15	CLT105	Jacks at 20.15 Kn	5 STS
C105-29.97-5STS-29.97	CLT105	Jacks at 29.97 kN	5 STS
C105-39.80-5STS-39.80	CLT105	Jacks at 39.80 kN	5 STS
C105-49.62-5STS-49.62	CLT105	Jacks at 49.62 kN	5 STS
C175-S-S-S	CLT175	Simple support	Simple support
C175-S-3STS-S	CLT175	Simple support	3 STS
C175-S-5STS-S	CLT175	Simple support	5 STS
C175-S-5STS+29.97-S	CLT175	Simple support	5 STS + Jacks at 29.97 kN
C175-S-5STS+39.80-S	CLT175	Simple support	5 STS + Jacks at 39.80 kN
C175-10.32-5STS-10.32	CLT175	Jacks at 10.32 kN	5 STS
C244-S-S-S	CLT244	Simple support	Simple support
C244-S-5STS-S	CLT244	Simple support	5 STS
C244-S-5STS+10.32-S	CLT244	Simple support	5 STS + Jacks at 10.32 kN
C244-S-5STS+29.97-S	CLT244	Simple support	5 STS + Jacks at 29.97 kN
C244-10.32-5STS-10.32	CLT244	Jacks at 10.32 kN	5 STS
C244-39.80-5STS-39.80	CLT244	Jacks at 39.80 kN	5 STS

Table 3-4 Two-span and wall-supported GLT floor test configurations

Floor ID	Panel Type	Boundary Condition of End Supports	Boundary Condition of Middle Support
G130-S-S-S	GLT130	Simple support	Simple support
G130-S-5STS+10.32-S	GLT130	Simple support	5 STS + Jacks at 10.32 kN
G130-S-5STS+29.97-S	GLT130	Simple support	5 STS + Jacks at 29.97 kN
G130-S-5STS+44.71-S	GLT130	Simple support	5 STS + Jacks at 44.71 kN
G175-S-S-S	GLT175	Simple support	Simple support
G175-S-3STS-S	GLT175	Simple support	3 STS
G175-S-5STS-S	GLT175	Simple support	5 STS
G175-10.32-5STS-10.32	GLT175	Jacks at 10.32 kN	5 STS
G175-29.97-5STS-29.97	GLT175	Jacks at 29.97 kN	5 STS
G175-44.71-5STS-44.71	GLT175	Jacks at 44.71 kN	5 STS
G215-S-S-S	GLT215	Simple support	Simple support
G215-S-5STS-S	GLT215	Simple support	5 STS
G215-10.32-5STS-10.32	GLT215	Jacks at 10.32 kN	5 STS
G215-29.97-5STS-29.97	GLT215	Jacks at 29.97 kN	5 STS
G215-44.71-5STS-44.71	GLT215	Jacks at 44.71 kN	5 STS

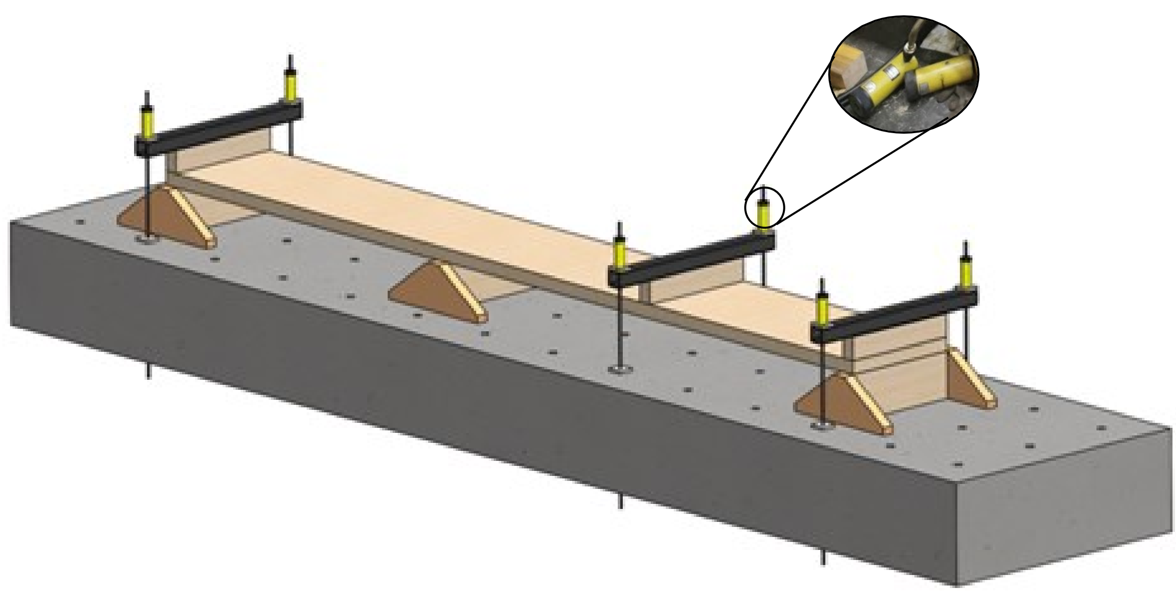


Figure 3-2 Two-span MTP floor set-up

Table 3-5 One-span and wall-supported reference floor test configurations

Floor ID	Panel Type	Span Length (mm)	Boundary Condition
C105-3658-S	CLT105	3658	Simple support
C105-3658-5.41	CLT105	3658	Jacks at 5.41 kN
C105-3658-29.97	CLT105	3658	Jacks at 29.97 kN
C105-3658-20.15	CLT105	3658	Jacks at 20.15 kN
C105-3658-29.97	CLT105	3658	Jacks at 29.97 kN
C105-3658-39.80	CLT105	3658	Jacks at 39.80 kN
C105-3658-49.62	CLT105	3658	Jacks at 49.62 kN
C175-3658-S	CLT175	3658	Simple support
C175-3658-29.97	CLT175	3658	Jacks at 29.97 kN
C244-3658-S	CLT244	3658	Simple support
C244-3658-29.97	CLT244	3658	Jacks at 29.97 kN
C244-3658-39.80	CLT244	3658	Jacks at 39.80 kN
G175-3658-S	GLT175	3658	Simple support
G175-3658-10.32	GLT175	3658	Jacks at 10.32 kN
G175-3658-29.97	GLT175	3658	Jacks at 29.97 kN
G175-3658-44.71	GLT175	3658	Jacks at 44.71 kN
G215-3658-S	GLT215	3658	Simple support
G215-3658-10.32	GLT215	3658	Jacks at 10.32 kN
G215-3658-29.97	GLT215	3658	Jacks at 29.97 kN
G215-3658-44.71	GLT215	3658	Jacks at 44.71 kN
C105-2438-S	CLT105	2438	Simple support
C105-2438-5.41	CLT105	2438	Jacks at 5.41 kN
C105-2438-10.32	CLT105	2438	Jacks at 10.32 kN
C105-2438-20.15	CLT105	2438	Jacks at 20.15 kN
C105-2438-29.97	CLT105	2438	Jacks at 29.97 kN
C105-2438-39.80	CLT105	2438	Jacks at 39.80 kN
C105-2438-49.62	CLT105	2438	Jacks at 49.62 kN
C175-2438-S	CLT175	2438	Simple support
C175-2438-10.32	CLT175	2438	Jacks at 10.32 kN

3.3.2 Beam-supported MTP Floor Test

The beam-supported MTP floor panel tests were conducted on both one-span and two-span floor systems. There were 14 tests in total, in which six were one-span tests, and the other eight were two-span tests. In the six one-span tests, two were conducted on 4.88m (16 ft) long floor panel, and the remaining four were on a 6.09m (20 ft) long panel. As for the two-span tests, all test specimens had 6.09m (20 ft) long floor panel. The two different span ratios considered were 3:2 and 1:1. All the test specimens were built using a 175 mm thick GLT panel. Two boundary conditions: simple support and self-tapping screws at 200 mm spacing, were investigated. The self-tapping screw size was 8mm x 280 mm. For reference purposes, 4.88m long one-span specimen, 6.09m long one-span specimen, and 6.09m long two-span specimen with a span ratio of 3:2 were tested on wall supports.

The main difference between the single-span and two-span systems was the presence of an intermediate supporting beam in the two-span test set-up. As shown in Figure 3-3, all the supporting beams were hung over the walls made with 600 mm tall GLT pieces. Two 50mm x 50mm (2 in × 2 in) wood blocks were attached to the foot of the wall to enhance the stability of the test set-up. The lumber beam was fastened to the GLT wall using one self-tapping screw. The two-layer and three-layer beams adopted HBS8160 and HBS8240 screws respectively.

To prevent uplift at the end of the beam during static deflection tests, a 22.7 kg (50 lb) weight was placed on top of each beam-supporting end.

Since all the specimens were built with 175 mm thick GLT panels, specimen I.D. does not include the panel information for beam-supported floor specimens. The one-span specimen I.D. starts with "S" representing single-span, followed by three variables: beam type, floor span length in mm, and boundary condition. While two-span specimens' I.D. starts with "T" meaning two-span, the three variables following are beam type, span ratio, and boundary condition. The lumber beam has two types, 1829 mm (6 ft) long two-layer beam and 1829 mm (6 ft) long three-layer beam recorded as "2LB" and "3LB" in the specimen I.D. For example, S-3LB-6096-S means a 6096 mm long single-span floor supported by three-layer lumber beams, and all the end supports are simple conditions. A two-span floor supported on three 3-layer lumber beams, with a span ratio of 3:2, and the floor panel fastened to the lumber beams using three self-tapping screws is identified as specimen T-3LB-3:2-3STS-3STS-3STS. All beam-supported test configurations are listed in Table 3-6.

Table 3-6 Beam-supported GLT floor test and reference test configurations

Floor ID	Number of Span	Support Type	Span Length (mm)	Boundary Conditions
G175-4877-S (ref)	One	Wall	4877	Simple support
G175-4877-3STS (ref)	One	Wall	4877	3 STS at 200 mm
S-3LB-4877-S	One	3LB	4877	Simple support
S-3LB-4877-3STS	One	3LB	4877	3 STS at 200 mm
G175-6096-S (ref)	One	Wall	6096	Simple support
G175-6096-3STS (ref)	One	Wall	6096	3 STS at 200 mm
S-3LB-6096-S	One	3LB	6096	Simple support
S-3LB-6096-3STS	One	3LB	6096	3 STS at 200 mm
S-2LB-6096-S	One	2LB	6096	Simple support
S-2LB-6096-3STS	One	2LB	6096	3 STS at 200 mm
G175-3658-S (ref)	One	Wall	3658	Simple support
G175-3658-3STS (ref)	One	Wall	3658	3 STS at 200 mm
G175-S-3STS-S (ref)	Two	Wall	3658/2438	3 STS at 200 mm
T-3LB-3:2-S-S-S	Two	3LB	3658/2438	Simple support
T-3LB-3:2-S-3STS-S	Two	3LB	3658/2438	Simple-3 STS at 200 mm-Simple
T-3LB-3:2-3STS-3STS-3STS	Two	3LB	3658/2438	3 STS at 200 mm
T-2LB-3:2-S-S-S	Two	2LB	3658/2438	Simple support
T-2LB-3:2-S-3STS-S	Two	2LB	3658/2438	Simple-3 STS at 200 mm-Simple
T-2LB-3:2-3STS-3STS-3STS	Two	2LB	3658/2438	3 STS at 200 mm
T-3LB-1:1-3STS-3STS-3STS	Two	3LB	3658/2438	3 STS at 200 mm
T-2LB-1:1-3STS-3STS-3STS	Two	3LB	3658/2438	3 STS at 200 mm

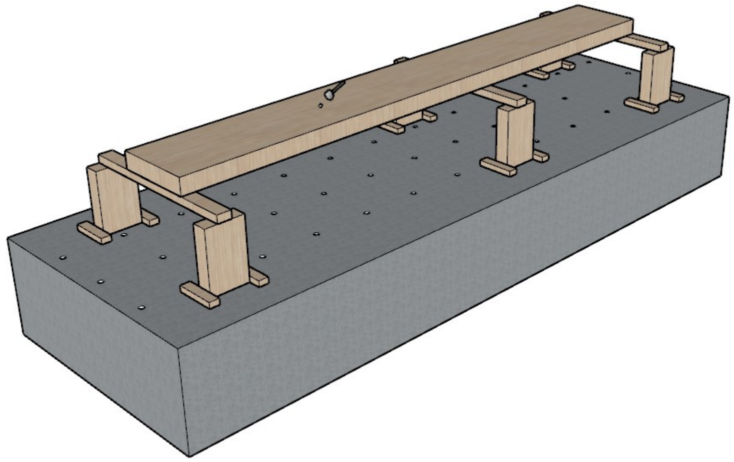


Figure 3-3 Impact hammer modal test set-up of two-span and beam-supported floor

3.3 Test Method and Procedure

3.3.1 Static mid-span deflection test

Due to the change in the availability of deflection measuring devices, the test instrumentation and method had changed slightly at different stages of the test program. As shown in Figure 3-4, in the two-span floor test, the out-of-plane floor displacement in the middle of longer span was monitored at two edges of the floor. In the beginning, two draw-wire sensors were installed at the monitoring locations and the wires were connected to angle brackets which were mounted at the bottom of floor panel. Later, Linear Variable Differential Transformers (LVDTs) were used instead. The LVDTs were installed underneath the floor with the tips touching the floor panel. The signals of the draw-wire sensors and LVDTs were recorded by a computer-based data acquisition system. In the final stage, the displacement was measured by two Mitutoyo Dial Indicators with graduations of 0.01 mm. The installation and location of dial indicators were the same as LVDTs. The data reading and recording were performed manually. The two-span floor tests supported on wall used all three devices while the beam-supported floor tests mainly relied on Dial Indicators.

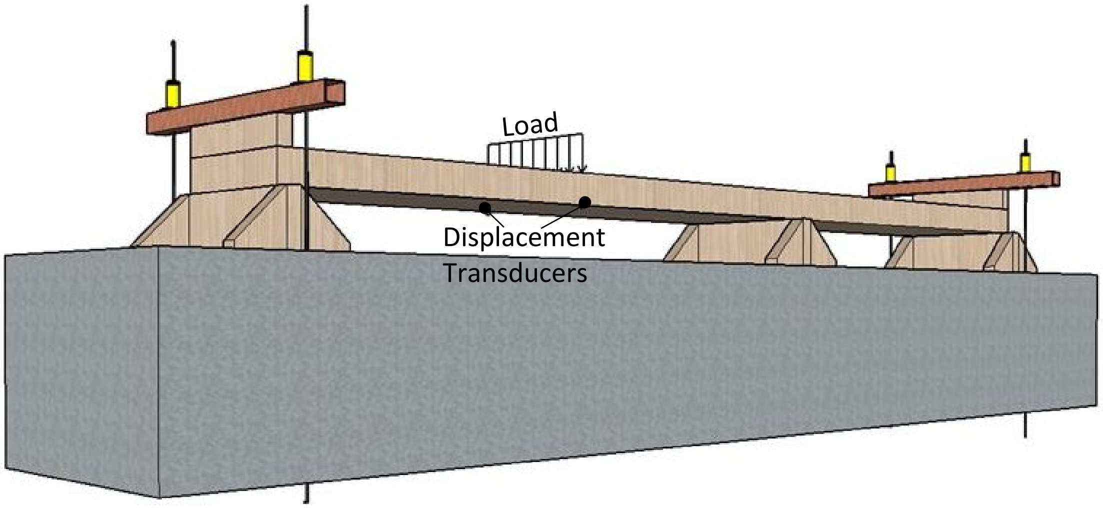
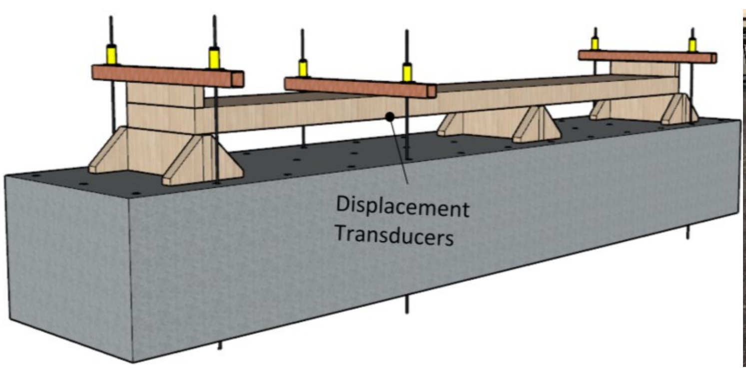


Figure 3-4 Static deflection test set-up

Deflection tests included two load application systems as shown in Figure 3-5 and 3-6. The first one was an ENERPAC hollow plunger hydraulic cylinder similar to applying top loads over supported ends which are supported on walls. Two hydraulic cylinders were installed on the thread rods on the two sides of the floor specimen. The load was applied via an HSS as shown in Figure 3-5. These two hydraulic cylinders were connected to a hand pump. Therefore, they could reach the same pressure level at the same time. The second way of applying loads for deflection test was using a 22.7 kg (50-lb) weight. Both loading systems were used for the two-span floor specimens supported on walls. The dead load approach was only applied to the beam-supported test specimens.



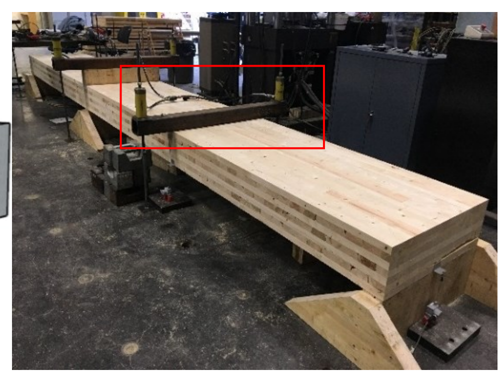


Figure 3-5 Static deflection test set-up using hydraulic cylinders

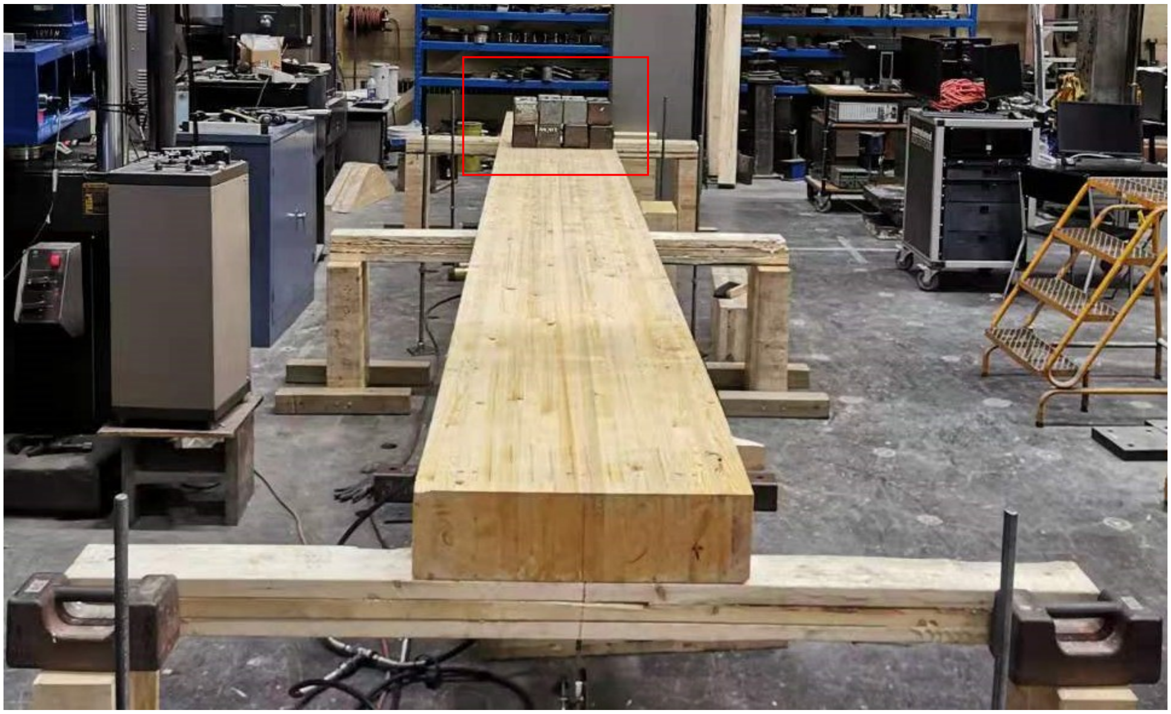


Figure 3-6 Static deflection test set-up using 222 N weights

When the hydraulic cylinders were used, the actuator pressure was increased to above 2.76 MPa (400 psi) using a hand pump at the start. The pressure was gradually decreased to 2.07 MPa (300 psi), 1.38 MPa (200 psi), and then 0.69 MPa (100 psi). At each step, the corresponding deflection after the pressure reading has stabilized was recorded. The total load applied to the floor was the combination of load generated by hydraulic cylinders and weight of loading instruments, including hydraulic cylinders, HSS, and steel plates. The load generated by each hydraulic cylinder was found by multiplying the hydraulic pressure and effective area of the cylinder. The total weight of the hydraulic cylinders, HSS and steel plates was 334.5 N.

When dead weights were used to apply the load, the test started with placing four 222 N (50-lb) weights to the middle of the longer span each time and recorded the displacement readings under the load. After that, four additional weights, 890 N, 1779 N, 2669 N and 3558 N, were applied, and floor deflection after each weight increase was measured. Later, to simplify the test process, load increments of 445 N, 890 N, 1334 N and 1779 N were applied instead. Table 3-7 summarizes the loads applied on test floors based on the use of hydraulic cylinders and dead weight. Each specimen was loaded in four increments, and the results are expressed as deflection per unit applied load, which is the slope of the regression line fitted through the four data points. This slope value can also be considered as deflection under a kN load.

On top of reference test specimens, the static load deflection tests were also conducted on the 2-ply and 3-ply lumber beams for analysis purposes, as shown in Figure 3-7. The lumber beam test was conducted before installing the mass timber floor panel on top, and all the lumber beams used were tested. The beam specimen was supported on walls and fastened by one self-tapping screw to the wall below.

Figure 3-7 Beam static deflection test set-up

Table 3-7 Loads applied in static load deflection test

Hydraulic Cylinders		Dead Weights			
Pressure (MPa)	Total Load (kN)	Weight (lbf)	Total Load (kN)	Weight (lbf)	Total Load (kN)
0.69	2.79	200	0.89	100	0.45
1.38	5.25	400	1.78	200	0.89
2.07	7.70	600	2.67	300	1.33
2.76	10.16	800	3.56	400	1.78

3.3.2 Impact hammer modal testing

The impact hammer modal tests were conducted on all the floor panel specimens. The object of modal tests was to determine the fundamental natural frequencies of the floor specimens. The equipment used for the measurements is Photon+ from Brüel & Kjær with one accelerometer and an instrumented hammer. The acceleration of the floor and hammer impact force signals were recorded using Brüel & Kjær's RT Pro data acquisition software.

In the modal test, the accelerometer was mounted to the top surface of floor panel in the middle of the floor span or in the middle of longer span in two-span tests as shown in Figure 3-7. The floor was excited into motion by hitting the floor panel with the impact hammer. The time signals of the accelerometer and the impact hammer were recorded by the Photon+ and subsequently analyzed by a modal analysis software in a laptop computer. The software produces frequency response function (FRF), from which the fundamental natural frequency of floor specimen can be determined as the first peak of the FRF, as illustrated in Figure 3-9.

Similar to deflection tests, the modal tests were also repeated on the reference test specimens and lumber beams to determine their fundamental natural frequency.

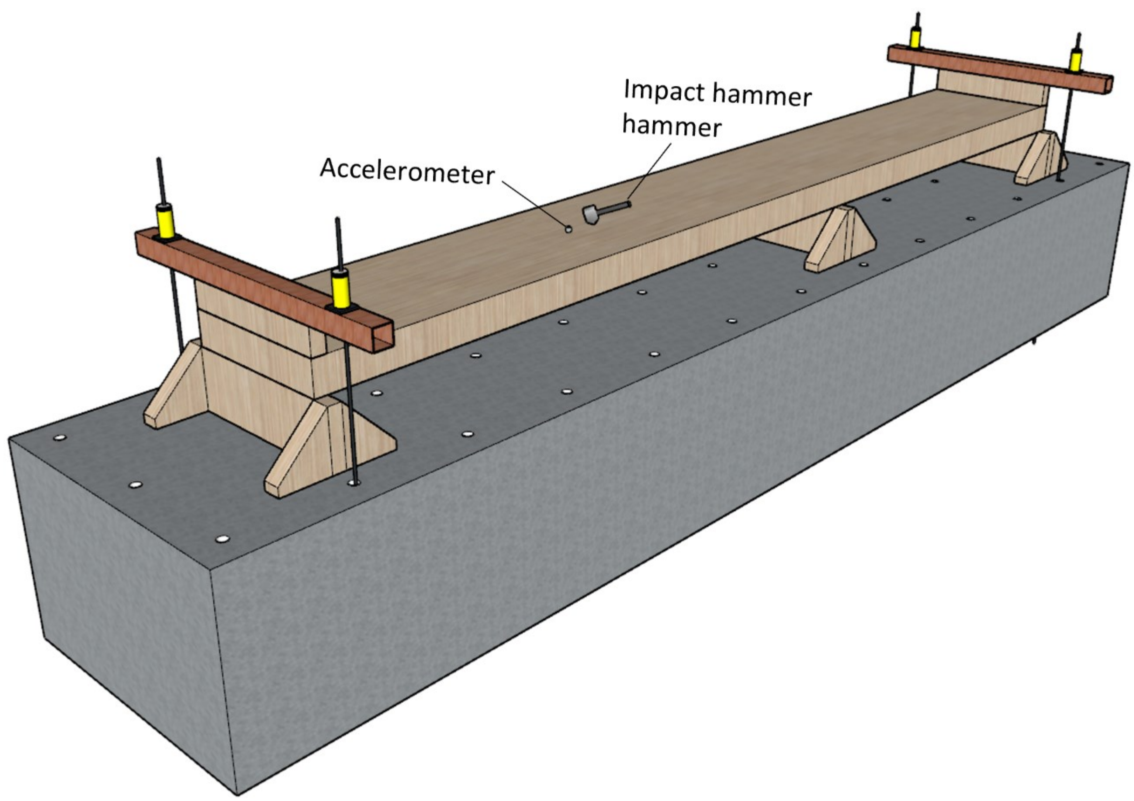


Figure 3-8 Test set-up of impact hammer modal test of two-span and wall supported floor

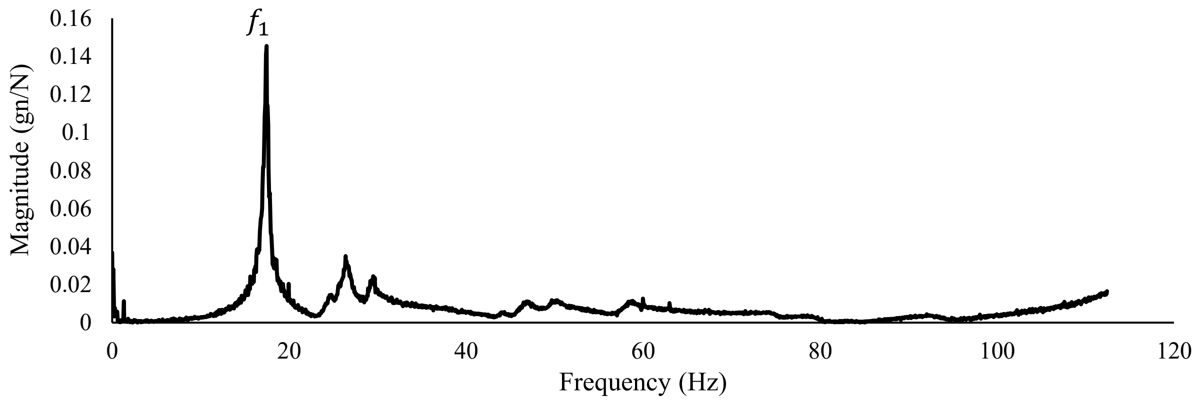


Figure 3-9 Typical plot of FRF

3.4 Test Results

Test results of beam supported floor panels are shown in Table 3-8. Results from the static load deflection test and impact hammer modal test of two-span and wall supported floor panels are shown in Table 3-9. Table 3-10 shows the results of refence and beam tests.

Table 3-8 Test results of beam-supported floor specimens

Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)	Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)
S-3LB-4877-S	1.26	12.63	T-3LB-3:2-S-3STS-S	-	17.38
S-3LB-4877-3STS	1.18	13.13	T-3LB-3:2-3STS-3STS-3STS	0.71	17.63
S-3LB-6096-S	2.04	8.63	T-2LB-3:2-S-S-S	-	12.75
S-3LB-6096-3STS	1.93	9.38	T-2LB-3:2-S-3STS-S	-	13.13
S-2LB-6096-S	2.65	7.50	T-2LB-3:2-3STS-3STS-3STS	1.03	13.25
S-2LB-6096-3STS	2.49	8.13	T-3LB-1:1-3STS-3STS-3STS	0.62	17.75
T-3LB-3:2-S-S-S	-	16.75	T-2LB-1:1-3STS-3STS-3STS	0.92	13.00

Table 3-9 Test results of two-span floor specimens

Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)	Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)
C105-S-S-S	0.99	18.13	C244-S-5STS+10.32-S	0.14	25.81
C105-S-3STS-S	0.92	19.50	C244-S-5STS+29.97-S	0.14	27.81
C105-S-5STS-S	0.90	19.50	C244-10.32-5STS-10.32	0.12	29.88
C105-S-5STS+10.32-S	0.87	19.50	C244-39.80-5STS-39.80	0.12	30.13
C105-S-5STS+29.97-S	0.85	19.63	G130-S-S-S	0.61	25.63
C105-S-5STS+49.62-S	0.86	19.63	G130-S-5STS+10.32-S	0.57	27.63
C105-5.41-5STS-5.41	0.83	20.38	G130-S-5STS+29.97-S	0.56	27.88
C105-10.32-5STS-10.32	0.77	20.75	G130-S-5STS+44.71-S	0.54	27.31
C105-20.15-5STS-20.15	0.77	21.06	G175-S-S-S	0.36	25.88
C105-29.97-5STS-29.97	0.72	21.38	G175-S-3STS-S	0.31	23.00
C105-39.80-5STS-39.80	0.73	21.44	G175-S-5STS-S	0.31	23.25
C105-49.62-5STS-49.62	0.70	21.63	G175-10.32-5STS-10.32	0.27	27.25
C175-S-S-S	0.28	20.75	G175-29.97-5STS-29.97	0.26	28.50
C175-S-3STS-S	0.24	22.25	G175-44.71-5STS-44.71	0.26	28.75
C175-S-5STS-S	0.24	22.75	G215-S-S-S	0.17	40.75
C175-S-5STS+29.97-S	0.23	22.63	G215-S-5STS-S	0.15	44.75
C175-S-5STS+39.80-S	0.23	22.50	G215-10.32-5STS-10.32	0.12	46.75
C175-10.32-5STS-10.32	0.24	23.00	G215-29.97-5STS-29.97	0.12	51.75
C244-S-S-S	0.18	23.06	G215-44.71-5STS-44.71	0.12	48.00
C244-S-5STS-S	0.14	24.94			

Table 3-10 Reference test and beam test results

Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)	Floor ID	Deflection under 1 kN (mm)	f_1 (Hz)
G175-6096-S	1.51	11.50	G175-3658-44.71	0.27	34.00
G175-6096-3STS	1.41	12.38	G215-3658-S	0.21	34.69
G175-4877-S	0.79	17.13	G215-3658-10.32	0.18	37.75
G175-4877-3STS	0.75	18.50	G215-3658-29.97	0.17	40.75
C105-3658-S	1.24	16.56	G215-3658-44.71	0.16	42.00
C105-3658-5.41	1.10	17.19	C105-2438-S	0.42	33.88
C105-3658-10.32	1.03	18.19	C105-2438-5.41	0.35	34.38
C105-3658-20.15	0.93	19.38	C105-2438-10.32	0.36	30.25
C105-3658-29.97	0.88	20.25	C105-2438-20.15	0.29	36.25
C105-3658-39.80	0.86	20.94	C105-2438-29.97	0.29	37.50
C105-3658-49.62	0.85	21.50	C105-2438-39.80	0.26	38.50
C175-3658-S	0.33	23.75	C105-2438-49.62	0.26	39.13
C175-3658-10.32	0.30	25.56	C175-2438-S	0.12	46.56
C244-3658-S	0.16	29.00	C175-2438-10.32	0.11	57.44
C244-3658-10.32	0.15	30.31	2LB-2	2.48	59.38
C244-3658-39.80	0.14	32.50	2LB-3	2.70	56.63
G175-3658-S	0.36	26.00	2LB-4	2.46	54.63
G175-3658-3STS	0.31	31.25	3LB-1	1.12	60.63
G175-3658-10.32	0.29	30.75	3LB-2	1.19	67.25
G175-3658-29.97	0.26	33.38	3LB-3	1.18	64.25

3.5 Discussion of test results

3.5.1 Two-span CLT and GLT floors supported on walls

3.5.1.1 Effects of fixity of middle support on static deflection and first natural frequency

The fixity of the middle support alters as a result of using STS and applying top load over in a series of tests on a two-span floor with the first and third supports remaining simple. As the number of STS and top load increases, the fixity of middle support increases. Figures 3-10 and 3-11 show the effect of fixity of the middle support on static deflection and first natural frequency respectively. The x-axes are the boundary conditions of the middle support, where S presents the simple support, number of STS used, and the number presents the top load generated by hydraulic jack pressure in kN. In both figures the end fixity of the test specimens can be considered to increase as we go from S specimen on the left to 5STS+49.62 specimen.

As Figure 3-10 shows, as the fixity increases in the middle support, both CLT and GLT floor panels exhibit reduced deflections. In contrast, the first natural frequency increases as the middle support fixity increases. Both smaller deflections and greater natural frequencies are signs of the floor panel getting stiffer. Generally, the static deflection and first natural frequency are not very sensitive to changes in the fixity of the middle support when the fixity of middle support is already high. Both parameters change more noticeably between S and 5STS+10.32. Beyond 5STS+10.32, the trend line remains horizontal.

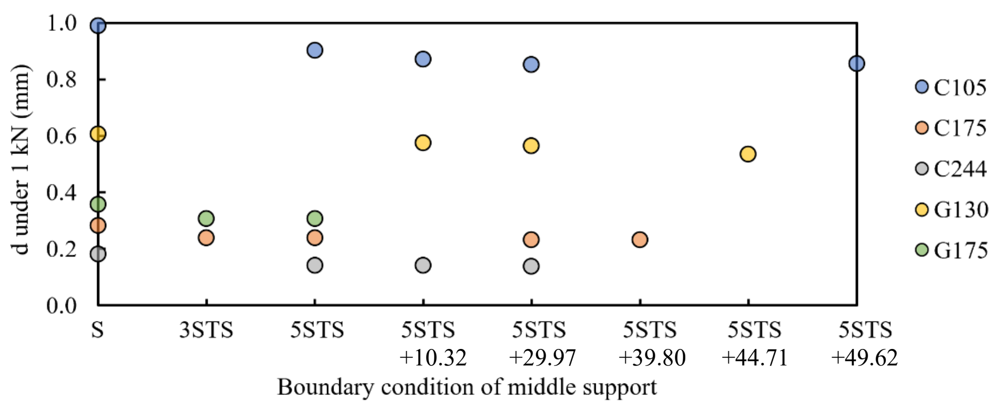


Figure 3-10 Effect of fixity of middle support on static deflection - simple condition 1st and 3rd support

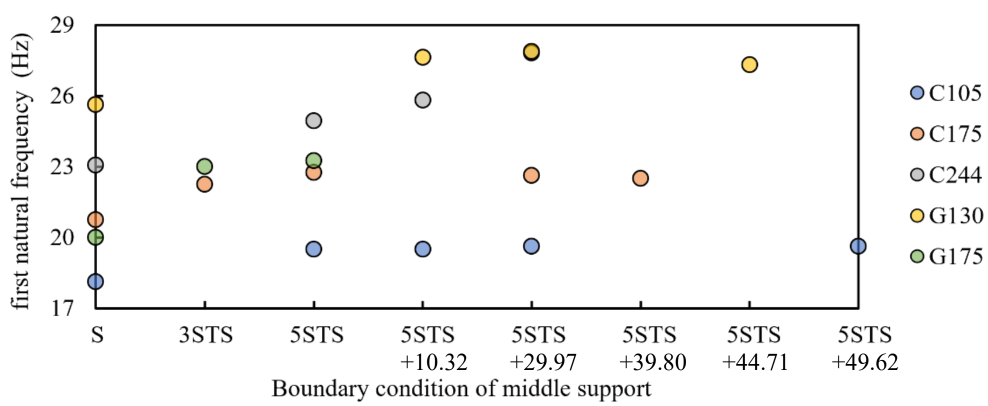


Figure 3-11 Effect of fixity of middle support on first natural frequency - simple condition 1st and 3rd support

3.5.1.2 Effects of fixity of first and third supports on static deflection and first natural frequency

The fixity of the first and third supports (the end supports of longer span and short span) increases by applying top load over the middle support connected by 5STS. Figure 3-12 shows the effect of top load on the static deflection. The x-axis shows the boundary conditions of first and third supports, where S presents the simple support, and the number presents the top load generated by hydraulic jack pressure in kN. As the top load increases, the corresponding static deflection decreases. For the panel with smallest thickness (C105), the effect is the most significant.

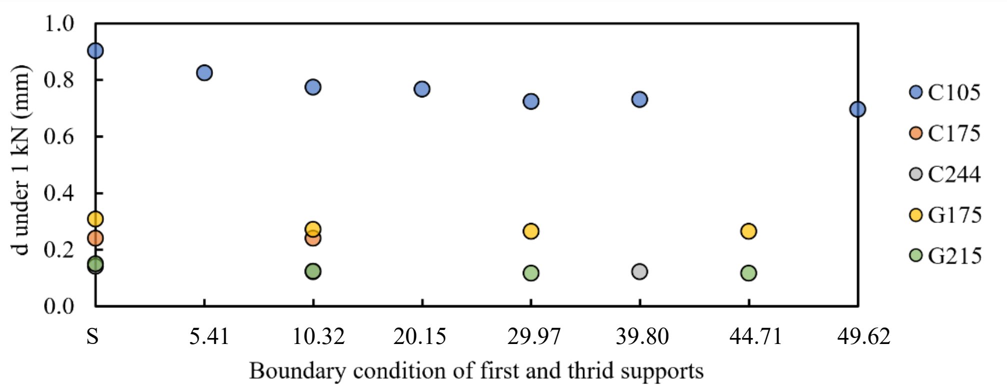


Figure 3-12 Effect of top load over 1st and 3rd supports on static deflection - 5 STS at middle support without top load

Figure 3-13 shows the effect of top load over end supports on the natural frequency, where the x-axis is the same as Figure 3-12. The natural frequency increases after the increase of the top load.

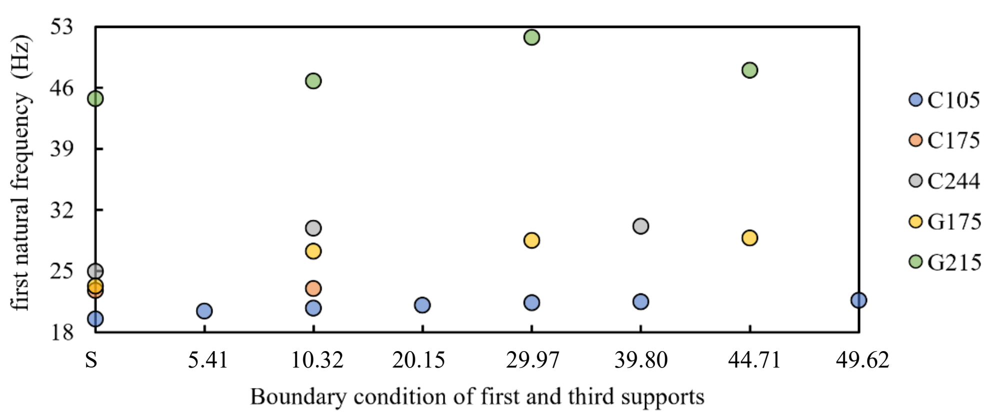


Figure 3-13 Effect of top load over 1st and 3rd supports on natural frequency - 5 STS at middle support

3.5.2 GLT floors supported on beams

3.5.2.1 Effects of rotational restraint in supports on static deflection and natural frequency of one-span and beam-supported floors

Three one-span floors supported by beams were tested under two different connections, including simple support and 3 STS. Their corresponding deflections and frequencies are listed in Table 3-11. It is evident that the rotational restraint caused by 3STS can help lower the static deflection and raise fundamental natural frequency. The change in deflection and frequency is smaller than 10%.

Table 3-11 Deflection and first natural frequency of one-span and beam-supported floor with different rotational restraint

Span (mm)	Beam type	d under 1 kN (mm)			First natural frequency (Hz)		
		Simple	3STS	Difference	Simple	3STS	Difference
4877	3LB	1.257	1.176	-6.42%	12.625	13.125	3.96%
6096	3LB	2.037	1.930	-5.26%	8.625	9.375	8.70%
6096	2LB	2.650	2.489	-6.09%	7.500	8.125	8.33%

3.5.2.2 Effects of supporting beam stiffness on static deflection and natural frequency of one-span and beam-supported floors

The influence of supporting beam stiffness was studied by using lumber beams with different number of lumber laminations. The 6096 mm long GLT floor was tested when supported by 2-layer and 3-layer lumber beam respectively. The beam stiffnesses of supporting lumber beams are shown in Table 3-12 based on Eq. (3.1).

$$K = \frac{P}{d} = \frac{384EI}{(8L^3 - 4Lw^2 + w^3)} \tag{3.1}$$

where K is the beam stiffness, L is span of the lumber beam, w is the width of loading area and EI is the flexural stiffness. As Figure 3-7 shows, the loading width is the width of the 22.7 kg (50-lb) weight, which is 250 mm.

The effects of flexural stiffness on static deflection and first natural frequency are significant, as Table 3-13 shows. The floor tests include two different support conditions, simple support and

3STS at 200 mm. As the stiffness of supporting beams doubles, the deflection decreases by about 23% and the first natural frequency increases by about 15% for both connections.

Table 3-12 Stiffness of 2-layer and 3-layer lumber beams used in one-span floor test

Beam I.D.	d under 1 kN (mm)	Beam Stiffness (kN/mm)	Flexural Stiffness ($kNmm^2$)	Average Flexural Stiffness ($kNmm^2$)
2LB-2	2.48	0.42	5.31E+07	5.33E+07
2LB-4	2.46	0.42	5.35E+07	
3LB-2	1.19	0.88	1.11E+08	1.11E+08
3LB-3	1.18	0.89	1.12E+08	

Table 3-13 Deflection and frequencies of one-span and beam-supported floor with two supporting beam stiffnesses

Beam type	Flexural stiffness ($kNmm^2$)	d under 1 kN (mm)		First natural frequency (Hz)	
		Simple	3STS	Simple	3STS
2LB	5.33E+07	2.65	2.49	7.50	8.13
3LB	1.11E+08	2.04	1.93	8.63	9.38
Difference (%)	108.94	-23.14	-22.47	15.00	15.39

3.5.2.3 Effects of supporting beam stiffness on static deflection and natural frequency of two-span floors

There are four two-span tests discussed in this section. All four tests have connections of 3 STS at 200 mm on all three supports. The two-span floor panel was supported by 2-layer and 3-layer lumber beams separately. The total length of two-span floor is 6096 mm. The two span ratios considered in this study are 3:2 and 1:1, which means the span lengths of each set-up is 3658 mm:2438 mm and 3048 mm:3048 mm. The stiffnesses of supporting lumber beams are shown in Table 3-14. The comparison between deflection and frequency is shown in Table 3-15 and Table 3-16.

Table 3-14 Stiffness of 2-layer and 3-layer lumber beams used in two-span floor tests

Beam I.D.	d under 1 kN (mm)	Beam Stiffness (kN/mm)	Flexural Stiffness ($kNmm^2$)	Average Flexural Stiffness ($kNmm^2$)
2LB-2	2.48	0.42	5.31E+07	
2LB-3	2.70	0.39	4.88E+07	5.18E+07
2LB-4	2.46	0.42	5.35E+07	
3LB-1	1.12	0.94	1.18E+08	
3LB-2	1.19	0.88	1.11E+08	1.14E+08
3LB-3	1.18	0.89	1.12E+08	

Table 3-15 Deflection of two-span and beam-supported floors with two supporting beam stiffnesses

Beam type	Flexural stiffness of beam ($kNmm^2$)	d under 1 kN (mm)	
		3:2	1:1
2LB	5.18E+07	1.03	0.92
3LB	1.14E+08	0.71	0.63
Difference (%)	119.27	-31.48	-31.87

Table 3-16 First natural frequency of two-span and beam-supported floors with two supporting beam stiffnesses

Beam type	Flexural stiffness of beam ($kNmm^2$)	S-S-S	S-3STS-S	3STS-3STS-3STS	
		3:2	3:2	3:2	1:1
2LB	5.18E+07	12.75	13.13	13.25	13.00
3LB	1.14E+08	16.75	17.38	17.63	17.75
Difference (%)	119.27	31.37	32.37	33.06	36.54

It can be observed that when the stiffness of supporting beams increases by almost 120%, the deflection decreases by 31% for both span ratios. Meanwhile, the first natural frequency increases by more than 30% for three boundary conditions and two span ratios.

3.5.2.4 Effects of span ratio on static deflection and natural frequency of two-span and beam-supported floors

It can be observed from Table 3-15 that with the same total length the floor with a span ratio closer to 1 has a smaller deflection. However, the effect of the span ratio on natural frequency is insignificant.

3.6 Summary

A total of 39 two-span mass timber floor specimens and 14 beam-supported GLT floor specimens were tested. The two-span mass timber floor specimens varied in the MTP product type, panel thickness, and connection type. The beam-support MTP floor specimens contained single-span and two-span layouts; they varied in span length (in one-span cases only), boundary condition (in one-span cases only), and beam stiffness.

A test set-up was built to meet the needs of this testing program, and various instruments were adopted to measure and record the behaviour of specimens. The static deflection at mid-span and fundamental natural frequency from each test were recorded.

The test results revealed that for the two-span and wall-supported floors the fixity of connections can affect deflection and frequency, especially when the fixity is low. The boundary condition and supporting beam stiffness of two-span and beam-supported floor could affect the two parameters significantly. In contrast, the effect span ratio is small based on limited test data.

Chapter 4 - Analytical Models for Calculating Mid-span Deflection and Fundamental Natural Frequency of Two-Span Floor with Restrained Connections

4.1 Introduction

In the literature, inadequate attention has been paid to the effects of two-span configuration and connection restraints on the mid-span deflection and natural frequency of MTP floors, which is the focus of this chapter. In this Chapter formulae have been developed to predict mid-span deflection and fundamental natural frequency of a one-way MTP floor by modifying the existing formulae in the literature for mid-span deflection under a point load and natural frequency of rotationally restrained beams. All the formulae have been validated against the test results from the previous chapter.

4.2 Mid-span Deflection Equation of Two-span Floor with Restrained Connections

As discussed in Section 2.2.1, Zhang et al (2019) assumed CLT timber floors to be Euler–Bernoulli beams in the study of end support restraint effects on vibration performance. The mid-span deflection under a point load of a rotationally restrained, single-span beam, d, is determined by Eq. (4.1):

$$d = C_{d\_rr} \frac{PL^3}{48EI} \tag{4.1}$$

where P is the point load applied to the beam, L and EI are the span and flexural stiffness of the beam respectively, and C_{d_rr} is the coefficient of deflection accounting for the effect of both end restraints, which can be expressed as

$$C_{d\_rr} = \frac{5r_1r_2 - 9r_1 - 9r_2 + 16}{16 - 4r_1r_2} \tag{4.2}$$

where $r_1$and r_2 are the fixity factors of the beam's two ends. The fixity factor, r, was introduced by Monforton and Wu (1963) as a measure of the level of rotational restraint, which is expressed as Eq. (2.2). A zero value for r represents the simple support condition while r=1 means a fully fixed boundary condition. Using the fixity factor in the calculation rather than the rotational stiffness provides a direct measure on how the boundary condition would perform relative to the two extreme support conditions of simple and fully fixed. Besides, the use of fixity factor facilitates matrix analysis method, which is adopted in this study.

Figure 4-1 shows a typical two-span, rotationally restrained beam. The left span is the longer span with a length, L. The right span has a length of μL, where μ is the span ratio and has maximum value of one. The three supports of the beam have rotational stiffness R_1, R_2, and R_3 and fixity factor r_1, r_2, and r_3.

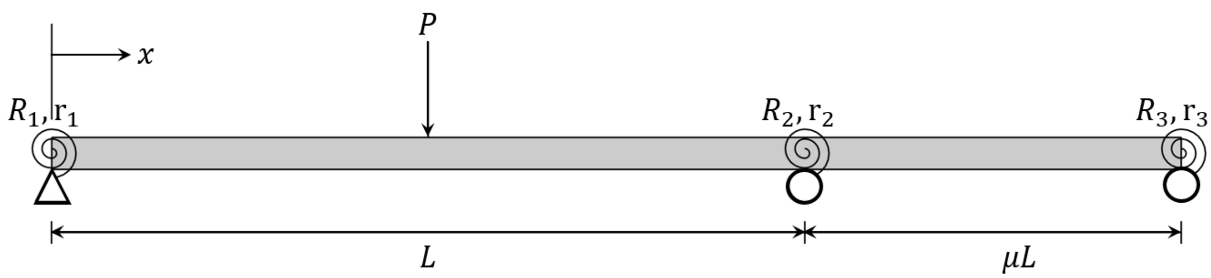


Figure 4-1 A two-span, rotationally restrained beam with a point load applied to mid-span of the longer span

Similar to Eq (4.1), the final deflection equation contains a coefficient accounting for the effect of restrained connections of a simply supported two-span beam with the same configuration as Figure 4-1, as shown in Eq. (4.3),

$$d = C_{d\_rrr} \frac{(16\mu + 7)PL^3}{768(\mu + 1)EI} \tag{4.3}$$

According to Eq. (2.2), the fixity factors of the three supports can be calculated as follows,

$$r_1 = \frac{1}{1 + 3\frac{EI}{R_1 L}}, \qquad r_2 = \frac{1}{1 + 3\frac{EI}{R_2 L}}, \qquad and, \qquad r_3 = \frac{1}{1 + 3\frac{EI}{R_3 \mu L}} \tag{4.4}$$

As Figure 4-2 shows, the system consists of three elements with length of $L/_2$, $L/_2$, and μL respectively. The structure has 5 degrees of freedom (DOFs). The point load is always considered applying to middle of L, because applying at this location will cause the maximum deflection.

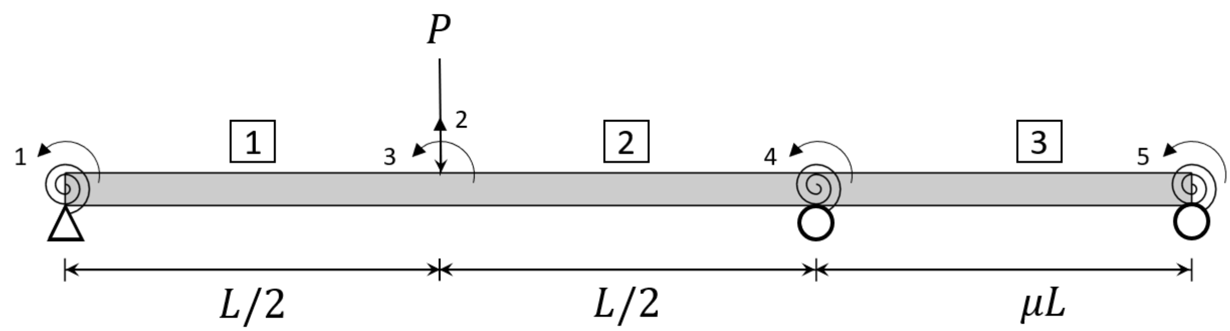


Figure 4-2 DOFs and elements of a two-span and rotationally restrained beam with a point load applied

The stiffness matrix of the entire beam including rotational stiffnesses (R_1, R_2 and R_3) is shown in Eq. (4.5).

$$K = \begin{bmatrix} \frac{8EI}{L} + R_1 & -\frac{24EI}{L^2} & \frac{4EI}{L} & 0 & 0 \\ -\frac{24EI}{L^2} & \frac{192EI}{L^3} & 0 & \frac{24EI}{L^2} & 0 \\ \frac{4EI}{L} & 0 & \frac{16EI}{L} & \frac{4EI}{L} & 0 \\ 0 & \frac{24EI}{L^2} & \frac{4EI}{L} & \frac{8EI}{L} + \frac{4EI}{\mu L} + R_2 & \frac{2EI}{\mu L} \\ 0 & 0 & 0 & \frac{2EI}{\mu L} & \frac{4EI}{\mu L} + R_3 \end{bmatrix} \quad (4.5)$$

The stiffness matrices of the three beam elements are shown below:

$$K_1 = K_2 = \begin{bmatrix} \frac{96EI}{L^3} & \frac{24EI}{L^2} & -\frac{96EI}{L^3} & \frac{24EI}{L^2} \\ \frac{24EI}{L^2} & \frac{8EI}{L} & -\frac{24EI}{L^2} & \frac{4EI}{L} \\ -\frac{96EI}{L^3} & -\frac{24EI}{L^2} & \frac{96EI}{L^3} & -\frac{24EI}{L^2} \\ \frac{24EI}{L^2} & \frac{4EI}{L} & -\frac{24EI}{L^2} & \frac{8EI}{L} \end{bmatrix}$$

$$K_3 = \begin{bmatrix} \frac{12EI}{(\mu L)^3} & \frac{6EI}{(\mu L)^2} & -\frac{12EI}{(\mu L)^3} & \frac{6EI}{(\mu L)^2} \\ \frac{6EI}{(\mu L)^2} & \frac{4EI}{\mu L} & -\frac{6EI}{(\mu L)^2} & \frac{2EI}{\mu L} \\ -\frac{12EI}{(\mu L)^3} & -\frac{6EI}{(\mu L)^2} & \frac{12EI}{(\mu L)^3} & -\frac{6EI}{(\mu L)^2} \\ \frac{6EI}{(\mu L)^2} & \frac{2EI}{\mu L} & -\frac{6EI}{(\mu L)^2} & \frac{4EI}{\mu L} \end{bmatrix} \tag{4.6}$$

The force vector can be expressed as

$$F = \begin{bmatrix} 0 \\ P \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.7}$$

The deflection under the point load, along the DOF 2 in Figure 4-2, can be found as the second element in the displacement vector by solving the system stiffness matrix and force vector. After substituting rotational stiffnesses with fixity factors using Eq. (4.4) for simplification purposes, the deflection coefficient is obtained as shown in Eq. (4.8),

$$C_{d\_rrr} = \frac{(4+4\mu)\big(4(7-4r_1)(1-r_2)+\mu(16-9r_1-9r_2+5r_1r_2)(4-r_3)\big)}{(7+16\mu)\big(4(4-r_1)(1-r_2)+\mu(4-r_1r_2)(4-r_3)\big)} \tag{4.8}$$

4.3 Natural Frequency Equation of Two-span Floor with Restrained Connections

In the natural frequency study, the two-span floors are also considered as Euler-Bernoulli beams. Zhang et al (2019) introduced a Rayleigh-Morleigh formula (Petersen and Werkle 2018) to approximate the fundamental natural frequency of a one-span uniform beam with constant cross section and rotationally restrained supports. The formula is expressed as

$$\omega^2 = \frac{\int_0^L q|y(x)|\,dx}{\int_0^L \rho A y^2(x)\,dx} \tag{4.9}$$

where, ω is the natural circular frequency, $y(x)$ is the static displacement function, L is the span of the beam, q is the uniformly distributed load, ρ is the density and A is the cross-sectional area. After converting circular frequency to frequency, the formula becomes

$$f = \frac{1}{2\pi}\sqrt{\frac{\int_0^L q|y(x)|\,dx}{\int_0^L \rho A y^2(x)\,dx}} \tag{4.10}$$

The formula was developed from Rayleigh's quotient based on the static displacement under a uniformly distributed force, whose shape closely resembles the first mode shape of one span beam. For a two-span beam, the Reyleigh-Morleigh formula above is still applicable, except uniformly distributed forces applying in the opposite directions to each span, as shown in Figure 4-3. The uniform load, q, applies in opposite directions because the deflection shape of the beam is similar to the first mode shape of a two-span beam. The load, q, will cancel out in Eq. (4.10).

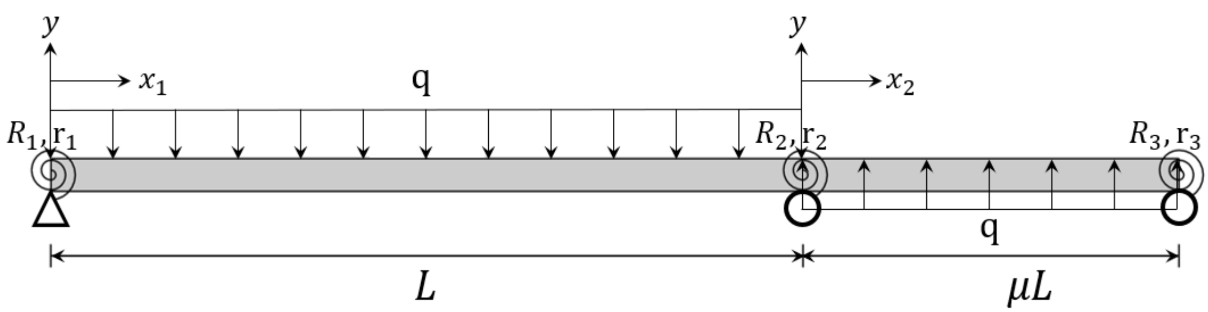


Figure 4-3 A two-span, rotationally restrained beam under uniformly distributed load

The static displacement functions were derived from the governing equation of a Euler-Bernoulli beam. They are in the form of

$$y(x_1) = \frac{-q}{24EI}{x_1}^4 + \frac{{x_1}^3}{6}C_1 + \frac{{x_1}^2}{x}C_2 + x_1C_3 + C_4$$

$$y(x_2) = \frac{q}{24EI}{x_2}^4 + \frac{{x_2}^3}{6}C_5 + \frac{{x_2}^2}{x}C_6 + x_2C_7 + C_8 \tag{4.11}$$

where $y(x_1)$ and $y(x_2)$ are static transverse displacement functions for the first span and second span respectively. C_1 to C_8 are the constants to be solved by applying the boundary conditions, which are listed in Table 4-1.

Table 4-1 Boundary Conditions of Two-span Rotationally Restrained Beam

Explanations	Boundary Conditions
Displacement @ support 1	$y(x_1 = 0) = 0$
Displacement @ support 2	$y(x_1 = L) = 0$
Displacement @ support 2	$y(x_2 = 0) = 0$
Displacement @ support 3	$y(x_2 = \mu L) = 0$
Moment @ support 1	$EI\frac{d^2y(x_1=0)}{dx_1^2} = R_2\frac{dy(x_1=0)}{dx_1}$
Moment @ support 2	$EI\frac{d^2y(x_1=L)}{dx_1^2} - EI\frac{d^2y(x_2=0)}{dx_2^2} + R_2\frac{dy(x_1=L)}{dx_1} = 0$
Moment @ support 3	$-EI\frac{d^2y(x_2=\mu L)}{dx_2^2} = R_2\frac{dy(x_2=\mu L)}{dx_2}$
Slope @ support 2	$\frac{dy(x_1=L)}{dx_1} = \frac{dy(x_2=0)}{dx_2}$

Eq. (4.11) can now be solved by applying the eight boundary conditions.

The constants C_1 to C_8 are solved as below:

$C_1 = Lq\big(12(1-r_2) + \mu(4+r_1-r_2-r_1r_2)(4-r_3) + \mu^3(2+r_1)(1-r_2)(2-r_3)\big)/(2EIM)$

$C_2 = -L^2qr_1N/(4EIM)$

$C_3 = -L^3q(1-r_1)N/(12EIM)$

$C_4 = 0$

$C_5 = -Lq((2-r_1)(1-r_2)(2+r_3) - \mu^2(4-r_1)(1-r_2)(4-r_3) - \mu^3(4-r_1r_2)(5-2r_3))/$
$(2\mu EIM)$

$C_6 = -L^2q(4(2-r_1)(1-r_2) - \mu^3(4-r_1r_2)(2-r_3))/(4EIM)$

$C_7 = L^3q\mu(1-r_2)((2-r_1)(4-r_3) + \mu^2(4-r_1)(2-r_3))/(12EIM)$

$C_8 = 0$ (4.12)

where,

$M = 4(4-r_1)(1-r_2) + \mu(4-r_1r_2)(4-3r_3)$

$$N = 4 - 4r_2 + \mu(2 - r_2)(4 - r_3) + 2\mu^3(1 - r_2)(2 - r_3) \tag{4.13}$$

Following from the above, the fundamental natural frequency equation, f_{rrr}, of a two-span, rotationally restrained beam can be expressed as the product of a coefficient, C_{f_rrr}, which accounts for the effect of restrained boundary conditions, and the natural frequency equation of a simply supported two-span beam with the same configuration, f_{sss}. f_{rrr} is obtained by substituting Eq. (4.11) and Eq. (4.12) into Eq. (4.10).

$$f_{rrr} = C_{f\_rrr} \times f_{sss} \tag{4.14}$$

where,

$$f_{sss} = 6.189 \times \sqrt{\frac{EI(3 + 8\mu + 10\mu^3 + 8\mu^5 + 3\mu^6)}{AL^4\rho(19 + 76\mu + 48\mu^2 + 9\mu^3 + 144\mu^4 - 96\mu^5 + 144\mu^6 + 9\mu^7 + 48\mu^8 + 76\mu^9 + 19\mu^{10})}}$$

Due to the complexity of coefficient C_{f_rrr}, the full expression is shown in Appendix A. C_{f_rrr} is a function of r_1, r_2, r_3, and μ. It will be further simplified in a future section in this chapter.

4.4 Model Validation

4.4.1 Verification of Mid-span Deflection and Fundamental Natural Frequency Equations against Test Results

The analytical expressions to calculate the mid-span deflection and fundamental natural frequency were verified against the laboratory test results presented in Chapter 3. The verification used the circular reasoning fallacy. The detailed verification process is shown in Figure 4-4 below.

As the one-span test branch shows in Figure 4-4, the primary purpose of conducting the one-span tests in Chapter 3 was to find the fixity factors of various boundary conditions. The fixity factors were calculated based on Eq. (2.2). Since all specimens of one-span reference tests have the same boundary conditions on both ends, the fixity factors of both boundary conditions should be the same. Eq. (2.2) was further simplified as

$$C_{d\_rr} = \frac{8 - 5r}{8 + 4r} \tag{4.15}$$

Rearranging the equation, the fixity factor can be calculated using

$$r = \frac{8 - 8C_{d\_rr}}{5 + 4C_{d\_rr}} \quad (4.16)$$

The results of fixity factors in the one-span tests with spans of 2438 mm and 3658 mm are shown in Table 4-2 and Table 4-3, respectively.

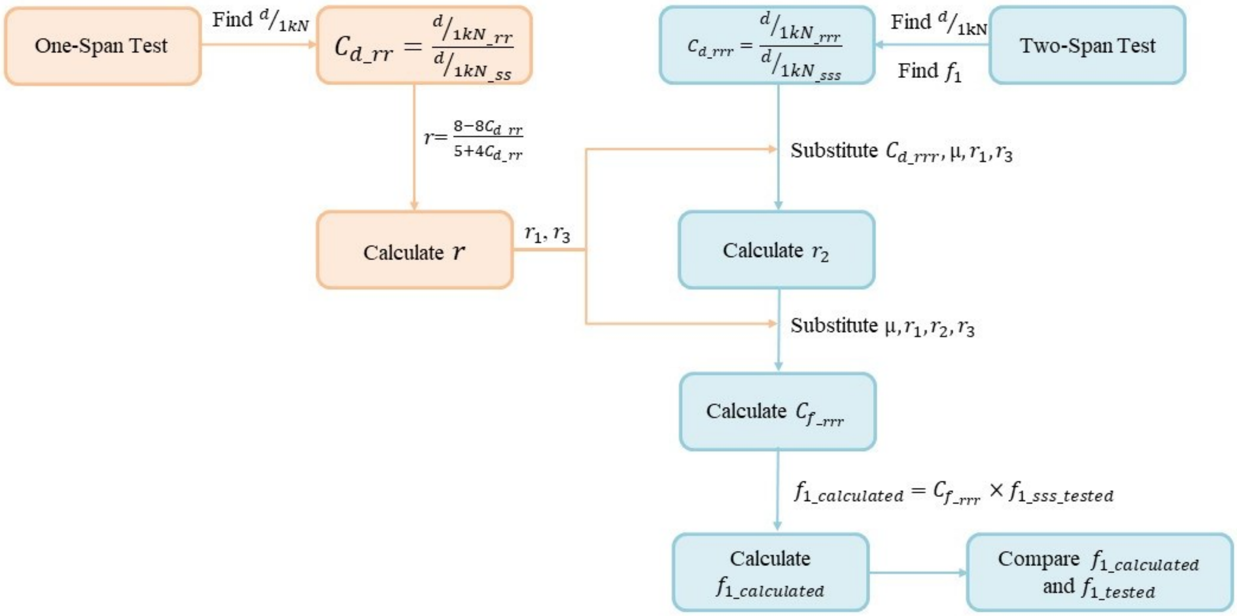


Figure 4-4 Flowchart of analytical expression verification using circular argument

Table 4-2 Fixity Factor of the One-span Reference Beam Tests of 2438 mm Span

Floor ID	Span (mm)	d under 1 kN (mm)	C_{d_rr}	Fixity Factor
C105-2438-S	2438	0.424	1.000	0.000
C105-2438-5.41	2438	0.345	0.812	0.182
C105-2438-10.32	2438	0.360	0.848	0.145
C105-2438-20.15	2438	0.292	0.689	0.321
C105-2438-29.97	2438	0.286	0.674	0.339
C105-2438-39.80	2438	0.257	0.605	0.425
C105-2438-49.62	2438	0.258	0.608	0.422
C175-2438-S	2438	0.121	1.000	0.000
C175-2438-10.32	2438	0.105	0.870	0.123

Table 4-3 Fixity Factor of the One-span Reference Beam Tests of 3658 mm Span

Floor ID	Span (mm)	d under 1 kN (mm)	C_{d_rr}	Fixity Factor
C105-3658-S	3658	1.244	1.000	0.000
C105-3658-5.41	3658	1.100	0.885	0.108
C105-3658-10.32	3658	1.029	0.827	0.167
C105-3658-20.15	3658	0.931	0.748	0.252
C105-3658-29.97	3658	0.882	0.709	0.297
C105-3658-39.80	3658	0.859	0.690	0.319
C105-3658-49.62	3658	0.851	0.685	0.326
C175-3658-S	3658	0.330	1.000	0.000
C175-3658-10.32	3658	0.296	0.895	0.098
C244-3658-S	3658	0.163	1.000	0.000
C244-3658-10.32	3658	0.146	0.897	0.096
C244-3658-39.80	3658	0.137	0.845	0.148
G175-3658-S	3658	0.359	1.000	0.000
G175-3658-3STS	3658	0.294	0.818	0.176
G175-3658-10.32	3658	0.258	0.719	0.285
G175-3658-29.97	3658	0.265	0.739	0.262
G175-3658-44.71	3658	0.212	1.000	0.000
G215-3658-S	3658	0.183	0.865	0.128
G215-3658-10.32	3658	0.169	0.797	0.198
G215-3658-29.97	3658	0.158	0.749	0.251
G215-3658-44.71	3658	0.163	0.897	0.096

In the two-span test, all the test specimens have a span of 3658 mm, which is the same as the span length of the one-span tests shown in Table 4-2. When the floor panel, supporting wall and connection type are the same, the fixity factor in Table 4-2 should be equal to r_1 of the two-span tests. Similarly, the fixity factor in Tables 4-3 can substitute for r_3 of the corresponding two-span tests. In addition, the spans of all two-span specimens remained the same, and the span ratio, μ, remained constant at 2:3 throughout the verification process. After substituting the value of μ, Eq. (4.8) becomes,

$$C_{d\_rrr} = \frac{20(106 - 60r_1 - 78r_2 - 16r_3 + 44r_1r_2 + 9r_1r_3 + 9r_2r_3 + 15r_1r_2r_3)}{53(40 - 6r_1 - 24r_2 - 4r_3 + 2r_1r_2 + r_1r_2r_3)} \quad (4.17)$$

Rearranging Eq. (4.17), r_2 can be calculated using Eq. (4.18),

$$r_2 = \frac{20(106 - 60r_1 - 16r_3 + 9r_1r_3) + 106C_{d_{rrr}}(-20 + 3r_1 + 2r_3)}{20(78 - 44r_1 - 9r_3 + 5r_1r_3) + 53C_{d_{rrr}}(-24 + 2r_1 + r_1r_3)} \quad (4.18)$$

The results of r_2 value of 105 mm and 175 mm thick CLT floor panel tests are shown in Table 4-4. With the known r_1, r_2, r_3 and μ, the frequency coefficient, C_{f_rrr}, can be calculated using the C_{f_rrr} equation in Appendix A. Table 4-5 shows the fundamental natural frequency of 105 mm and 175 mm thick CLT floor panel tests, including predicted ones obtained from analytical expressions and ones obtained from the experiments.

Table 4-4 Fixity Factor of the Middle Support (r_2) of 105 mm and 175 mm thick CLT floor panel tests calculated from r_1 and r_3.

Floor ID	d under 1 kN (mm)	$C_{d_{rrr}}$	r_1	r_3	r_2
C105-S-S-S	0.988				
C105-S-3STS-S	0.923	0.934	0.000	0.000	0.376
C105-S-5STS-S	0.903	0.914	0.000	0.000	0.461
C105-S-5STS+10.32-S	0.871	0.881	0.000	0.000	0.574
C105-S-5STS+29.97-S	0.851	0.861	0.000	0.000	0.633
C105-S-5STS+49.62-S	0.855	0.866	0.000	0.000	0.620
C105-1.38-5STS-5.41	0.825	0.835	0.108	0.182	0.592
C105-10.32-5STS-10.32	0.774	0.784	0.167	0.145	0.692
C105-20.15-5STS-20.15	0.767	0.776	0.252	0.321	0.613
C105-29.97-5STS-29.97	0.723	0.732	0.297	0.339	0.710
C105-39.80-5STS-39.80	0.731	0.740	0.319	0.425	0.650
C105-49.62-5STS-49.62	0.696	0.704	0.326	0.422	0.759
C175-S-S-S	0.283				
C175-S-3STS-S	0.239	0.844	0.000	0.000	0.680
C175-S-5STS-S	0.239	0.845	0.000	0.000	0.676
C175-S-5STS+29.97-S	0.233	0.822	0.000	0.000	0.733
C175-S-5STS+39.80-S	0.233	0.822	0.000	0.000	0.734
C175-10.32-5STS-10.32	0.239	0.845	0.098	0.123	0.575

Table 4-5 Comparison between tested and predicted fundamental natural frequency for 105 mm and 175 mm thick CLT floor

Floor ID	Tested C_{frrr}	Predicted C_{frrr}	Tested f_1 (Hz)	Predicted f_1 (Hz)	Difference in f_1 (%)
C105-S-S-S			18.125		
C105-S-3STS-S	1.076	1.048	19.500	18.997	-2.580
C105-S-5STS-S	1.076	1.064	19.500	19.291	-1.071
C105-S-5STS+10.32-S	1.076	1.091	19.500	19.781	1.441
C105-S-5STS+29.97-S	1.076	1.109	19.625	20.098	2.408
C105-S-5STS+49.62-S	1.083	1.105	19.625	20.024	2.034
C105-1.38-5STS-5.41	1.083	1.129	20.375	20.468	0.455
C105-10.32-5STS-10.32	1.124	1.177	20.750	21.337	2.830
C105-20.15-5STS-20.15	1.145	1.182	21.063	21.426	1.726
C105-29.97-5STS-29.97	1.162	1.230	21.375	22.293	4.295
C105-39.80-5STS-39.80	1.179	1.219	21.438	22.093	3.056
C105-49.62-5STS-49.62	1.193	1.262	21.625	22.875	5.779
C175-S-S-S			20.750		
C175-S-3STS-S	1.072	1.124	22.250	23.333	4.868
C175-S-5STS-S	1.096	1.123	22.750	23.305	2.442
C175-S-5STS+29.97-S	1.090	1.145	22.625	23.756	4.998
C175-S-5STS+39.80-S	1.084	1.146	22.500	23.770	5.645
C175-39.80-5STS-39.80	1.108	1.120	23.000	23.245	1.064

Table 4-5 shows that most of the predicted fundamental natural frequency values are greater than the test results. However, a good agreement between the tested and predictions can still be observed due to the differences being within 6%.

4.4.2 Sensitivity Analysis and Simplified Equations

As discussed in the previous section, obtaining the fixity factors of the two end supports of the two-span floor panels requires two separate one-span floor panel tests. These test specimens span the same as each span of two-span panels. Due to limited test material and time, the one-span reference tests cannot be conducted entirely for the rest of the two-span floor specimens.

Eliminating parameters in the analytical expression can be an effective way to simplify the expressions and it can potentially decrease the one-span reference tests. In order to find the best parameter to eliminate from the expressions, two sensitivity analyses have been performed of Eq. (4.8) and the equation of C_{f_rrr}. Both equations contain four parameters and all of them range between 0 and 1. The sensitivity analyses adopted the method of Sobol and performed by SALib, an open-source library in Python. The detailed Python code is included in Appendix A. The results from the sensitivity analyses are shown in Table 4-6.

Table 4-6 Sensitivity analysis results of C_{d_rrr} and C_{f_rrr} equations

C_{d_rrr}	First-order indices	Second-order indices	Total-order indices	C_{f_rrr}	First-order indices	Second-order indices	Total-order indices
μ	0.1056	-	0.1297	μ	0.1875	-	0.2622
r_1	0.7542	-	0.7633	r_1	0.5855	-	0.6069
r_2	0.1072	-	0.1381	r_2	0.1453	-	0.1958
r_3	0.0014	-	0.0021	r_3	0.0072	-	0.0180
μ, r_1	-	0.00053	-	μ, r_1	-	0.01668	-
μ, r_2	-	0.02202	-	μ, r_2	-	0.04779	-
μ, r_3	-	0.00009	-	μ, r_3	-	0.00801	-
r_1, r_2	-	0.00686	-	r_1, r_2	-	0.00045	-
r_1, r_3	-	0.00006	-	r_1, r_3	-	0.00026	-
r_2, r_3	-	0.00036	-	r_2, r_3	-	-0.00117	-

There is a negative value of second-order index of C_{f_rrr} equation. It is the computing errors due to the limited sample size and very little second-order effects. It is obvious that the first-order and total-order indices of parameter r_3 are far smaller than the other parameters in both equations. The second-order indices of r_3 and other parameters are generally small, which means the higher-order interactions between r_3 and other parameters are weak. The small total-order indices of r_3 are also indicating the weak higher-order interactions. In conclusion, r_3 appears to have little first-order and second-order effects in both equations. Therefore, the reference tests with the shorter span, which provide the value of r_3, were omitted. From Table 4-4, it can be observed that almost all the values of fixity factor r_3 are greater than the corresponding r_1. In the validation process, it is conservative to substitute the value r_3 with r_1.

In order to show the validity of substituting r_3 with r_1, verifying the analytical expressions against the two-span tests of 105 mm and 175 mm thick CLT floor panels has been repeated. The results of the new verification are shown in Table 4-7 and Table 4-8.

Table 4-7 Comparison between fixity factors of middle support calculated using r_1 and r_3 values and r_1 values only for 105 mm and 175 mm thick CLT floor panel

Floor ID	d under 1 kN (mm)	$C_{d_{rrr}}$	$r_3 = r_1$	r_2 ($r_3 = r_1$)	r_2 ($r_3 \neq r_1$)	Difference in r_2 (%)
C105-5.41-5STS-5.41	0.825	0.835	0.108	0.592	0.597	0.836
C105-10.32-5STS-10.32	0.774	0.784	0.167	0.692	0.691	-0.120
C105-20.15-5STS-20.15	0.767	0.776	0.252	0.613	0.617	0.728
C105-29.97-5STS-29.97	0.723	0.732	0.297	0.710	0.711	0.220
C105-39.80-5STS-39.80	0.731	0.740	0.319	0.650	0.656	0.900
C105-49.62-5STS-49.62	0.696	0.704	0.326	0.759	0.762	0.332
C175-10.32-5STS-10.32	0.239	0.845	0.098	0.577	0.575	0.310

Table 4-8 Comparison between tested and predicted fundamental natural frequencies assuming $r_1 = r_3$ for 105 mm and 175 mm thick CLT floor panel

Floor ID	Tested $C_{f_{rrr}}$	Predicted $C_{f_{rrr}}$ ($r_3 = r_1$)	Tested f_1 (Hz)	Predicted f_1 ($r_3 = r_1$) (Hz)	Predicted f_1 ($r_3 \neq r_1$) (Hz)	Difference in Predicted f_1 (%)
C105-5.41-5STS-5.41	1.124	1.127	20.375	20.431	20.468	-0.179
C105-10.32-5STS-10.32	1.145	1.178	20.750	21.345	21.337	0.036
C105-20.15-5STS-20.15	1.162	1.180	21.063	21.392	21.426	-0.157
C105-29.97-5STS-29.97	1.179	1.229	21.375	22.279	22.293	-0.062
C105-39.80-5STS-39.80	1.183	1.216	21.438	22.047	22.093	-0.207
C105-49.62-5STS-49.62	1.193	1.261	21.625	22.854	22.875	-0.090
C175-10.32-5STS-10.32	1.108	1.120	23.000	23.230	23.245	-0.063

From Table 4-7, it can be observed that assuming r_3 equals to r_1 has a very limited impact on the value of r_2. The difference between the new r_2 values and the originally calculated r_2 is within 2%. Similarly, Table 4-8 shows that the substitution has little impact on the predicted fundamental natural frequencies, with a difference of less than 1%. Due to the negligible discrepancies, it is valid to substitute the parameter r_3 with r_1 in the Eq. 4.8 and equation of C_{f_rrr}.

The Eq. (4.8) becomes Eq. (4.19),

$$C_{d\_rrr} = \frac{(4\mu + 4)\big(4(7 - 4r_1)(1 - r_2) + \mu(4 - r_1)(16 - 9r_1 - 9r_2 + 5r_1r_2)\big)}{(16\mu + 7)(4 - r_1)(4 + 4\mu - 4r_2 - \mu r_1r_2)} \tag{4.19}$$

The simplified equation of C_{f_rrr} is also included in Appendix A. For the rest of the two-span MTP panel tests with only r_1 known, the validation results are shown in Tables 4-9 and 4-10.

Table 4-9 Fixity factor of the middle support (r_2) of two-span MTP floor panel tests calculated from r_1 and r_1=r_3.

Floor ID	d under 1 kN (mm)	$C_{d_{rrr}}$	r_1	r_2
C244-S-5STS-S	0.141	0.775	0.000	0.830
C244-S-5STS+10.32-S	0.142	0.779	0.000	0.824
C244-S-5STS+29.97-S	0.138	0.757	0.000	0.862
C244-10.32-5STS-10.32	0.122	0.672	0.096	0.957
C244-39.80-5STS-39.80	0.122	0.671	0.148	0.939
G130-S-5STS+10.32-S	0.574	0.947	0.000	0.318
G130-S-5STS+29.97-S	0.564	0.929	0.000	0.399
G130-S-5STS+44.71-S	0.535	0.882	0.000	0.572
G175-S-3STS-S	0.308	0.861	0.000	0.634
G175-S-5STS-S	0.307	0.860	0.000	0.636
G175-10.32-5STS-10.32	0.271	0.759	0.176	0.749
G175-29.97-5STS-29.97	0.264	0.740	0.285	0.698
G175-44.71-5STS-44.71	0.263	0.738	0.262	0.729
G215-S-5STS-S	0.149	0.902	0.000	0.503
G215-10.32-5STS-10.32	0.123	0.742	0.128	0.823
G215-29.97-5STS-29.97	0.116	0.703	0.198	0.859
G215-44.71-5STS-44.71	0.116	0.701	0.251	0.832

Table 4-10 Comparison between tested and predicted fundamental natural frequencies for two-span MTP floor panel tests by assuming r_1=r_3.

Floor ID	Tested $C_{f_{rrr}}$	Predicted $C_{f_{rrr}}$	Tested f_1 (Hz)	Predicted f_1 (Hz)	Difference (%)
C244-S-5STS-S	1.081	1.193	24.938	27.502	10.284
C244-S-5STS+10.32-S	1.119	1.189	25.813	27.421	6.231
C244-S-5STS+29.97-S	1.206	1.212	27.813	27.943	0.470
C244-10.32-5STS-10.32	1.295	1.310	29.875	30.204	1.102
C244-39.80-5STS-39.80	1.306	1.311	30.125	30.229	0.347
G130-S-5STS+10.32-S	1.078	1.039	27.625	26.612	-3.665
G130-S-5STS+29.97-S	1.088	1.052	27.875	26.966	-3.262
G130-S-5STS+44.71-S	1.066	1.091	27.313	27.952	2.340
G175-S-3STS-S	1.150	1.109	23.000	28.698	24.772
G175-S-5STS-S	1.163	1.110	23.250	28.712	23.494
G175-10.32-5STS-10.32	1.363	1.204	27.250	31.152	14.320
G175-29.97-5STS-29.97	1.425	1.221	28.500	31.583	10.817
G175-44.71-5STS-44.71	1.438	1.224	28.750	31.682	10.199
G215-S-5STS-S	1.098	1.074	44.750	43.754	-2.225
G215-10.32-5STS-10.32	1.147	1.225	46.750	49.910	6.760
G215-29.97-5STS-29.97	1.270	1.269	51.750	51.701	-0.095
G215-44.71-5STS-44.71	1.178	1.270	48.000	51.761	7.836

Combining the validation results in Tables 4-5 and 4-10, it is obvious that most differences are within 11%, except three outliers which are less than 25%.

4.5 Summary and Conclusions

In this chapter, two analytical expressions for coefficients C_{d_rrr} and C_{f_rrr} have been developed. These two coefficients can help find the mid-span deflection under point load and fundamental natural frequency of a two-span MTP floor panel with restrained boundary conditions by multiplying the coefficients to those of the two-span MTP floor panels with the same configuration except having simply supported boundary conditions.

Two sensitivity analyses were performed of the proposed analytical expressions. The results of sensitivity analyses indicated that one parameter, r_3, has insignificant effects on both equations. The expressions are then simplified by substituting r_3 with r_1.

Analytical expressions were verified against the two-span floor panel tests discussed in Chapter 3. In total 38 MTP floor panel tests were included in the verification process, with varying MTP type, panel thickness and boundary conditions. The verification adopted the circular argument fallacy. It started by finding the value of r_2 by using analytical expression of C_{d_rrr}. Then, the determined C_{f_rrr} based on r_2 was used to find the predicted f_1, which is then compared to the f_1 obtained by the tests directly. The results showed that the values of predicted f_1 and tested f_1 are close. The differences are mostly within 11%, and three outliers exceed 11% but are still within 25%.

Chapter 5 - Analytical Models for Calculating Mid-span deflection and Fundamental Natural Frequency of Beam-supported Floor with Restrained Connections

5.1 Introduction

Besides floors supported by walls discussed in the previous chapter, it is common to use beams as floor supports in practice. In this Chapter, mid-span deflection and natural frequency formulae that consider the effects of supporting beam stiffness have been developed. The floor panel is considered as one-way Euler-Bernoulli beam. The formulae included both one-span and two-span floor structures. The experimental results presented in Chapter 3 are compared with the predicted floor properties. Conclusions are drawn from the comparison and discussion.

5.2 Mid-span Deflection Equation of Beam-supported Floor with Restrained Connections

5.2.1 Beam-supported Single-span Floor

A one-span floor supported by beams at two ends and having restrained boundary conditions has been simplified as a model shown in Figure 5-1. The floor is seen as a one-way Euler-Bernoulli beam. The restrained boundary conditions and supporting beams are represented by rotational springs and translational spring supports. The floor has a span of L. The fixity factors for the two supports are r_1 and r_2. Their values can range between 0 for simple support and 1 for fully restrained support. The translational springs, K_1 and K_2, are determined from the flexural stiffnesses of the supporting beams.

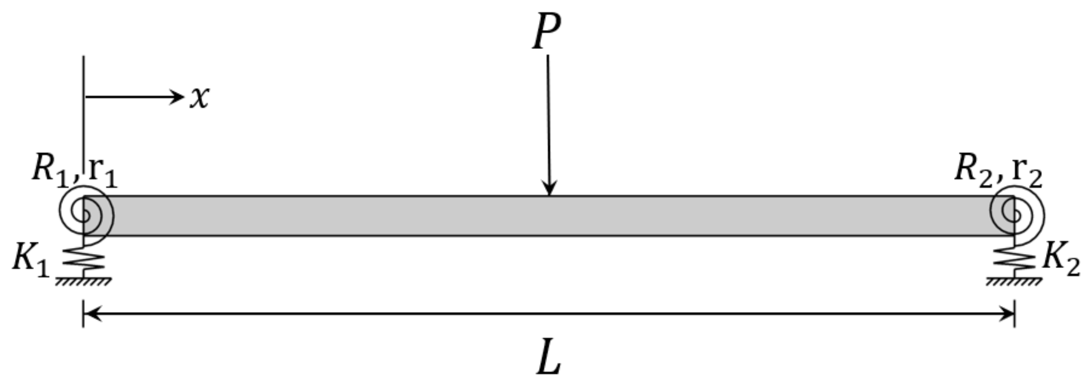


Figure 5-1 A one-span, rotationally restrained beam supported by springs with a point load at mid-span

The deflection equation was obtained using the matrix method. As Figure 5-2 shows, the structure can be divided into two elements with length of L/2 and has 6 degrees of freedom (DOFs) in total.

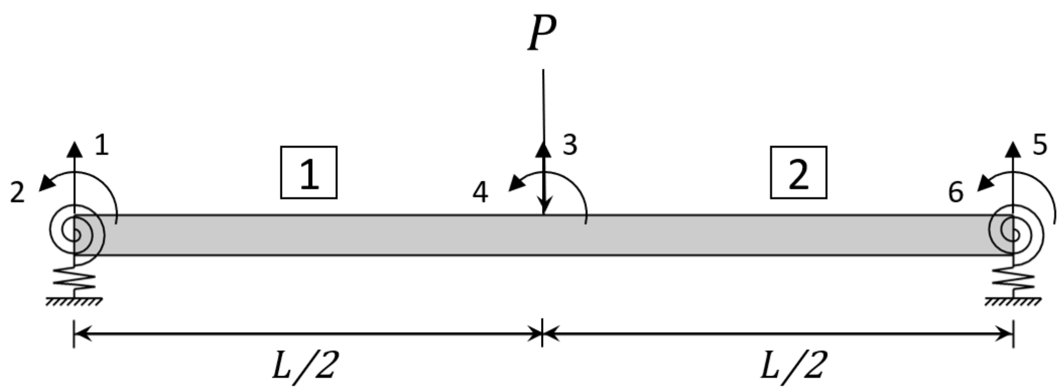


Figure 5-2 DOFs and elements of a one-span and rotationally restrained beam supported by springs with a point load applied

The stiffness matrix of the beam is:

$$K = \begin{bmatrix} \frac{96EI}{L^3} + K_1 & \frac{24EI}{L^2} & -\frac{96EI}{L^3} & \frac{24EI}{L^2} & 0 & 0 \\ \frac{24EI}{L^2} & \frac{8EI}{L} + R_1 & -\frac{24EI}{L^2} & \frac{4EI}{L} & 0 & 0 \\ \frac{24EI}{L^2} & -\frac{24EI}{L^2} & \frac{192EI}{L^3} & 0 & -\frac{96EI}{L^3} & \frac{24EI}{L^2} \\ \frac{24EI}{L^2} & \frac{4EI}{L} & 0 & \frac{16EI}{L} & -\frac{24EI}{L^2} & \frac{4EI}{L} \\ 0 & 0 & -\frac{96EI}{L^3} & -\frac{24EI}{L^2} & \frac{96EI}{L^3} + K_2 & -\frac{24EI}{L^2} \\ 0 & 0 & \frac{24EI}{L^2} & \frac{4EI}{L} & -\frac{24EI}{L^2} & \frac{8EI}{L} + R_2 \end{bmatrix} \tag{5.1}$$

where EI is the flexural stiffness of the floor panel.

The force vector is:

$$F = \begin{bmatrix} 0 \\ 0 \\ -P \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.2}$$

The displacement vector can be calculated using:

$$\Delta = K^{-1} \times F \tag{5.3}$$

The deflection of interest in this study is the mid-span deflection which is DOF 3 in Figure 5-2. It can be found as the third element in the displacement vector, obtained by solving Eq. (5.3). For simplification purposes, the rotational stiffnesses in the equation have been substituted by fixity factors according to Eq. (4.4). It has the form of:

$$d_{B\_1} = C_{dB\_1} \times d_{ss}$$

$$d_{B\_1} = C_{dB\_1} \times \frac{PL^3}{48EI} \tag{5.4}$$

The equation is a coefficient, C_{dB_1}, which considers the effect of supporting beams and restrained connections, into a deflection equation of simply supported floor under a point load in the middle (d_{ss}). The full deflection equation for the generalized case is present in Appendix B due to the excessive length. When the beam has the same rotational restraints and same supporting spring constants at both ends, the mid-span deflection can be simplified to:

$$d_{B\_1} = \left(\frac{24EI}{KL^3} + \frac{8 - 5r}{8 + 4r}\right)\frac{-PL^3}{48EI} \tag{5.5}$$

5.2.2 Beam-supported Two-span Floor

Figure 5-3 shows a typical rotationally restrained two-span floor supported on beams. The left span, L, is the set as the longer span. The span ratio, μ, ranges between 0 to 1. The floor is also considered as a one-way Euler-Bernoulli beam. The stiffness of the restrained boundary conditions and supporting beams are R and k, which are numbered according to the supports. The fixity factor of each support is defined as Eq. (4.4), same as two-span and wall supported floor in Chapter 4.

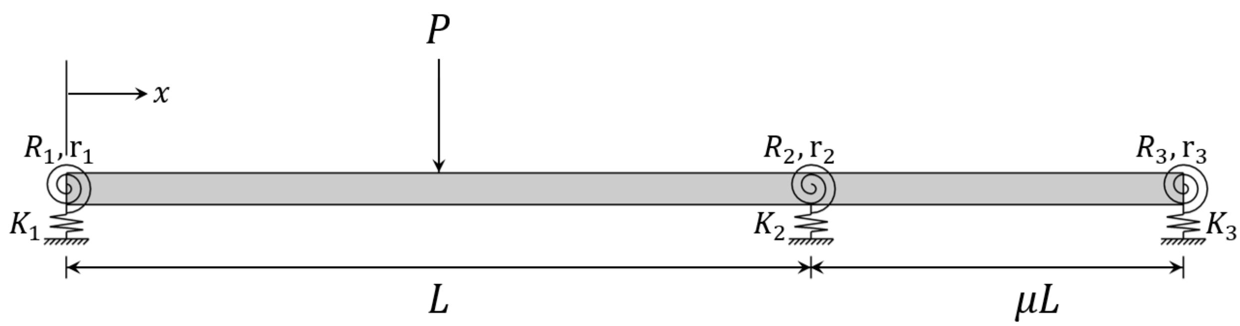


Figure 5-3 A two-span, rotationally restrained beam supported by springs with a point load at mid-span

For the purpose of matrix analysis, the two-span floor was divided into three elements, as shown in Figure 5-4. A total of eight DOFs of the floor have been labelled. The deflection to be determined is along the third DOF.

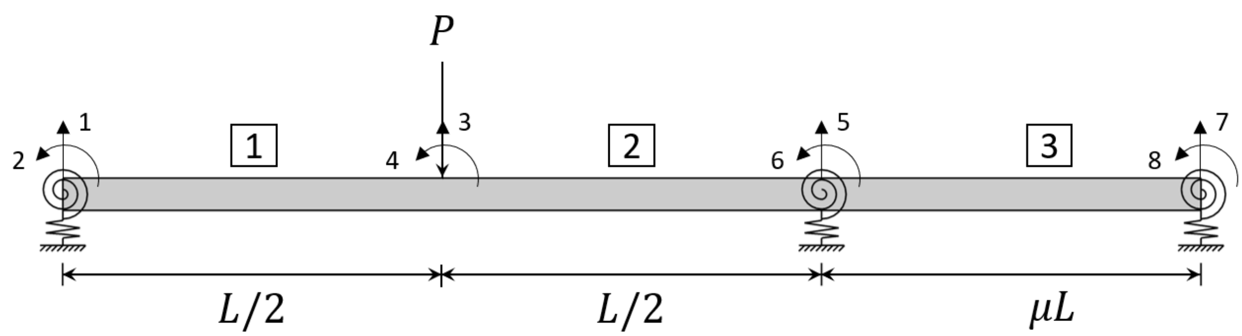


Figure 5-4 DOFs and elements of a two-span and rotationally restrained beam supported by springs with a point load applied

The stiffness matrix of the two-span floor system shown in Figure 5-4 is:

$K =$

$$\begin{bmatrix} \frac{96EI}{L^3}+K_1 & \frac{24EI}{L^2} & -\frac{96EI}{L^3} & \frac{24EI}{L^2} & 0 & 0 & 0 & 0 \\ \frac{24EI}{L^2} & \frac{8EI}{L}+R_1 & -\frac{24EI}{L^2} & \frac{4EI}{L} & 0 & 0 & 0 & 0 \\ -\frac{96EI}{L^3} & -\frac{24EI}{L^2} & \frac{192EI}{L^3} & 0 & -\frac{96EI}{L^3} & \frac{24EI}{L^2} & 0 & 0 \\ \frac{24EI}{L^2} & \frac{4EI}{L} & 0 & \frac{16EI}{L} & -\frac{24EI}{L^2} & \frac{4EI}{L} & 0 & 0 \\ 0 & 0 & -\frac{96EI}{L^3} & -\frac{24EI}{L^2} & \frac{96EI}{L^3}+\frac{12EI}{(\mu L)^3}+K_2 & -\frac{24EI}{L^2}+\frac{6EI}{(\mu L)^2} & -\frac{12EI}{(\mu L)^3} & \frac{6EI}{(\mu L)^2} \\ 0 & 0 & \frac{24EI}{L^2} & \frac{4EI}{L} & -\frac{24EI}{L^2}+\frac{6EI}{(\mu L)^2} & \frac{8EI}{L}+\frac{4EI}{(\mu L)}+R_2 & -\frac{6EI}{(\mu L)^2} & \frac{2EI}{(\mu L)} \\ 0 & 0 & 0 & 0 & -\frac{12EI}{(\mu L)^3} & -\frac{6EI}{(\mu L)^2} & \frac{12EI}{(\mu L)^3}+K_3 & -\frac{6EI}{(\mu L)^2} \\ 0 & 0 & 0 & 0 & \frac{6EI}{(\mu L)^2} & \frac{2EI}{(\mu L)} & -\frac{6EI}{(\mu L)^2} & \frac{4EI}{(\mu L)}+R_3 \end{bmatrix} \tag{5.6}$$

The force vector is:

$$F = \begin{bmatrix} 0 \\ 0 \\ -P \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.7}$$

The mid-span deflection under the point load can be found from the third element in the deflection vector, calculated from the stiffness matrix and force vector using Eq. (5.3). The equation is the product of a coefficient, C_{dB_2}, including effects of two span, supporting springs, and restrained connections and deflection equation of a simply supported beam with same configurations and span L, Eq. (5.8). The entire equation of C_{dB_2} is shown in Appendix B.

$$d_{B\_2} = C_{dB\_2} \times d_{ss}$$

$$d_{B\_2} = C_{dB\_2} \times \frac{PL^3}{48EI} \tag{5.8}$$

5.3 Natural Frequency Equation of Beam-supported Floor with Restrained Connections

5.3.1 Beam-supported Single-span Floor

The natural frequency equation of the one-span and beam-supported beam was also based on the Rayleigh-Morleigh formula presented in Eq. (4.10). The formula was derived from Rayleigh's quotient according to the static displacement function of the beam with uniformly distributed load applied as shown in Figure 5-5.

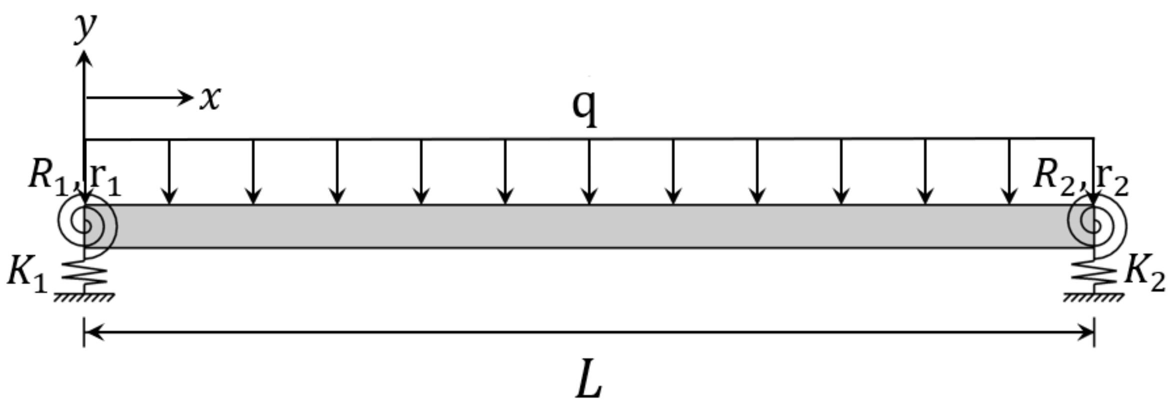


Figure 5-5 A one-span beam with rotationally restrained and translational supports under a uniformly distributed load

The natural frequency analytical equation development was started with deriving the static displacement functions from the governing equation of Euler-Bernoulli beams as Eq. (5.9) and the four constants, C_1 to C_4, require four boundary conditions as shown in Table 5-1.

$$y(x) = \frac{-q}{24EI}x^4 + \frac{x^3}{6}C_1 + \frac{x^2}{2}C_2 + xC_3 + C_4 \tag{5.9}$$

The first two boundary conditions, displacements at the two supports, are determined by stiffness matrix method. As Figure 5-6 shows, the one-span beam with a uniformly distributed load has four DOFs and only one member.

Table 5-1 Boundary conditions of beam shown in Figure 5-5

Explanations	Boundary Conditions
Displacement @ support 1	$y(x=0)$
Displacement @ support 2	$y(x=L)$
Moment @ support 1	$EI\frac{d^2y(x=0)}{dx^2} = R_1\frac{dy(x=0)}{dx}$
Moment @ support 2	$EI\frac{d^2y(x=L)}{dx^2} = R_2\frac{dy(x=L)}{dx}$

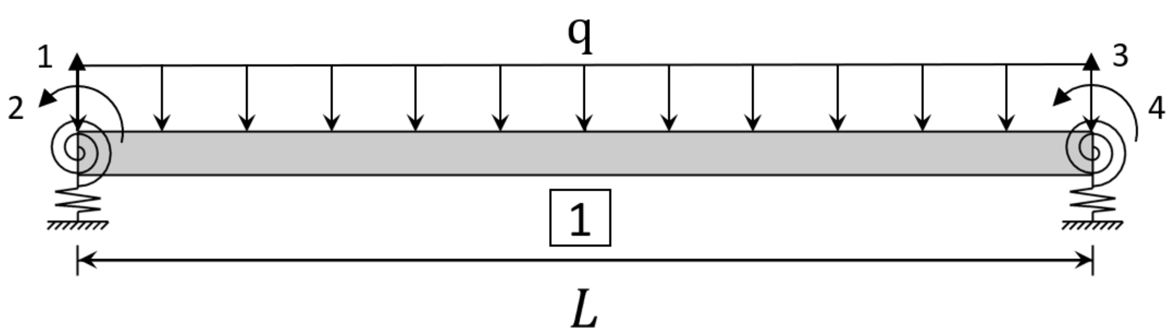


Figure 5-6 DOFs and elements of a one-span and rotationally restrained beam supported by springs with a uniformly distributed load

The stiffness matrix of the beam is:

$$K = \begin{bmatrix} \frac{12EI}{L^3} + K_1 & \frac{6EI}{L^2} & -\frac{12EI}{L^3} & \frac{6EI}{L^2} \\ \frac{6EI}{L^2} & \frac{4EI}{L} + R_1 & -\frac{6EI}{L^2} & \frac{2EI}{L} \\ -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & \frac{12EI}{L^3} + K_2 & -\frac{6EI}{L^2} \\ \frac{6EI}{L^2} & \frac{2EI}{L} & -\frac{6EI}{L^2} & \frac{4EI}{L} + R_2 \end{bmatrix} \qquad (5.10)$$

The uniformly distributed load, q, has to be transformed to equivalent nodal forces in the stiffness matrix method as Figure 5.7 illustrates.

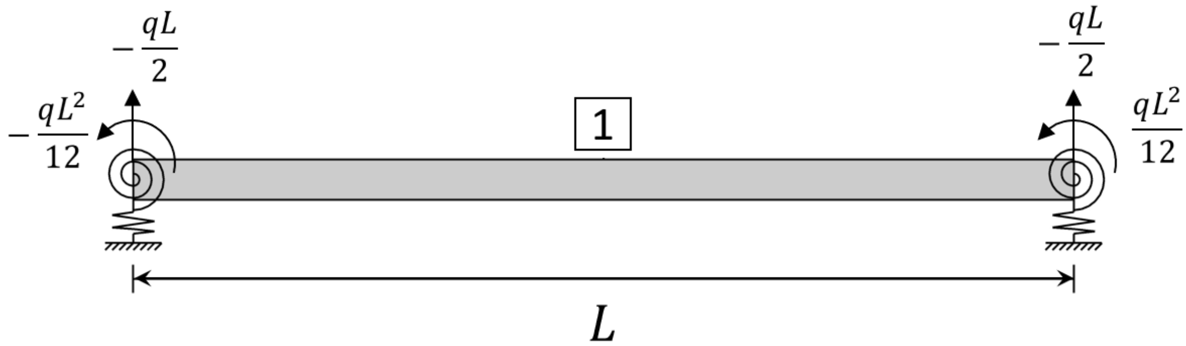


Figure 5-7 Equivalent nodal forces of the uniformly distributed load q

The force vector becomes:

$$F = -\begin{bmatrix} \frac{qL}{2} \\ \frac{qL^2}{12} \\ \frac{qL}{2} \\ -\frac{qL^2}{12} \end{bmatrix} \tag{5.11}$$

The displacement vector can be found using Eq. 5.3 and the displacements at two supports are the first and third elements. The static displacement function, Eq. (5.9), can now be solved with the four boundary conditions. Therefore, the fundamental natural frequency can be determined based on the Eq. (4.10).

The final frequency equation containing a coefficient, C_{fB_1} to consider the effects of support springs and restrained connections, and frequency equation of a simply supported beam with the same configuration, is shown in Eq. (5.12). The full equation for C_{fB_1} is presented in Appendix B.

$$f_{B\_1} = C_{fB\_1} \times f_{ss}$$
$$= C_{fB\_1} \times \frac{\pi}{2L^2}\sqrt{\frac{EI}{\rho A}} \tag{5.12}$$

5.3.2 Beam-supported Two-span Floor

Similar to two-span wall-supported floor in Section 4.3, the Reyleigh-Morleigh formulae, Eq. (4.10), was adopted to derive the frequency equation for a two-span floor with rotationally restrained and spring supports. Rayleigh's quotient is based on the beam's static displacement function under uniformly distributed forces applying in the opposite directions to each span, as shown in Figure 5-8.

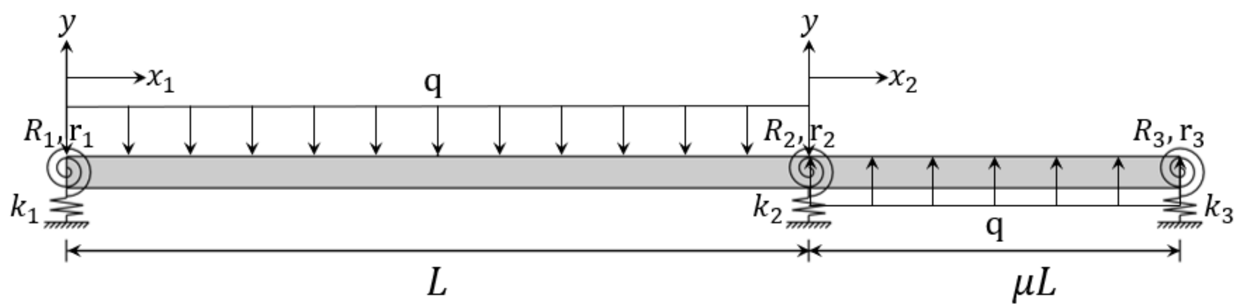


Figure 5-8 A two-span, rotationally restrained floor sitting on beams under uniformly distributed load

The static displacement functions were derived from governing equation of Euler-Bernoulli beam, same as Eq. (4.11). The static transverse displacement functions, $y(x_1)$ and $y(x_2)$, are for the first and second span respectively. The two equations can be solved by substituting the eight boundary conditions shown in Table 5-2.

The first three boundary conditions, the displacements at the three supports, are unknown. Same as last section, the stiffness matrix method was introduced to solve the displacements. The two-span beam under uniformly distributed load has two elements and six DOFs as Figure 5-9 shown.

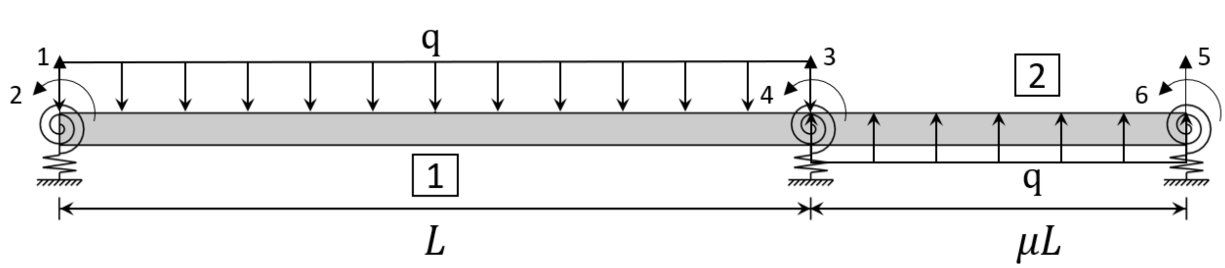


Figure 5-9 DOFs and elements of a two-span and rotationally restrained beam supported by springs with uniformly distributed loads

Table 5-2 Boundary Conditions of Two-span Rotationally Restrained Beam Sitting on Flexible Supports as shown in Figure 5-8

Explanations	Boundary Conditions
Displacement @ support 1	$y(x_1 = 0)$
Displacement @ support 2	$y(x_1 = L)$
Displacement @ support 2	$y(x_2 = L)$
Displacement @ support 3	$y(x_2 = L + \mu L)$
Moment @ support 1	$EI \dfrac{d^2 y(x_1 = 0)}{dx_1^2} = R_1 \dfrac{dy(x_1 = 0)}{dx_1}$
Moment @ support 2	$EI \dfrac{d^2 y(x_1 = L)}{dx_1^2} - EI \dfrac{d^2 y(x_2 = L)}{dx_2^2} + R_2 \dfrac{dy(x_1 = L)}{dx_1} = 0$
Moment @ support 3	$-EI \dfrac{d^2 y(x_2 = L + \mu L)}{dx_2^2} = R_3 \dfrac{dy(x_2 = L + \mu L)}{dx_2}$
Slope @ support 2	$\dfrac{dy(x_1 = L)}{dx_1} = \dfrac{dy(x_2 = L)}{dx_2}$

The stiffness matrix of the whole structure is:

$$K = \begin{bmatrix} \frac{12EI}{L^3} + K_1 & \frac{6EI}{L^2} & -\frac{12EI}{L^3} & \frac{6EI}{L^2} & 0 & 0 \\ \frac{6EI}{L^2} & \frac{4EI}{L} + R_1 & -\frac{6EI}{L^2} & \frac{2EI}{L} & 0 & 0 \\ -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & \frac{12EI}{L^3} + \frac{12EI}{(\mu L)^3} + K_2 & -\frac{6EI}{L^2} + \frac{6EI}{(\mu L)^2} & -\frac{12EI}{(\mu L)^3} & \frac{6EI}{(\mu L)^2} \\ \frac{6EI}{L^2} & \frac{2EI}{L} & -\frac{6EI}{L^2} + \frac{6EI}{(\mu L)^2} & \frac{4EI}{L} + \frac{4EI}{\mu L} + R_2 & -\frac{6EI}{(\mu L)^2} & \frac{2EI}{\mu L} \\ 0 & 0 & -\frac{12EI}{(\mu L)^3} & -\frac{6EI}{(\mu L)^2} & \frac{12EI}{(\mu L)^3} + K_3 & -\frac{6EI}{(\mu L)^2} \\ 0 & 0 & \frac{6EI}{(\mu L)^2} & \frac{2EI}{\mu L} & -\frac{6EI}{(\mu L)^2} & \frac{4EI}{\mu L} + R_3 \end{bmatrix} \quad (5.13)$$

The uniformly distributed loads applying to both spans have been transformed into equivalent nodal force as Figure 5.10 illustrates.

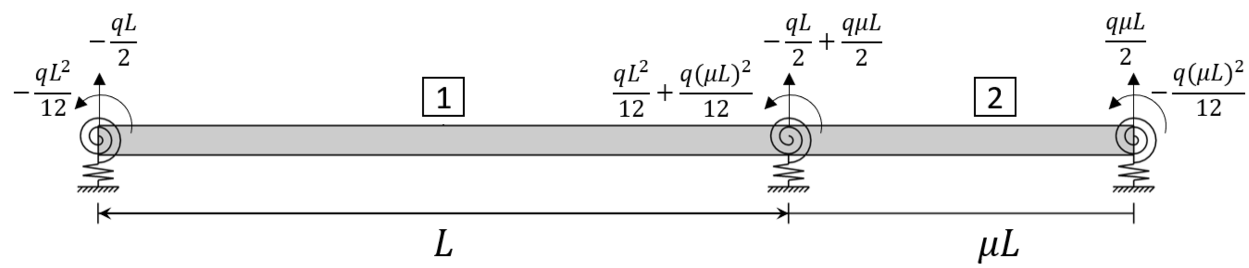


Figure 5-10 Equivalent nodal force of uniformly distributed load q and -q

As a result of this transformation, the force vector is obtained as shown below:

$$F = -\begin{bmatrix} \frac{qL}{2} \\ \frac{qL^2}{12} \\ \frac{qL}{2} - \frac{q\mu L}{2} \\ -\frac{qL^2}{12} - \frac{q(\mu L)^2}{12} \\ -\frac{q\mu L}{2} \\ \frac{q(\mu L)^2}{12} \end{bmatrix} \tag{5.14}$$

The displacements at the three supports can be found as the first, third and fifth element in displacement vector, which can be determined by substituting Eq. (5.13) and (5.14) into Eq. (5.3). After substituting all eight boundary conditions into Eq. (4.11), the displacement function for a two-span, rotationally restrained, and beam supported floor can be solved. The fundamental natural frequency equation can be determined based on the Eq. (4.10).

The final frequency equation contains a coefficient, C_{fB_2}, to include effects of two span, supporting springs, and restrained connections, and a frequency equation of a simply supported one-span floor with the same configuration and span L, as shown in Eq. (5.15):

$$\begin{aligned} f_{B\_2} &= C_{fB\_2} \times f_{ss} \\ &= C_{fB\_2} \times \frac{\pi}{2L^2}\sqrt{\frac{EI}{\rho A}} \end{aligned} \tag{5.15}$$

Due to the extensive length of the equation, the full equation of C_{fB_2} is presented in Appendix B.

The proposed analytical equation has a limitation of predicting a floor with two span equal and two identical connections and supports at the ends, as Figure 5-11 shown. The Rayleigh-Morleigh formula is only valid when the deflected shape largely agrees with the mode shape (Petersen and Werkle 2018). When the middle support is stiff, the first mode shape of the beam is close to the deflected shape as shown in Figure 5-12 (a). When the stiffness of middle support becomes very low, the first mode shape would deviate from a sine wave and more likely resemble the shape shown in Figure 5-12 (b) (Catterou et al. 2020). Therefore, as the middle support stiffness decreases, the deflected shape and first mode shape are no longer alike and the Rayleigh-Morleigh formula cannot provide a good approximation of the mode shape function.

As Figure 5-11 shows, when the two connections and supports at ends are identical and the uniformly distributed loads applying to the two spans are opposite, the deflection of the intermediate support is always zero regardless the change of middle support stiffness.

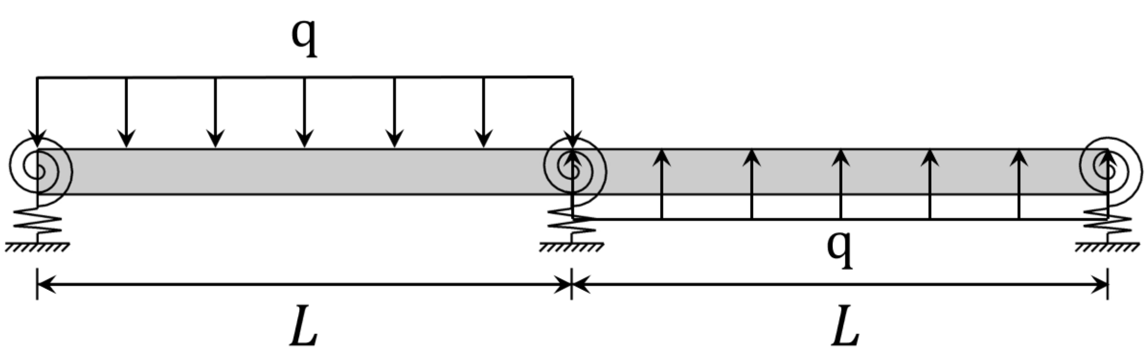


Figure 5-11 A two-equal-span beam under uniformly distributed load

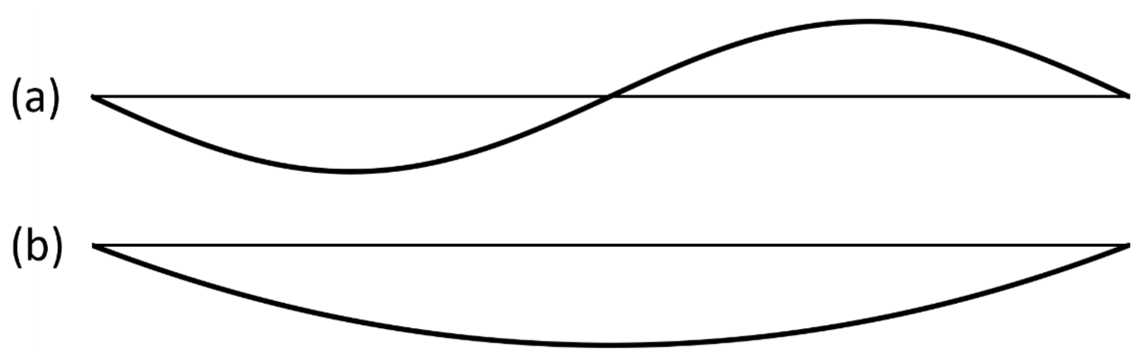


Figure 5-12 The first mode shapes of a beam shown in Figure 5-11 with (a) stiff middle support (b) very flexible middle support

5.4 Validation

In previous Section 5.2 and 5.3, the analytical equations of mid-span deflection under point load and fundamental natural frequency of one-span and two-span rotationally restrained floor sitting on beams were introduced. The sensitivity analysis has been conducted to simplify the analytical equations. These four equations will be verified against the laboratory results from beam-supported MTP floor test presented in Chapter 3.

5.4.1 Sensitivity Analysis

5.4.1.1 Supporting Beam Stiffness Factor

The supporting beam stiffness factor, k, is introduced as a measurement of the flexural stiffness of supporting beams in terms of the flexural stiffness of floor panels, which is expressed as:

$$k = \frac{K_{Floor}}{K_{Beam}} = \frac{\frac{48EI_{Floor}}{L_{Floor}{}^3}}{K_{Beam}} \tag{5.16}$$

where K_{Floor} is the flexural stiffness of floor panel. K_{Floor} is calculated from the floor panel flexural stiffness, which was measured using the concentrated load applying to mid-span. L_{Floor} and EI_{Floor} are the span and flexural stiffness of floor panel. K_{Beam} is the spring constant and flexural stiffness of supporting beam. The flexural stiffness of beam is determined by its configuration. For example, in practice the supporting beam normally experience distributed load throughout the beam span from the floor. In this case, the flexural stiffness of the beam is calculated using the beam under uniformly distributed load. Since in the beam-support test shown in Chapter 3, the floor is sitting in the middle section of the beams. The beam flexural stiffness equation for this particular configuration will be introduced in Section 5.4.2.

Since in practice the flexural stiffness of floor beams are usually stiffer than floor panels that are supported by these beams, the value of supporting beam factor is less than one. Substituting supporting beam factor for spring constant in the deflection and frequency equations of beam supported floor will make the coefficients, C_{dB_1}, C_{dB_2} , C_{fB_1} and C_{fB_2}, dimensionless.

For example, the deflection equation of a single-span, beam-supported floor, Eq. (5.5), becomes:

$$d_{B\_1} = C_{dB\_1} \frac{-PL^3}{48EI}$$

$$C_{dB\_1} = \frac{8 + 4k - 5r + 2kr}{8 + 4r} \tag{5.17}$$

For coefficient, C_{fB_1}, in frequency equation of single-span, beam-supported floor, when fixity factor and supporting beam factor are the same, it becomes:

$$C_{fB\_1} = 3.94 \sqrt{\frac{(8 + 4r)(8 - 6r + 10k + 5kr)}{8(124 - 182r + 67r^2) - 252k(-8 + 2r + 3r^2) + 315k^2(2 + r)^2}} \tag{5.18}$$

The coefficient equations of C_{dB_2} and C_{fB_2} will be further simplified in next section.

5.4.1.2 Sensitivity Analysis and Simplified Equation

Among all proposed coefficients in analytical equations, C_{dB_2} and C_{fB_2} are long and contain seven parameters. Sensitivity analyses have been conducted to eliminating parameters and simplify the coefficient expressions. The numerator of C_{fB_2} equation cannot be solved until the values of all seven parameters are substituted due to the presence of absolute value as shown in Eq. (4.10). Therefore, the absolute value calculation was simplified as shown in Eq. (5-19) to ensure that the sensitivity analysis model can be executed properly.

$$\int_0^L |y(x)|\, dx \approx \int_0^L -y(x_1)\, dx + \int_0^{\mu L/2} -y(x_2)\, dx + \int_0^{\mu L/2} y(x_2)\, dx \tag{5.19}$$

Similar to Section 4.5.2, the sensitivity analyses were conducted on SALib using Sobol method. The detailed Python code is included in Appendix B. The results from the sensitivity analyses of C_{dB_2} and C_{fB_2} when all the seven parameters range between 0 and 1 are shown in Table 5-3. It is clear that both first-order and total-order indices of parameter r_3 and k_3 are far smaller than the other parameters. The second-order indices between r_3 and the other parameters are also small. The test results assume that the supporting beam stiffness factor is less than 1, which is true only when the beam stiffness is greater than the floor stiffness.

For the beam-supported floor tests conducted in Chapter 3, the beam stiffnesses are relatively small compared to floor panel stiffness. The supporting beam stiffness factors can be as high as 12. The sensitivity analyses were performed with the value of beam stiffness factor ranging from 0 to 12 and the results are shown in Table 5-4. The results shown in Table 5-4 are similar to Table 5-3 as the first and total indices of parameter r_3 and k_3 are also the smallest among all the parameters and the second-order indices between r_3 and the other parameters are small. Therefore, it is valid to substitute r_3 with r_1 into C_{dB_2} and C_{fB_2} to simplify the equation and provide conservative results. The simplified equation of C_{dB_2} is included in Appendix B. Since C_{fB_2} requires calculation in multiple steps and the substitution with values of all parameters, the substitution of r_3 with r_1 does not lead to simplification of the equation.

Table 5-3 First, second, and total-order indices of C_{dB_2} and C_{fB_2} equations

C_{dB_2}	First-order indices	Second-order indices	Total-order indices	C_{fB_2}	First-order indices	Second-order indices	Total-order indices
μ	0.0972	-	0.1444	μ	0.2164	-	0.3357
r_1	0.2873	-	0.3143	r_1	0.0909	-	0.1385
r_2	0.1576	-	0.1845	r_2	0.2024	-	0.2358
r_3	0.0079	-	0.0175	r_3	0.0058	-	0.0182
k_1	0.1505	-	0.1653	k_1	0.2742	-	0.3389
k_2	0.2175	-	0.2533	k_2	0.0223	-	0.0420
k_3	-0.0009	-	0.0058	k_3	0.0170	-	0.0326
μ, r_1	-	0.0034	-	μ, r_1	-	0.0286	-
μ, r_2	-	0.0092	-	μ, r_2	-	0.0295	-
μ, r_3	-	0.0027	-	μ, r_3	-	0.0185	-
μ, k_1	-	0.0075	-	μ, k_1	-	0.0439	-
μ, k_2	-	0.0212	-	μ, k_2	-	0.0272	-
μ, k_3	-	0.0035	-	μ, k_3	-	0.0238	-
r_1, r_2	-	0.0004	-	r_1, r_2	-	-0.0002	-
r_1, r_3	-	-0.0032	-	r_1, r_3	-	-0.0036	-
r_1, k_1	-	0.0017	-	r_1, k_1	-	0.0120	-
r_1, k_2	-	0.0071	-	r_1, k_2	-	-0.0045	-
r_1, k_3	-	-0.0041	-	r_1, k_3	-	-0.0060	-
r_2, r_3	-	-0.0020	-	r_2, r_3	-	0.0000	-
r_2, k_1	-	0.0023	-	r_2, k_1	-	0.0026	-
r_2, k_2	-	0.0082	-	r_2, k_2	-	-0.0025	-
r_2, k_3	-	-0.0008	-	r_2, k_3	-	-0.0001	-
r_3, k_1	-	-0.0022	-	r_3, k_1	-	0.0025	-
r_3, k_2	-	-0.0069	-	r_3, k_2	-	-0.0024	-
r_3, k_3	-	-0.0049	-	r_3, k_3	-	0.0000	-
k_1, k_2	-	0.0006	-	k_1, k_2	-	0.0075	-
k_1, k_3	-	-0.0004	-	k_1, k_3	-	0.0018	-
k_2, k_3	-	-0.0017	-	k_2, k_3	-	-0.0030	-

Table 5-4 First, second, and total-order indices of C_{dB_2} and C_{fB_2} equations when beam stiffness factor ranges from 0 to 12

C_{dB_2}	First-order indices	Second-order indices	Total-order indices	C_{fB_2}	First-order indices	Second-order indices	Total-order indices
μ	0.0733	-	0.1172	μ	0.0039	-	0.3024
r_1	0.0032	-	0.0149	r_1	-0.0024	-	0.0188
r_2	0.0440	-	0.0683	r_2	0.0658	-	0.1005
r_3	0.0069	-	0.0160	r_3	-0.0013	-	0.0056
k_1	0.4562	-	0.5139	k_1	0.4489	-	0.6814
k_2	0.2594	-	0.3148	k_2	0.0377	-	0.0696
k_3	0.0389	-	0.0751	k_3	0.0512	-	0.1351
μ, r_1	-	-0.0007	-	μ, r_1	-	0.0338	-
μ, r_2	-	0.0054	-	μ, r_2	-	0.0541	-
μ, r_3	-	0.0020	-	μ, r_3	-	0.0335	-
μ, k_1	-	0.0081	-	μ, k_1	-	0.1392	-
μ, k_2	-	0.0090	-	μ, k_2	-	0.0411	-
μ, k_3	-	0.0114	-	μ, k_3	-	0.0660	-
r_1, r_2	-	0.0010	-	r_1, r_2	-	0.0067	-
r_1, r_3	-	0.0012	-	r_1, r_3	-	-0.0007	-
r_1, k_1	-	0.0087	-	r_1, k_1	-	-0.0041	-
r_1, k_2	-	0.0030	-	r_1, k_2	-	-0.0001	-
r_1, k_3	-	0.0015	-	r_1, k_3	-	0.0035	-
r_2, r_3	-	-0.0001	-	r_2, r_3	-	0.0100	-
r_2, k_1	-	0.0126	-	r_2, k_1	-	0.0181	-
r_2, k_2	-	-0.0002	-	r_2, k_2	-	0.0059	-
r_2, k_3	-	-0.0010	-	r_2, k_3	-	-0.0053	-
r_3, k_1	-	0.0006	-	r_3, k_1	-	0.0033	-
r_3, k_2	-	0.0018	-	r_3, k_2	-	0.0020	-
r_3, k_3	-	0.0030	-	r_3, k_3	-	0.0053	-
k_1, k_2	-	0.0253	-	k_1, k_2	-	0.0294	-
k_1, k_3	-	0.0012	-	k_1, k_3	-	0.0424	-
k_2, k_3	-	0.0166	-	k_2, k_3	-	-0.0066	-

5.4.2 Verification of Beam-supported Floor Equations against Test Results

The verification started with identifying the spring constants, K, of supporting beams according to static deflection results from beam tests in Chapter 3. The loading configurations of supporting beams in beam tests and in floor tests are shown in Figure 5-13.

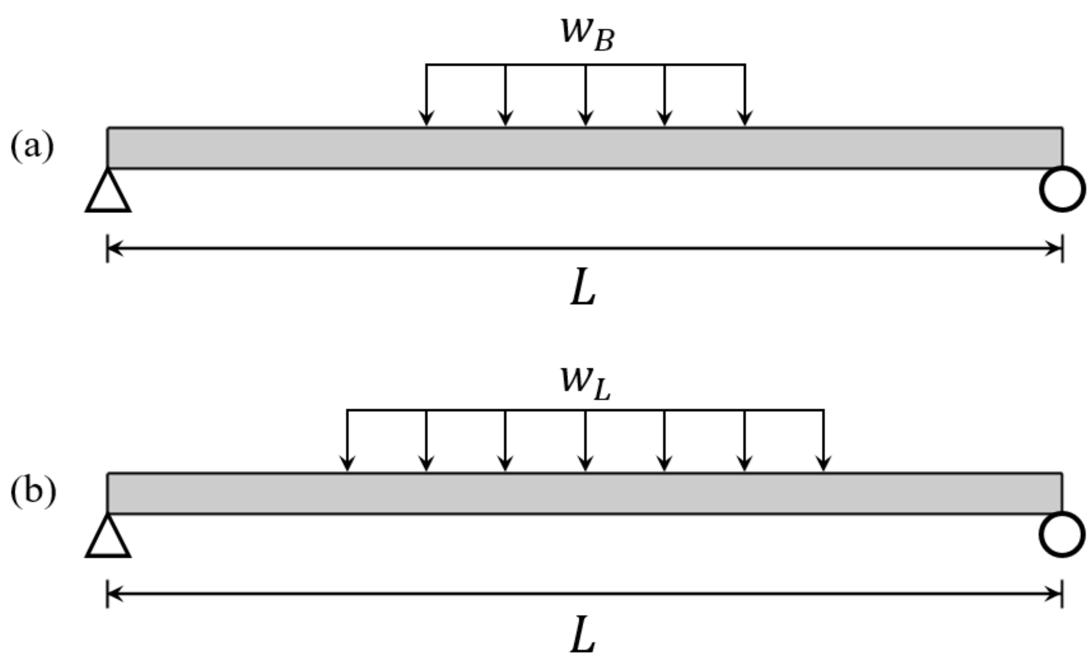


Figure 5-13 Loading configurations of supporting beams in (a) beam tests and in (b) floor tests

The beam's flexural stiffness with uniformly distributed load applying to the middle segment of the beam can be determined using the Equation 3.1. The beam's bending stiffness in the floor tests from the deflection results from the beam tests can be expressed as:

$$K = K_B \frac{(8L^3 - 4Lw_B{}^2 + w_B{}^3)}{(8L^3 - 4Lw_F{}^2 + w_F{}^3)} \tag{5.19}$$

Where L is the span of beam. In the deflection test of supporting beams, the load was applying by placing 22.7 kg (50-lb) weights in the middle. The loading width w_B is equal to the weight's length, which is 250 mm. Meanwhile, the loading width w_F in the floor test is equal to floor panel width, 600 mm. The bending stiffness of supporting beams are listed in the Table 5-5.

Table 5-5 Stiffnesses of Supporting Beams

Specimen I.D.	d under 1 kN (mm)	K_{Beam} (kN/mm)
2LB-2	2.48	0.42
2LB-3	2.70	0.39
2LB-4	2.46	0.42
3LB-1	1.12	0.94
3LB-2	1.19	0.88
3LB-3	1.18	0.89

5.4.2.1 Beam-supported Single-span Floor

In the beam-supported single-span floor tests, all the test specimens have the same type of connections and supporting beams at both ends. Therefore, the value of fixity factors and supporting beam stiffness factors are the same. To determine the value of coefficient C_{dB_1} and C_{fB_1}, the values of fixity factors, supporting beam stiffness factors, deflections, and natural frequency need to be determined from the reference one-span tests in Chapter 3. In Table 5-6, the fixity factors and supporting beam stiffness factors of floor panel are listed, in which the supporting beam stiffness factors were determined from the simply supported floor tests.

Table 5-6 Fixity Factors and Flexural Stiffness of the One-span Reference Beam Tests

Specimen I.D.	Span Length (mm)	d under 1 kN (mm)	K_{Floor} (kN/mm)	f_1 (Hz)	r
G175-4877-S	4877	0.79	1.26	17.13	0.00
G175-4877-3STS	4877	0.75	1.26	18.50	0.05
G175-6096-S	6096	1.51	0.66	11.50	0.00
G175-6096-3STS	6096	1.41	0.66	12.38	0.06

The supporting beam factors of each test are determined by Eq. (5.16) using beam and floor stiffnesses, K_{Beam} and K_{Floor}, from Table 5-5 and 5-6. The results are listed in Table 5-7. The tested and predicted deflections and fundamental natural frequencies of one-span and beam-supported floor tests are shown in Table 5-8 and Table 5-9.

Table 5-7 Average Supporting Beam Factors of One-span and Beam-supported Floor Test

Specimen I.D.	Supporting Beam I.D.	k_{Beam}	Average k_{Beam}
S-3LB-4877-S	3LB-3	1.43	1.39
	3LB-1	1.35	
S-3LB-4877-3STS	3LB-3	1.43	1.39
	3LB-1	1.35	
S-3LB-6096-S	3LB-3	0.75	0.75
	3LB-2	0.75	
S-3LB-6096-3STS	3LB-3	0.75	0.75
	3LB-2	0.75	
S-2LB-6096-S	2LB-2	1.57	1.57
	2LB-4	1.56	
S-2LB-6096-3STS	2LB-2	1.57	1.57
	2LB-4	1.56	

Table 5-8 Comparison between tested and predicted deflection under 1 kN for one-span and beam-supported floor

Specimen I.D.	Tested d under 1 kN (mm)	k	r	C_{dB_1}	d_{ss} (mm)	Predicted d under 1 kN (mm)	Difference in d under 1 kN (%)
S-3LB-4877-S	1.26	1.39	0.00	1.69	0.79	1.34	6.81
S-3LB-4877-3STS	1.18	1.39	0.05	1.64	0.79	1.30	10.51
S-3LB-6096-S	2.04	0.75	0.00	1.38	1.51	2.08	2.17
S-3LB-6096-3STS	1.93	0.75	0.06	1.31	1.51	1.98	2.53
S-2LB-6096-S	2.65	1.57	0.00	1.78	1.51	2.70	1.84
S-2LB-6096-3STS	2.49	1.57	0.06	1.72	1.51	2.60	4.32

Table 5-9 Comparison between tested and predicted fundamental natural frequencies for one-span and beam-supported floor

Specimen I.D.	Tested f_1 (Hz)	k	r	C_{fB_1}	f_{ss} (Hz)	Predicted f_1 (Hz)	Difference in f_1 (%)
S-3LB-4877-S	12.63	1.39	0.00	0.66	17.13	11.34	-10.17
S-3LB-4877-3STS	13.13	1.39	0.05	0.66	17.13	11.34	-13.62
S-3LB-6096-S	8.63	0.75	0.00	0.78	11.50	8.92	3.44
S-3LB-6096-3STS	9.38	0.75	0.06	0.78	11.50	8.91	-4.91
S-2LB-6096-S	7.50	1.57	0.00	0.64	11.50	7.34	-2.10
S-2LB-6096-3STS	8.13	1.57	0.06	0.64	11.50	7.34	-9.68

The predicted deflections under 1 kN obtained from analytical equation are greater than the actual deflections obtained from tests. Most differences are within 10%. In contrast, most predicted frequencies are smaller than tested ones. The differences between tested and predicted frequencies are within also within 10% except the 4877 mm long floor test. It is evident that the relatively significant deflection and frequency comparison differences all occur on the 4877 mm long floor. It may be due to variability in the measurement or experimental error.

5.4.2.2 Beam-supported Two-span Floor

The beam-supported two-span floor tests were conducted on GLT floor panel with 175 mm thickness and two spans of 3658 mm and 2438 mm. There are a couple of reference tests required to determine the fixity factors, supporting beam factors, d_{ss}, and f_{ss}. The one-span floor test having the same configuration as the longer span and simple supports, G175-3658-S, determines d_{ss} and f_{ss}. The detailed references and the data are provided in Table 5-10.

Table 5-10 Fixity Factors and Flexural Stiffness of the One-span Reference Beam Tests

Specimen I.D.	d_{ss} (mm)	f_{ss} (Hz)	K_{Floor} (kN/mm)	r_1	r_2
G175-3658-S	0.37	26.00	2.78	-	-
G175-3658-3STS	-	-	-	0.14	-
G175-S-3STS-S	-	-	-	-	0.63

Table 5-11 Average Supporting Beam Factors of Two-span and Beam-supported Floor Test

Specimen I.D.	Supporting Beam I.D.	k_{Beam} (kN/mm)	Average k_{Beam} (kN/mm)
T-3LB-3:2-S-S-S	3G6-3	3.15	
T-3LB-3:2-S-3STS-S	3G6-1	2.98	3.10
T-3LB-3:2-3STS-3STS-3STS	3G6-2	3.17	
T-2LB-3:2-S-S-S	2G6-2	6.62	
T-2LB-3:2-S-3STS-S	2G6-3	7.20	6.80
T-2LB-3:2-3STS-3STS-3STS	2G6-4	6.58	

The predicted deflections under 1 kN and fundamental natural frequencies of two-span and beam-supported floor using the analytical equations are presented in Table 5-12 and Table 5-13 respectively.

Most predicted deflections and frequencies are greater than tested deflection. The validity of the proposed analytical equation for fundamental natural frequency is verified against the tested frequencies due to the small differences being within 4%, except two outliners. Both outliners occur on the two-span floor with S-S-S boundary conditions. The possible reason for the relatively large error is that the specimens were not tested on perfect simple supports but sitting on the supports. Assuming fixity factors to be zero might be underestimated. The deflection equation provides a close prediction in T-3LB-3:2-3STS-3STS-3STS, which has a difference below 4%. However, the difference in predicting T-2LB-3:2-3STS-3STS-3STS is large.

Table 5-12 Comparison between tested and predicted deflection under 1 kN for two-span and beam-supported floor

Specimen I.D.	Tested d (mm)	k	r_1	r_2	C_{dB_2}	d_{ss} (mm)	Predicted d (mm)	Difference in d (%)
T-3LB-3:2-3STS-3STS-3STS	0.71	3.10	0.14	0.63	2.05	0.36	0.74	3.92
T-2LB-3:2-3STS-3STS-3STS	1.03	6.80	0.14	0.63	3.48	0.36	1.25	20.95

Table 5-13 Comparison between tested and predicted fundamental natural frequencies for two-span and beam-supported floor

Specimen I.D.	Tested f_1 (mm)	k	r_1	r_2	C_{fB_2}	f_{ss} (mm)	Predicted f_1 (mm)	Difference in f_1 (%)
T-3LB-3:2-S-S-S	16.75	3.10	0.00	0.00	0.60	26.00	15.64	-6.65
T-3LB-3:2-S-3STS-S	17.38	3.10	0.00	0.63	0.69	26.00	17.84	2.67
T-3LB-3:2-3STS-3STS-3STS	17.63	3.10	0.14	0.63	0.70	26.00	18.07	2.52
T-2LB-3:2-S-S-S	12.75	6.80	0.00	0.00	0.44	26.00	11.40	-10.60
T-2LB-3:2-S-3STS-S	13.13	6.80	0.00	0.63	0.52	26.00	13.48	2.67
T-2LB-3:2-3STS-3STS-3STS	13.25	6.80	0.14	0.63	0.53	26.00	13.71	3.50

There are two two-span and beam supported floor tests conducted with the same span of 3048 mm but have no corresponding reference tests. The values of parameters, such as connection fixity factors and d_{ss} are unknown. The fixity factors were assumed to be the same, while d_{ss} and K_{Floor} were approximated based on 3658 mm span tests by changing the span length in the equations as shown in Table 5-14.

Table 5-14 Approximated d_{ss} and K_{Floor} for 3048 mm span

Parameter	Reference Specimen I.D.	Value based on 3658 mm span test	Converted value to 3048 mm span
d_{ss} (mm)	G175-3658-S	0.36	0.21
K_{Floor} (kN/mm)	G175-3658-S	2.78	4.81

Due to the new K_{Floor} value, the corresponding average supporting beam stiffness factors are 3.098 and 6.798 respectively. With all the approximated parameters, the comparison between the tested and predicted deflections shown in Table 5-15 can only show the validity of deflection equation.

Table 5-15 Comparison between tested and predicted deflection under 1 kN for two-equal-span and beam-supported floor

Specimen I.D.	Tested d (mm)	k	r_1	r_2	C_{dB_2}	d_{ss} (mm)	Predicted d (mm)	Difference in d (%)
T-3LB-1:1-3STS	0.624	5.354	0.137	0.634	3.144	0.208	0.653	4.620
T-2LB-1:1-3STS	0.917	11.751	0.137	0.634	5.728	0.208	1.190	29.863

Similar to Table 5-12, both predictions are greater than tested data and the prediction of the floor supported by 3-layer beams are more accurate than the one supported by 2-layer beams. It can be concluded that the analytical equation for deflection may only be valid for floors supported by stiff beams. In practice, the stiffness of the beam is always high, even higher than the floors, so the analytical equation may still be valid. Further experimental and numerical work is recommended.

5.5 Effects of Beam Stiffness on the Natural Frequency of One-way and Beam-supported Floor

This section focuses on the effect of beam stiffness on the natural frequency of one-way and beam-supported MTP floor, including both one-span and two-span structures using Eq. (5.12) and (5.15). The supports of MTP floors are assumed to be simple to isolate the effect of supporting beam stiffness. Thus, the fixity factors of all connections are zero. Assuming the flexural stiffness of beam is greater than that of floor panel, the supporting beam stiffness factor ranges between zero and one.

There are two one-span floor configurations considered. The first one is a floor supporting by two identical beams, which have equal supporting beam stiffness factors. The second one is a floor supported by a supporting beam and a rigid support. Since the stiffness of a rigid support is infinite, the corresponding supporting beam stiffness factor is zero. The following Figure 5-14 shows the effect of supporting beam stiffness on coefficient C_{fB_1} of the two configurations.

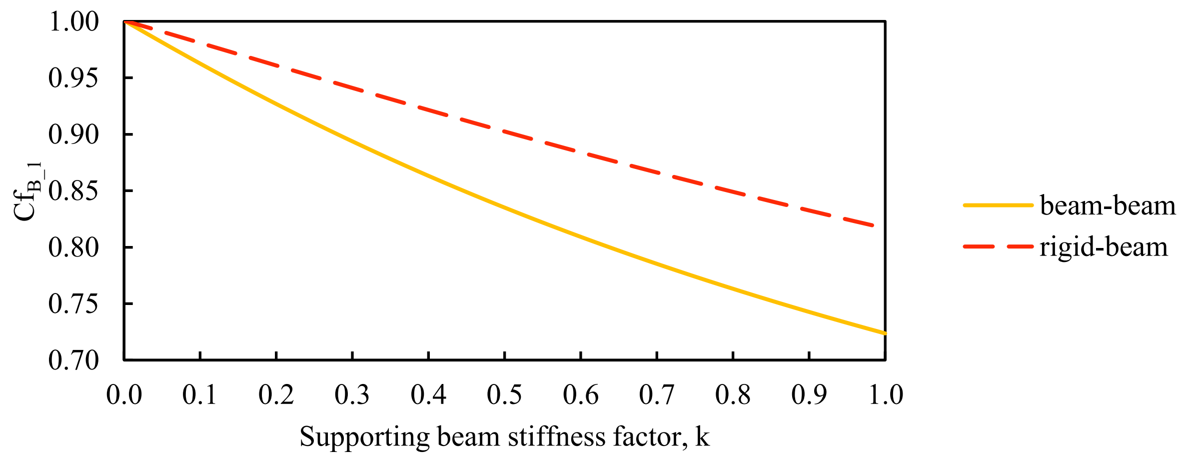


Figure 5-14 Effect of supporting beam stiffness factor on coefficient C_{fB_1}

It can be observed that as the supporting beam stiffness factor increases, the corresponding coefficients C_{fB_1} of both configurations decrease. The coefficient C_{fB_1} is the ratio between the beam-supported beam and simple beam. The smaller C_{fB_1} indicates a smaller fundamental natural frequency. The supporting beam stiffness factor increases along with the smaller supporting beam stiffness. Therefore, decreasing the supporting beam stiffness will lead to smaller fundamental natural frequency of MTP floor. Generally, the floor supported by one beam and one rigid support has greater frequency than floor supporting by two beams. The frequency difference is more significant when the beam stiffness is small. Specifically, for one-span floor, to reach 80% and 90% of the first natural frequency of floor with simple supports, the recommended supporting beam stiffness factors are shown in Table 5-16 for both supporting configurations.

Table 5-16 Recommended supporting beam stiffness factors to reach 80% and 90% first natural frequency of one-span simple floor

Percent of frequency of one-span floor with simple supports	Supporting beam stiffness factor for beam-beam	Supporting beam stiffness factor for rigid-beam
80%	0.64	1.11
90%	0.28	0.51

Figure 4-3 shows a typical two-span MTP floor, where support 1 and 3 are the end supports of the longer and shorter span respectively and support 2 is the intermediate support. The fixity factors of all supports are assumed to be zero as well. Three configurations are considered: 1. floor supported by three identical beams (beam-beam-beam); 2. a rigid support at 1 and supports 2 and

3 on identical beams (rigid-beam-beam); 3. a beam at support 2 and supports 1 and 3 on rigid support (rigid-beam-rigid). Three span ratios considered were 1/3, 1/2, and 2/3. The relationship between coefficient C_{fB_2} and supporting beam stiffness factor of two-span MTP floor configurations is shown in Figure 5-15.

Figure 5-15 indicates the smaller supporting beam stiffness factor can lead to smaller frequency of two-span MTP floor. The frequency of floor supported by three beams is significantly smaller than other two configurations. The curves of rigid-beam-beam support and rigid-beam-rigid support are very close. It also verifies the sensitivity analysis results presented in Section 5.4.1.2, which show the beam stiffness factor of support 3 has little impact on the frequency.

The recommended beam stiffness factors to maintain the same, 95%, and 90% frequency as the simple one-span floor with same length as the first span are shown in Table 5-17.

Table 5-17 Recommended supporting beam stiffness factors to reach 100%, 95%, and 90% of first natural frequency of simple floor with same length as longer span

Percent of frequency of one-span floor with simple supports	beam-beam-beam			rigid-beam-beam			rigid-beam-rigid		
	100%	95%	90%	100%	95%	90%	100%	95%	90%
$\mu = 1/3$	0.33	0.43	0.55	0.49	0.68	0.97	0.55	0.74	1.01
$\mu = 1/2$	0.42	0.54	0.69	0.72	1.01	1.47	0.76	1.05	1.44
$\mu = 2/3$	0.50	0.65	0.83	1.07	1.55	2.26	1.00	1.40	2.03

Figure 5-16 is the rearranged Figure 5-15 to compare the effect of beam stiffness on frequency crossing different span ratios. It illustrates the fact that a shorter span next to a large span increases the frequency of the large span more when the beam stiffness factor is below 0.11. As the beam stiffness factor increases, the effect of shorter second span will eventually become minor than longer second span.

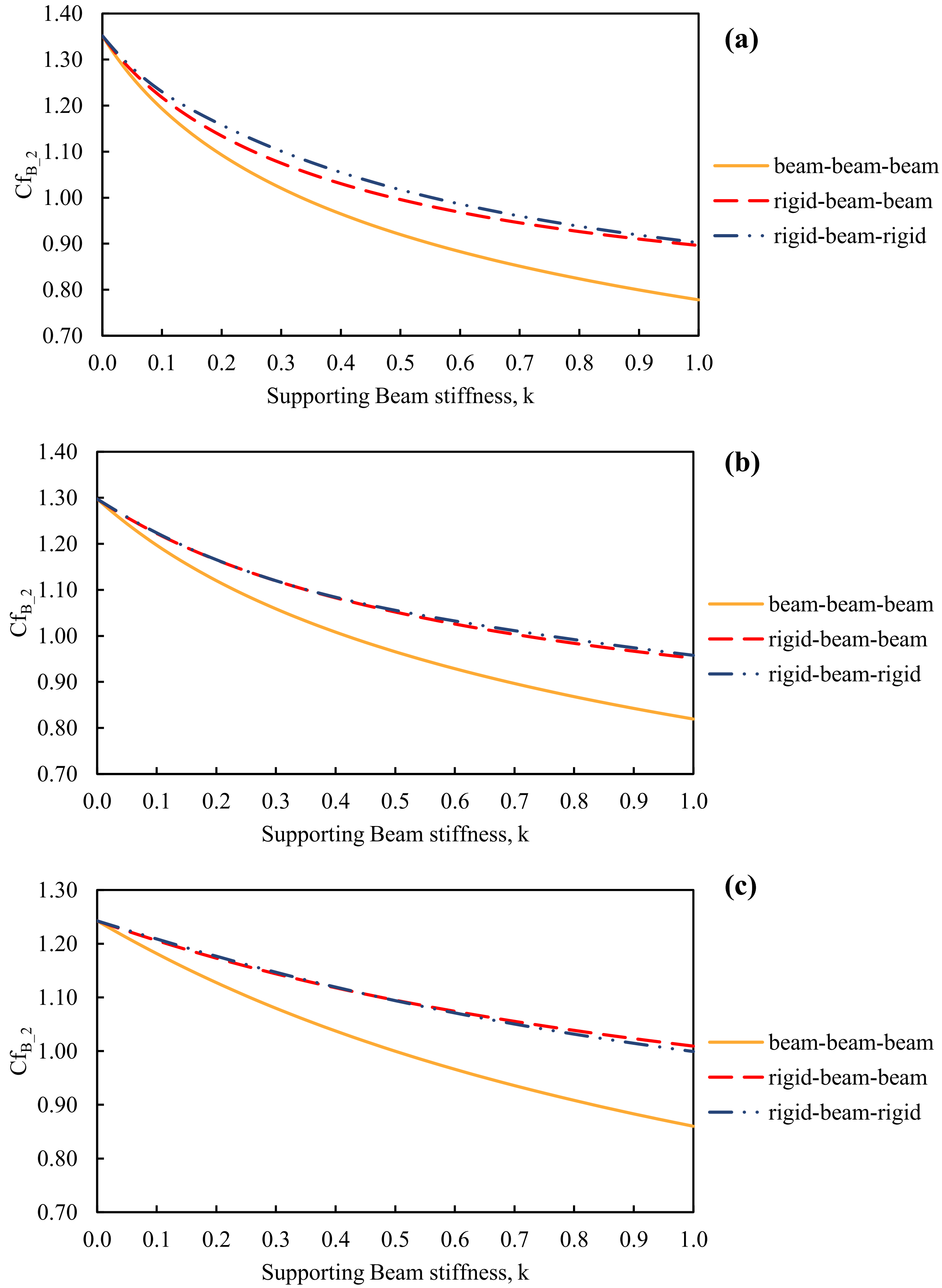


Figure 5-15 Effect of supporting beam stiffness factor on C_{fB_2} (a) $\mu = 1/3$; (b) $\mu = 1/2$; (c) $\mu = 2/3$

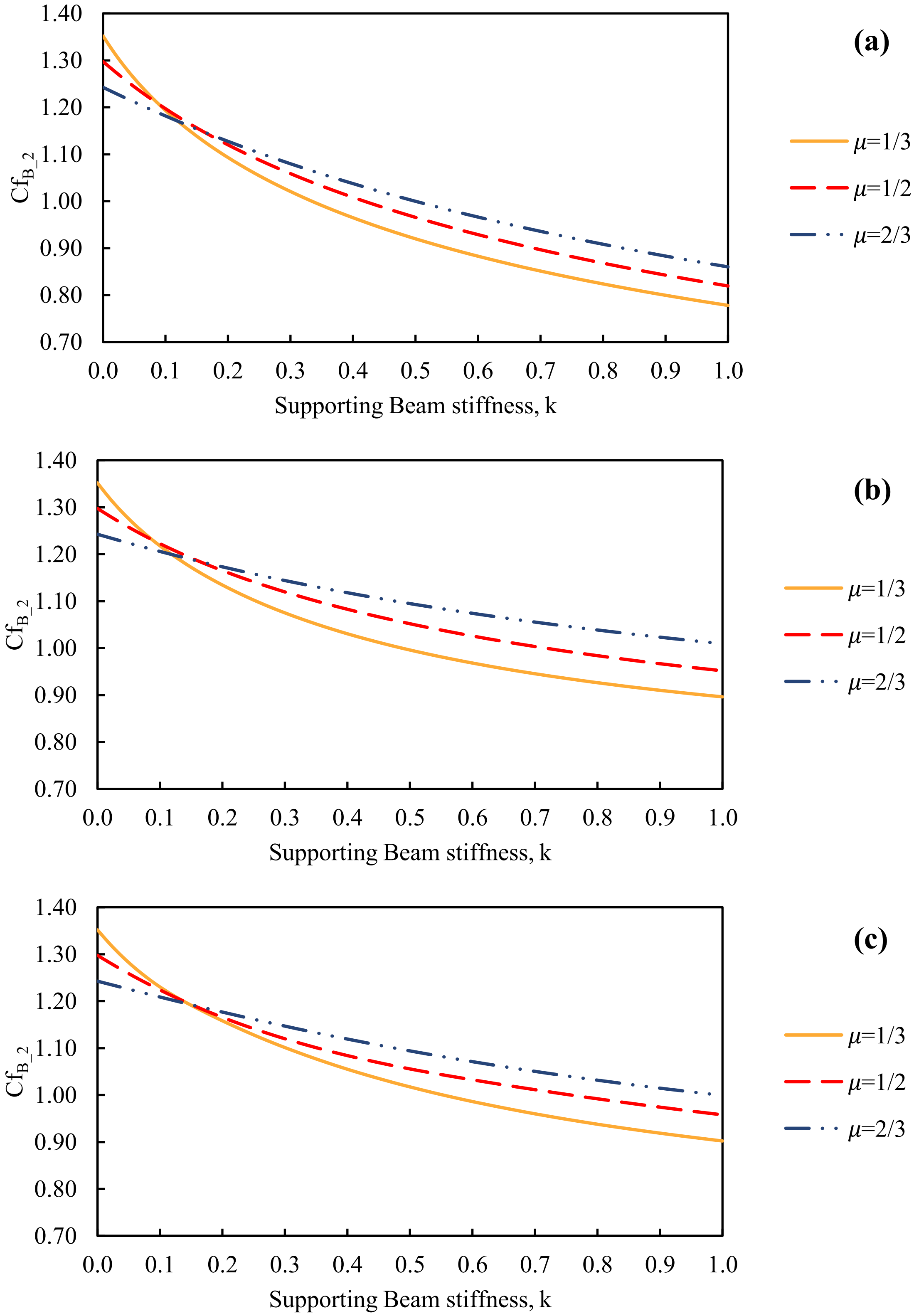


Figure 5-16 Effect of supporting beam stiffness factor on C_{fB_2} (a) beam-beam-beam; (b) rigid-beam-beam; (c) rigid-beam-rigid

5.6 Summary and Conclusions

Analytical expressions for static deflection and fundamental natural frequency of one-span and two-span MTP with restrained connections are proposed in this Chapter. Supporting beam stiffness factor, which is the ratio of flexural stiffness of floor panel to that of supporting beam, is introduced to simplify the analytical expressions and make the coefficients in the analytical expressions including C_{dB_1} , C_{dB_2} , C_{fB_1} , and C_{fB_2} dimensionless. Due to the extensive length of the equations defining coefficients C_{dB_2} and C_{fB_2}, sensitivity analyses were conducted, and the results showed that parameters $r_3$and k_3 have insignificant effects. The expressions were then simplified by substituting r_3 with r_1.

Verifications were performed against the experimental results from Chapter 3. For one-span floor, both predicted deflections and frequencies have relatively good agreement with test results as most differences are below 10%. For two-span floor, both predictions are very close to test results of floor supported by stiffer beams.

In addition, a separate investigation was conducted on the effect of beam stiffness on the fundamental natural frequency solely based on analytical expressions presented in this chapter. Both one-way and two-way structures were included, and they have 2 and 3 different supporting configurations respectively. The recommended supporting beam stiffness factors are provided to reach satisfactory first natural frequencies.

Chapter 6 - Conclusions and Recommendations

6.1 Summary

MTP panels are widely adopted as load-carrying plate elements in buildings, such as walls and floors. With the development of the MTP panels, the longer span can be achieved. In addition to MTP panel's property, such as low damping ratio and high stiffness to weight ratio, the vibration serviceability of MTP floors need to be properly designed. Current Canadian timber floor vibration serviceability design criterion consists of two parameters, static deflection under point load applied at mid-span and fundamental natural frequency. The current floor vibration design approach assumes simple support condition only. Such support condition is not often met in practice. This study focused on the development and verification analytical models to allow for other more practical support conditions to be considered in design calculations. These include the effects of multi-span continuity, rotational stiffness of support connections, and stiffness of supporting beam.

In total 49 MTP floor specimens with different configurations, connections, and support conditions were tested. A series of analytical models have been proposed to provide the equations of vibration serviceability design criterion parameters for one-span and two-span and rotationally restrained MTP floor, with and without beam support.

This chapter discusses the conclusions derived from this study and recommends areas for further research.

6.2 Conclusions

The conclusions from this study are as follows:

<u>Two-span MTP floors with restrained connections and rigid supports</u>

1. Both deflection and frequency become insensitive to the fixity of the middle support when the fixity factor is high.
2. The effect of end connection fixity factor is more significant on deflection in thinner floor panel. In contrast, the effect on natural frequency is more significant in thicker panels.

3. Two sensitivity analyses were performed of the proposed analytical expressions for static deflection and first natural frequency. The results indicate that the fixity factor of the end connection of shorter span has little effects on end results obtained from these analytical expressions. Therefore, they are substituted with the fixity factor of the end connection of longer span for simplification.
4. The original and simplified analytical expressions were verified against the test results of two-span MTP floor on wall support, and a relatively good agreement was found since differences of most specimens are within 10%.

<u>One-span MTP floors with restrained connections and beam supports</u>

1. Three one-span floors supported by beams were tested under two different connections, namely simple support and 3STS (3 self-tapping screws). The differences in deflection and frequency were within 10%.
2. When the stiffness of the supporting beams was doubled, the deflections decreased by 23% for both connections; and the first natural frequencies increased by 15% for both connections.
3. The analytical expressions were verified against the test results of one-span MTP floor on beam support tests. The differences between predicted and tested deflections and frequencies were less than 14% with the majority of the difference being less than 10%.
4. The effect of supporting beam stiffness on fundamental natural frequency solely was investigated based on the analytical equation with fixity factors all equal to zero. The two support systems considered are beam-beam and rigid-beam. The decrease of supporting beam stiffness contributed to smaller natural frequency of both support systems, except the change is steeper for beam-beam support. The recommended beam stiffness factors to maintain 80% of fundamental natural frequency of floor on rigid supports are 0.64 (beam-beam) and 1.11 (rigid-beam). Whereas, to reach 90% of fundamental natural frequency of floor on rigid supports, the recommended beam stiffness factors are 0.28 (beam-beam) and 0.51 (rigid-beam).

<u>Two-span MTP floors with restrained connections and beam supports</u>

1. Four two-span floors supported by beams were tested with two different stiffnesses of supporting beam. When the stiffness of supporting beams increases by almost 120%, the deflections decrease by 31% and first natural frequencies increase by more than 30%.
2. Test results showed that with the same total length, the floor with a span ratio closer to 1 has a smaller deflection. However, the effect of the span ratio on natural frequency was insignificant based on limited test data.
3. Two sensitivity analyses were performed of the proposed analytical expressions for static deflection and first natural frequency. The results indicated that the fixity factor and beam stiffness factor of the end connection of shorter span has little effects on both equations. Both expressions are simplified by substituting the fixity factor of end connection of shorter span with the fixity factor of the end connection of longer span.
4. The analytical expressions were verified against the test results of two-span MTP floor on beam support tests. A very good agreement was found between predicted and tested frequencies since the differences were below 2%. In contrast, analytical expression only provided a good prediction of deflection of floors supported by 3LB beams, which were the stiffer beams. In practice, the stiffness of beams is always high, often higher than the floors. Therefore, the analytical equation may still be valid.
5. For the two-span, beam-supported MTP floors, the effect of supporting beam stiffness on fundamental natural frequency was also investigated independently. All connection fixity factors were set to zero, and three support systems were considered: beam-beam-beam, rigid-beam-beam, and rigid-beam-rigid. The recommended beam stiffness factors in maintaining the 100%, 95%, and 90% of the frequency of simple one-span floor with the same length as the first span and span ratio is 1/3 are 0.33(100%)/0.43(95%)/0.55(90%) for beam-beam-beam, 0.49/0.68/0.97 (rigid-beam-beam), and 0.55/0.74/1.01 (rigid-beam-rigid), respectively. When the span ratio is 1/2, the recommended beam stiffness factors become 0.42/0.54/0.69 (beam-beam-beam), 0.72/1.01/1.47 (rigid-beam-beam), and 0.76/1.05/1.44 (rigid-beam-rigid). When the span ratio changes to 2/3, the recommended beam stiffness factors are 0.50/0.65/0.83 (beam-beam-beam), 1.07/1.55/2.26 (rigid-beam-beam), 1.00/1.040/2.03 (rigid-beam-rigid).

6.3 Recommendations for Future Research

The analytical equations developed are applicable to predict the static deflection and fundamental natural frequency for the following MTP floors:

1. Two-span and rotationally restrained MTP floor
2. One-span, rotationally restrained, and beam-supported MTP floor
3. Two-span, rotationally restrained, and beam-supported MTP floor (except the floor with equal spans and same end connections and supports)

The specific supporting beam stiffness factors in maintaining the satisfactory fundamental natural frequency of beam-supported MTP floors, both one-span and two-span, are recommended.

Some areas that would benefit from further study include:

1. Testing and further investigation of the intermediate support fixity factor of various connection types.
2. Further investigation of deflection expression for two-span and beam-supported against the beam stiffness factor.
3. Modifying the fundamental natural frequency equation for two-span, rotationally restrained, and beam-supported MTP floor to become applicable to the floor with equal spans and same end connections and supports.
4. Further simplification of the analytical models to facilitate design use.

References

Abeysekera, I. K., Hamm, P., Toratti, T., and Lawrence, A. 2018. Development of a floor vibration design method for Eurocode 5. New Zealand timber design, **27**(1): 23–31.

Brancheriau, L., and Bailleres, H. 2002. Natural vibration analysis of clear wooden beams: a theoretical review. Wood science and technology, **36**(4): 347–365. doi:10.1007/s00226-002-0143-7.

Canadian Commission on Building and Fire Codes (BCBFC). 2010. National Building Code of Canada: 2010. National Research Council of Canada.

Canadian Construction Materials Centre (CCMC). 1997. Concluding Report: Development of Design Procedures for Vibration Controlled Spans Using Engineered Wood Members. Canadian Construction Materials Centre, NRC, Ottawa.

Canadian Standards Association (CSA) O86-14 Updates 1 and 2. 2016. Engineering Design in Wood (Limit States Design). Canadian Standards Association, Toronto, Canada.

Canadian Standards Association (CSA) O86-19. 2019. Engineering Design in Wood (Limit States Design). Canadian Standards Association, Toronto, Canada.

Catterou, T., Castaing, J. B., and Garcia, P. 2020. Simplified computation of timber floor vibrations level due to human activities. XI International Conference on Structural Dynamics. Athens, Greece. pp. 1986-1912. doi: 10.47964/1120.9154.18373

Chui, P. P. T., and Chan, S. L. 1997. Vibration and deflection characteristics of semi-rigid jointed frames. Engineering Structures, 19(12): 1001–1010. doi: 10.1016/S0141-0296(97)00126-0.

Chui, Y. H., and Smith, I. 1990. Influence of rotatory inertia, shear deformation and support condition on natural frequencies of wooden beams. Wood science and technology, **24**(3): 233–245. doi:10.1007/BF01153557.

Clough, R. W., and Penzien, J. 1975. Dynamics of structures. McGraw-Hill, New York.

Elliott, K. S., Davies, G., Ferreira, M., Halil, G., and Mahdi, A. A. 2003. Can precast concrete structures be designed as semi-rigid frames? Part 1 - The experimental evidence. Protein Structure Initiative Structural Genomics Knowledgebase, 81: 14–27.

Hernández, S. A. M., and Chui, Y. H. 2012. Vibrational performance of cross laminated timber floors. World Conference on Timber Engineering. Auckland, New Zealand.

Hernández, S. A. M., and Chui, Y. H. 2014. Effect of end support conditions on the vibrational performance of cross-laminated timber floors. World Conference on Timber Engineering. Quebec City, Canada.

Hu, D., Li, G., and Sun, F. 2008. Practical analysis of natural periods for semi-rigidly connected multi-story composite frames. World Earthquake Engineering, **24**(2): 68–72. [Chinese].

Hu, L., and Chui, Y. H. 2004. Development of a design method to control vibrations induced by normal walking action in wood-based floors. World Conference on Timber Engineering, **2**: 217–222.

Hu, L., Cuerrier-Audair, S., Chui, Y. H., Ramzi, R., Gagnon, S., Mohammad, M., Ni, C., and Popovski, M. 2016. Design method for controlling vibrations of wood-concrete composite floors systems. World Conference on Timber Engineering, pp. 4237–4245.

Hu, L., and Gagnon, S. 2012. Controlling cross-laminated timber (CLT) floor vibrations: fundamentals and methods. World Conference on Timber Engineering, pp. 269–275.

Jacobson, L. S., and Ayre, R. S. 1958. Engineering Vibrations. McGraw-Hill, New York.

Jones, S. W., Kirby, P.A., and Nethercort, D.A. 1983. The analysis of frames with semi-rigid connections—a state-of-the-art report. Journal of Constructional Steel Research, **3**(2): 2–13.

Karacabeyli, E., and Gagnon, S. (Editor). 2019. Canadian CLT handbook, 2019 Edition. FPInnovations, Pointe-Claire, QC.

Kawashima, S., and Fujimoto, T. 1984. Vibration analysis of frames with semi-rigid connections. Computers & structures, **19**(1-2): 85–92. doi: 10.1016/0045-7949(84)90206-2.

Monforton G. R., and Wu T. S. 1963. Matrix Analysis of Semi-Rigidly Connected Frames. Journal of the Structural Division, **89**(6): 13–42. doi: 10.1061/JSDEAG.0000997.

Ohlsson, S. 1988. Springiness and Human-induced Floor Vibrations: A Design Guide (No. 1988). Swedish Council for Building Research.

Petersen, C., and Werkle, H. 2018. Bautechnische Anwendungen zur Dynamik. In Dynamik der Baukonstruktionen. Springer-Verlag. pp. 282–283. [German].

Qi, J. J., and Jiang, L. Z. 2010. Effects of interface slip and semi-rigid joint on elastic seismic response of steel-concrete composite frames. Journal of Central South University of Technology, **17**(6): 1327–1335. doi: 10.1007/s11771-010-0638-6.

Rathbun, J. C. 1936. Elastic properties of riveted connections. Transactions of the American Society of Civil Engineers, **101**(1): 524–563. doi: 10.1061/taceat.0004766.

Steel Structures Research Committee (SSRC). 1934. Second Report. Department of Scientific and Industrial Research.

Smith, I., and Chui, Y. H. 1988. Design of lightweight wooden floors to avoid human discomfort. Canadian Journal of Civil Engineering, **15**(2): 254–262. doi: 10.1139/l88-033.

Weckendorf, J., Toratti, T., Smith, I., and Tannert, T. 2016. Vibration serviceability performance of timber floors. European Journal of Wood and Wood Products, **74**(3): 353–367. doi:10.1007/s00107-015-0976-z.

Zhang, W., Ma, H., Zeng, J., Wu, S., and Wen, B. 2017. Vibration responses analysis of an elastic-support cantilever beam with crack and offset boundary. Mechanical systems and signal processing, 95: 205–218. doi:10.1016/j.ymssp.2017.03.032.

Zhang, S., Zhou, J., Niederwestberg, J., and Chui, Y. H. 2019. Effect of end support restraints on vibration performance of cross laminated timber floors: An analytical approach. Engineering Structures, 189: 186–194. doi:10.1016/j.engstruct.2019.03.042.

Appendix A

$C_{f\_rrr} =$

3.93982 (((4. (-4. + r1) (-1. + r2) + μ (-4. + r1 r2) (-4. + r3)) (4. (-3. + 2. r1) (-1. + r2) - 1. μ (8. - 5. r2 + r1 (-5. + 3. r2)) (-4. + r3) - 10. μ^3 (-2. + r1) (-1. + r2) (-2. + r3) + μ^6 (-4. + r1 r2) (-3. + 2. r3) - 1. μ^5 (-4. + r1) (-1. + r2) (-8. + 5. r3))) / (16. (76. - 101. r1 + 34. r1^2) (-1. + r2) ^ 2 - 4. μ (-1. + r2) (r1 (488. - 286. r2) + 76. (-5. + 3. r2) + r1^2 (-159. + 91. r2)) (-4. + r3) + μ^2 (496. - 612. r2 + 192. r2^2 + r1 (-612. + 736. r2 - 225. r2^2) + r1^2 (192. - 225. r2 + 67. r2^2)) (-4. + r3) ^ 2 - 12. μ^3 (152. - 178. r1 + 53. r1^2) (-1. + r2) ^ 2 (-2. + r3) + 24. μ^6 (32. - 31. r1 + 8. r1^2) (-1. + r2) ^ 2 (-2. + r3) ^ 2 + 3. μ^4 (-1. + r2) (r1 (452. - 274. r2) + 8. (-51. + 32. r2) + r1^2 (-128. + 75. r2)) (8. - 6. r3 + r3^2) + 24. μ^5 (-2. + r1) ^2 (-1. + r2) ^ 2 (32. - 31. r3 + 8. r3^2) + 12. μ^7 (8. - 6. r1 + r1^2) (-1. + r2) ^ 2 (102. - 113. r3 + 32. r3^2) + μ^11 (-4. + r1 r2) ^ 2 (76. - 101. r3 + 34. r3^2) + 4. μ^9 (-4. + r1) ^ 2 (-1. + r2) ^ 2 (124. - 153. r3 + 48. r3^2) - 3. μ^8 (-2. + r1) (-1. + r2) (-4. + r1 r2) (152. - 178. r3 + 53. r3^2) - 1. μ^10 (-4. + r1) (-1. + r2) (-4. + r1 r2) (380. - 488. r3 + 159. r3^2))) ^ (1/2)

$C_{f\_rrr}\ (r_3 = r_1) =$

3.93982 (((4. (-4. + r1) (-1. + r2) + μ (-4. + r1) (-4. + r1 r2)) (-10. μ^3 (-2. + r1) ^2 (-1. + r2) + 4. (-3. + 2. r1) (-1. + r2) - 1. μ^5 (-4. + r1) (-8. + 5. r1) (-1. + r2) + μ^6 (-3. + 2. r1) (-4. + r1 r2) - 1. μ (-4. + r1) (8. - 5. r2 + r1 (-5. + 3. r2)))) / (24. μ^5 (-2. + r1) ^2 (32. - 31. r1 + 8. r1^2) (-1. + r2) ^2 + 24. μ^6 (-2. + r1) ^2 (32. - 31. r1 + 8. r1^2) (-1. + r2) ^2 + 12. μ^7 (8. - 6. r1 + r1^2) (102. - 113. r1 + 32. r1^2) (-1. + r2) ^2 + 16. (76. - 101. r1 + 34. r1^2) (-1. + r2) ^2 + 4. μ^9 (-4. + r1) ^2 (124. - 153. r1 + 48. r1^2) (-1. + r2) ^2 - 12. μ^3 (-2. + r1) (152. - 178. r1 + 53. r1^2) (-1. + r2) ^2 - 3. μ^8 (-2. + r1) (152. - 178. r1 + 53. r1^2) (-1. + r2) (-4. + r1 r2) - 1. μ^10 (-4. + r1) (380. - 488. r1 + 159. r1^2) (-1. + r2) (-4. + r1 r2) + μ^11 (76. - 101. r1 + 34. r1^2) (-4. + r1 r2) ^2 + 3. μ^4 (8. - 6. r1 + r1^2) (-1. + r2) (r1 (452. - 274. r2) + 8. (-51. + 32. r2) + r1^2 (-128. + 75. r2)) - 4. μ (-4. + r1) (-1. + r2) (r1 (488. - 286. r2) + 76. (-5. + 3. r2) + r1^2 (-159. + 91. r2)) + μ^2 (-4. + r1) ^2 (496. - 612. r2 + 192. r2^2 + r1 (-612. + 736. r2 - 225. r2^2) + r1^2 (192. - 225. r2 + 67. r2^2)))) ^ (1/2)

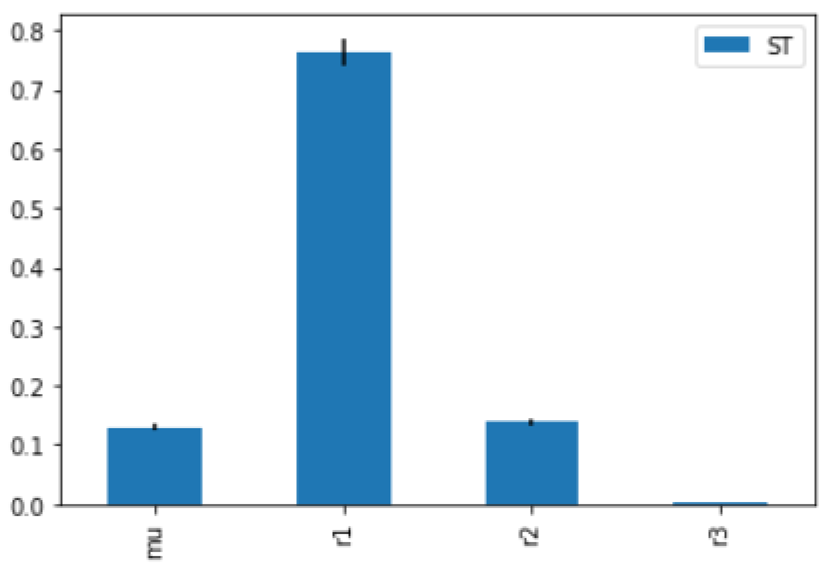
Sensitivity analysis codes and results

```
# Cdrrr
from SALib.sample import saltelli
from SALib.analyze import sobol
import numpy as np
from SALib.plotting.bar import plot as barplot
import matplotlib.pyplot as plot
import math
#Define the model inputs
problem_Cdrrr = {
    'num_vars': 4,
    'names': ['mu', 'r1', 'r2', 'r3'],
    'bounds': [[0, 1], [0, 1], [0, 1], [0, 1]]}
def evaluate_Cdrrr(X):
    return np.array([(4*(x[0] + 1)*(64*x[0] - 16*x[1] - 28*x[2] - 36*x[0]*x[1] - 36*x[0]*x[2] - 
16*x[0]*x[3] + 16*x[1]*x[2] + 20*x[0]*x[1]*x[2] + 9*x[0]*x[1]*x[3] + 9*x[0]*x[2]*x[3] - 
5*x[0]*x[1]*x[2]*x[3] + 28))/((16*x[0] + 7)*(16*x[0] - 4*x[1] - 16*x[2] - 4*x[0]*x[3] + 4*x[1]*x[2] - 
4*x[0]*x[1]*x[2] + x[0]*x[1]*x[2]*x[3] + 16))  for x in X])
# Generate samples
param_values_Cdrrr = saltelli.sample(problem_Cdrrr, 8192)
# Run model
Y_Cdrrr = evaluate_Cdrrr(param_values_Cdrrr)
# Generate the dimension of sample data
print(param_values_Cdrrr.shape, Y_Cdrrr.shape)
# Perform analysis
Si_Cdrrr = sobol.analyze(problem_Cdrrr, Y_Cdrrr, print_to_console=True)
# Print the first-order, second-order, total-order sensitivity indices
print()
# Generate sensitivity plot
Si_df_Cdrrr = Si_Cdrrr.to_df()
barplot(Si_df_Cdrrr[0])
plot.show()
```

	ST	ST_conf
mu	0.129783	0.006024
r1	0.763345	0.023338
r2	0.138090	0.005490
r3	0.002097	0.000096

	S1	S1_conf
mu	0.105645	0.011412
r1	0.754350	0.024467
r2	0.107309	0.011687
r3	0.001408	0.001407

	S2	S2_conf
(mu, r1)	0.000416	0.021046
(mu, r2)	0.021917	0.017205
(mu, r3)	0.000112	0.016811
(r1, r2)	0.006830	0.032064
(r1, r3)	0.000114	0.026286
(r2, r3)	0.000324	0.016395

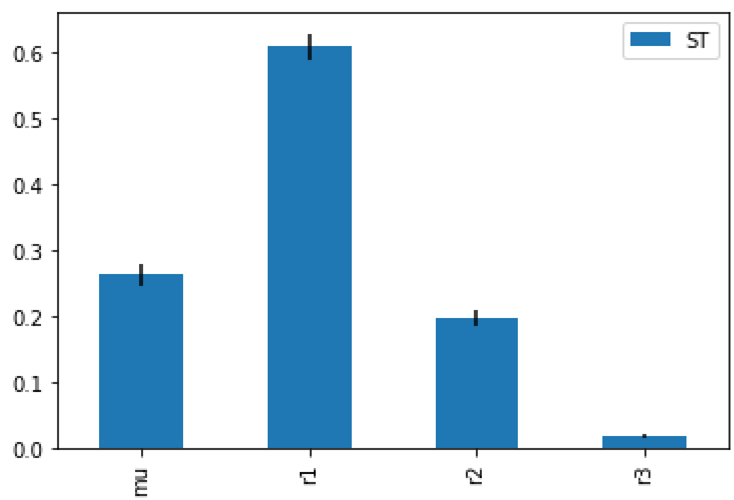


```
# Cfrrr
from SALib.sample import saltelli
from SALib.analyze import sobol
import numpy as np
from SALib.plotting.bar import plot as barplot
import matplotlib.pyplot as plot
import math
#Define the model inputs
problem_Cfrrr = {
    'num_vars': 4,
    'names': ['mu', 'r1', 'r2', 'r3'],
    'bounds': [[0, 1], [0, 1], [0, 1], [0, 1]]}
def evaluate_Cfrrr(X):
    return np.array([((4951760157141521099596496896*(32*x[0] - 8*x[1] - 12*x[2] - 20*x[0]*x[1] -
20*x[0]*x[2] - 8*x[0]*x[3] + 8*x[1]*x[2] - 20*x[0]**3*x[1] - 40*x[0]**3*x[2] - 20*x[0]**3*x[3] -
8*x[0]**5*x[1] - 32*x[0]**5*x[2] - 20*x[0]**5*x[3] - 8*x[0]**6*x[3] + 40*x[0]**3 + 32*x[0]**5 +
12*x[0]**6 + 20*x[0]**3*x[1]*x[2] + 10*x[0]**3*x[1]*x[3] + 20*x[0]**3*x[2]*x[3] + 8*x[0]**5*x[1]*x[2]
+ 5*x[0]**5*x[1]*x[3] - 3*x[0]**6*x[1]*x[2] + 20*x[0]**5*x[2]*x[3] + 12*x[0]*x[1]*x[2] +
5*x[0]*x[1]*x[3] + 5*x[0]*x[2]*x[3] - 3*x[0]*x[1]*x[2]*x[3] - 10*x[0]**3*x[1]*x[2]*x[3] -
5*x[0]**5*x[1]*x[2]*x[3] + 2*x[0]**6*x[1]*x[2]*x[3] +
12))/(1157631494221306995318550230033375*(((192*x[0]**6*x[1]**2*x[2]**2*x[3]**2 -
768*x[0]**6*x[1]**2*x[2]**2*x[3] + 768*x[0]**6*x[1]**2*x[2]**2 - 384*x[0]**6*x[1]**2*x[2]*x[3]**2
+ 1536*x[0]**6*x[1]**2*x[2]*x[3] - 1536*x[0]**6*x[1]**2*x[2] + 192*x[0]**6*x[1]**2*x[3]**2 -
768*x[0]**6*x[1]**2*x[3] + 768*x[0]**6*x[1]**2 - 744*x[0]**6*x[1]*x[2]**2*x[3]**2 +
2976*x[0]**6*x[1]*x[2]**2*x[3] - 2976*x[0]**6*x[1]*x[2]**2 + 1488*x[0]**6*x[1]*x[2]*x[3]**2 -
5952*x[0]**6*x[1]*x[2]*x[3] + 5952*x[0]**6*x[1]*x[2] - 744*x[0]**6*x[1]*x[3]**2 +
2976*x[0]**6*x[1]*x[3] - 2976*x[0]**6*x[1] + 768*x[0]**6*x[2]**2*x[3]**2 -
3072*x[0]**6*x[2]**2*x[3] + 3072*x[0]**6*x[2]**2 - 1536*x[0]**6*x[2]*x[3]**2 +
6144*x[0]**6*x[2]*x[3] - 6144*x[0]**6*x[2] + 768*x[0]**6*x[3]**2 - 3072*x[0]**6*x[3] +
3072*x[0]**6 + 225*x[0]**4*x[1]**2*x[2]**2*x[3]**2 - 1350*x[0]**4*x[1]**2*x[2]**2*x[3] +
1800*x[0]**4*x[1]**2*x[2]**2 - 609*x[0]**4*x[1]**2*x[2]*x[3]**2 + 3654*x[0]**4*x[1]**2*x[2]*x[3] -
4872*x[0]**4*x[1]**2*x[2] + 384*x[0]**4*x[1]**2*x[3]**2 - 2304*x[0]**4*x[1]**2*x[3] +
3072*x[0]**4*x[1]**2 - 822*x[0]**4*x[1]*x[2]**2*x[3]**2 + 4932*x[0]**4*x[1]*x[2]**2*x[3] -
6576*x[0]**4*x[1]*x[2]**2 + 2178*x[0]**4*x[1]*x[2]*x[3]**2 - 13068*x[0]**4*x[1]*x[2]*x[3] +
17424*x[0]**4*x[1]*x[2] - 1356*x[0]**4*x[1]*x[3]**2 + 8136*x[0]**4*x[1]*x[3] - 10848*x[0]**4*x[1] +
768*x[0]**4*x[2]**2*x[3]**2 - 4608*x[0]**4*x[2]**2*x[3] + 6144*x[0]**4*x[2]**2 -
1992*x[0]**4*x[2]*x[3]**2 + 11952*x[0]**4*x[2]*x[3] - 15936*x[0]**4*x[2] + 1224*x[0]**4*x[3]**2 -
7344*x[0]**4*x[3] + 9792*x[0]**4 - 636*x[0]**3*x[1]**2*x[2]**2*x[3] +
1272*x[0]**3*x[1]**2*x[2]**2 + 1272*x[0]**3*x[1]**2*x[2]*x[3] - 2544*x[0]**3*x[1]**2*x[2] -
```

```
636*x[0]**3*x[1]**2*x[3] + 1272*x[0]**3*x[1]**2 + 2136*x[0]**3*x[1]*x[2]**2*x[3] -
4272*x[0]**3*x[1]*x[2]**2 - 4272*x[0]**3*x[1]*x[2]*x[3] + 8544*x[0]**3*x[1]*x[2] +
2136*x[0]**3*x[1]*x[3] - 4272*x[0]**3*x[1] - 1824*x[0]**3*x[2]**2*x[3] + 3648*x[0]**3*x[2]**2 +
3648*x[0]**3*x[2]*x[3] - 7296*x[0]**3*x[2] - 1824*x[0]**3*x[3] + 3648*x[0]**3 +
67*x[0]**2*x[1]**2*x[2]**2*x[3]**2 - 536*x[0]**2*x[1]**2*x[2]**2*x[3] +
1072*x[0]**2*x[1]**2*x[2]**2 - 225*x[0]**2*x[1]**2*x[2]*x[3]**2 + 1800*x[0]**2*x[1]**2*x[2]*x[3] -
3600*x[0]**2*x[1]**2*x[2] + 192*x[0]**2*x[1]**2*x[3]**2 - 1536*x[0]**2*x[1]**2*x[3] +
3072*x[0]**2*x[1]**2 - 225*x[0]**2*x[1]*x[2]**2*x[3]**2 + 1800*x[0]**2*x[1]*x[2]**2*x[3] -
3600*x[0]**2*x[1]*x[2]**2 + 736*x[0]**2*x[1]*x[2]*x[3]**2 - 5888*x[0]**2*x[1]*x[2]*x[3] +
11776*x[0]**2*x[1]*x[2] - 612*x[0]**2*x[1]*x[3]**2 + 4896*x[0]**2*x[1]*x[3] - 9792*x[0]**2*x[1] +
192*x[0]**2*x[2]**2*x[3]**2 - 1536*x[0]**2*x[2]**2*x[3] + 3072*x[0]**2*x[2]**2 -
612*x[0]**2*x[2]*x[3]**2 + 4896*x[0]**2*x[2]*x[3] - 9792*x[0]**2*x[2] + 496*x[0]**2*x[3]**2 -
3968*x[0]**2*x[3] + 7936*x[0]**2 - 364*x[0]*x[1]**2*x[2]**2*x[3] + 1456*x[0]*x[1]**2*x[2]**2 +
1000*x[0]*x[1]**2*x[2]*x[3] - 4000*x[0]*x[1]**2*x[2] - 636*x[0]*x[1]**2*x[3] + 2544*x[0]*x[1]**2 +
1144*x[0]*x[1]*x[2]**2*x[3] - 4576*x[0]*x[1]*x[2]**2 - 3096*x[0]*x[1]*x[2]*x[3] +
12384*x[0]*x[1]*x[2] + 1952*x[0]*x[1]*x[3] - 7808*x[0]*x[1] - 912*x[0]*x[2]**2*x[3] +
3648*x[0]*x[2]**2 + 2432*x[0]*x[2]*x[3] - 9728*x[0]*x[2] - 1520*x[0]*x[3] + 6080*x[0] +
544*x[1]**2*x[2]**2 - 1088*x[1]**2*x[2] + 544*x[1]**2 - 1616*x[1]*x[2]**2 + 3232*x[1]*x[2] -
1616*x[1] + 1216*x[2]**2 - 2432*x[2] + 1216))/(362880*(16*x[0] - 4*x[1] - 16*x[2] - 4*x[0]*x[3] +
4*x[1]*x[2] - 4*x[0]*x[1]*x[2] + x[0]*x[1]*x[2]*x[3] + 16)**2) +
(x[0]**5*(34*x[0]**6*x[1]**2*x[2]**2*x[3]**2 - 101*x[0]**6*x[1]**2*x[2]**2*x[3] +
76*x[0]**6*x[1]**2*x[2]**2 - 272*x[0]**6*x[1]*x[2]*x[3]**2 + 808*x[0]**6*x[1]*x[2]*x[3] -
608*x[0]**6*x[1]*x[2] + 544*x[0]**6*x[3]**2 - 1616*x[0]**6*x[3] + 1216*x[0]**6 -
159*x[0]**5*x[1]**2*x[2]**2*x[3]**2 + 488*x[0]**5*x[1]**2*x[2]**2*x[3] -
380*x[0]**5*x[1]**2*x[2]**2 + 159*x[0]**5*x[1]**2*x[2]*x[3]**2 - 488*x[0]**5*x[1]**2*x[2]*x[3] +
380*x[0]**5*x[1]**2*x[2] + 636*x[0]**5*x[1]*x[2]**2*x[3]**2 - 1952*x[0]**5*x[1]*x[2]**2*x[3] +
1520*x[0]**5*x[1]*x[2]**2 - 636*x[0]**5*x[1]*x[3]**2 + 1952*x[0]**5*x[1]*x[3] - 1520*x[0]**5*x[1] -
2544*x[0]**5*x[2]*x[3]**2 + 7808*x[0]**5*x[2]*x[3] - 6080*x[0]**5*x[2] + 2544*x[0]**5*x[3]**2 -
7808*x[0]**5*x[3] + 6080*x[0]**5 + 192*x[0]**4*x[1]**2*x[2]**2*x[3]**2 -
612*x[0]**4*x[1]**2*x[2]**2*x[3] + 496*x[0]**4*x[1]**2*x[2]**2 - 384*x[0]**4*x[1]**2*x[2]*x[3]**2
+ 1224*x[0]**4*x[1]**2*x[2]*x[3] - 992*x[0]**4*x[1]**2*x[2] + 192*x[0]**4*x[1]**2*x[3]**2 -
612*x[0]**4*x[1]**2*x[3] + 496*x[0]**4*x[1]**2 - 1536*x[0]**4*x[1]*x[2]**2*x[3]**2 +
4896*x[0]**4*x[1]*x[2]**2*x[3] - 3968*x[0]**4*x[1]*x[2]**2 + 3072*x[0]**4*x[1]*x[2]*x[3]**2 -
9792*x[0]**4*x[1]*x[2]*x[3] + 7936*x[0]**4*x[1]*x[2] - 1536*x[0]**4*x[1]*x[3]**2 +
4896*x[0]**4*x[1]*x[3] - 3968*x[0]**4*x[1] + 3072*x[0]**4*x[2]**2*x[3]**2 -
9792*x[0]**4*x[2]**2*x[3] + 7936*x[0]**4*x[2]**2 - 6144*x[0]**4*x[2]*x[3]**2 +
19584*x[0]**4*x[2]*x[3] - 15872*x[0]**4*x[2] + 3072*x[0]**4*x[3]**2 - 9792*x[0]**4*x[3] +
7936*x[0]**4 - 159*x[0]**3*x[1]**2*x[2]**2*x[3]**2 + 534*x[0]**3*x[1]**2*x[2]**2*x[3] -
456*x[0]**3*x[1]**2*x[2]**2 + 159*x[0]**3*x[1]**2*x[2]*x[3]**2 - 534*x[0]**3*x[1]**2*x[2]*x[3] +
456*x[0]**3*x[1]**2*x[2] + 318*x[0]**3*x[1]*x[2]**2*x[3]**2 - 1068*x[0]**3*x[1]*x[2]**2*x[3] +
912*x[0]**3*x[1]*x[2]**2 + 318*x[0]**3*x[1]*x[2]*x[3]**2 - 1068*x[0]**3*x[1]*x[2]*x[3] +
912*x[0]**3*x[1]*x[2] - 636*x[0]**3*x[1]*x[3]**2 + 2136*x[0]**3*x[1]*x[3] - 1824*x[0]**3*x[1] -
1272*x[0]**3*x[2]*x[3]**2 + 4272*x[0]**3*x[2]*x[3] - 3648*x[0]**3*x[2] + 1272*x[0]**3*x[3]**2 -
4272*x[0]**3*x[3] + 3648*x[0]**3 + 384*x[0]**2*x[1]**2*x[2]**2*x[3]**2 -
```

```
1356*x[0]**2*x[1]**2*x[2]**2*x[3] + 1224*x[0]**2*x[1]**2*x[2]**2 -
768*x[0]**2*x[1]**2*x[2]*x[3]**2 + 2712*x[0]**2*x[1]**2*x[2]*x[3] - 2448*x[0]**2*x[1]**2*x[2] +
384*x[0]**2*x[1]**2*x[3]**2 - 1356*x[0]**2*x[1]**2*x[3] + 1224*x[0]**2*x[1]**2 -
2304*x[0]**2*x[1]*x[2]**2*x[3]**2 + 8136*x[0]**2*x[1]*x[2]**2*x[3] - 7344*x[0]**2*x[1]*x[2]**2 +
4608*x[0]**2*x[1]*x[2]*x[3]**2 - 16272*x[0]**2*x[1]*x[2]*x[3] + 14688*x[0]**2*x[1]*x[2] -
2304*x[0]**2*x[1]*x[3]**2 + 8136*x[0]**2*x[1]*x[3] - 7344*x[0]**2*x[1] +
3072*x[0]**2*x[2]**2*x[3]**2 - 10848*x[0]**2*x[2]**2*x[3] + 9792*x[0]**2*x[2]**2 -
6144*x[0]**2*x[2]*x[3]**2 + 21696*x[0]**2*x[2]*x[3] - 19584*x[0]**2*x[2] + 3072*x[0]**2*x[3]**2 -
10848*x[0]**2*x[3] + 9792*x[0]**2 + 192*x[1]**2*x[2]**2*x[3]**2 - 744*x[1]**2*x[2]**2*x[3] +
768*x[1]**2*x[2]**2 - 384*x[1]**2*x[2]*x[3]**2 + 1488*x[1]**2*x[2]*x[3] - 1536*x[1]**2*x[2] +
192*x[1]**2*x[3]**2 - 744*x[1]**2*x[3] + 768*x[1]**2 - 768*x[1]*x[2]**2*x[3]**2 +
2976*x[1]*x[2]**2*x[3] - 3072*x[1]*x[2]**2 + 1536*x[1]*x[2]*x[3]**2 - 5952*x[1]*x[2]*x[3] +
6144*x[1]*x[2] - 768*x[1]*x[3]**2 + 2976*x[1]*x[3] - 3072*x[1] + 768*x[2]**2*x[3]**2 -
2976*x[2]**2*x[3] + 3072*x[2]**2 - 1536*x[2]*x[3]**2 + 5952*x[2]*x[3] - 6144*x[2] + 768*x[3]**2 -
2976*x[3] + 3072))/(362880*(16*x[0] - 4*x[1] - 16*x[2] - 4*x[0]*x[3] + 4*x[1]*x[2] - 4*x[0]*x[1]*x[2] +
x[0]*x[1]*x[2]*x[3] + 16)**2))*(16*x[0] - 4*x[1] - 16*x[2] - 4*x[0]*x[3] + 4*x[1]*x[2] - 4*x[0]*x[1]*x[2] +
x[0]*x[1]*x[2]*x[3] + 16)))**(1/2) for x in X])

# Generate samples

param_values_Cfrrr = saltelli.sample(problem_Cfrrr, 4096)

# Run model

Y_Cfrrr = evaluate_Cfrrr(param_values_Cfrrr)

# Generate the dimension of sample data

print(param_values_Cfrrr.shape, Y_Cfrrr.shape)

# Perform analysis

Si_Cfrrr = sobol.analyze(problem_Cfrrr, Y_Cfrrr, print_to_console=True)

# Print the first-order, second-order, total-order sensitivity indices

print()

# Generate sensitivity plot

Si_df_Cfrrr = Si_Cfrrr.to_df()

barplot(Si_df_Cfrrr[0])

plot.show()
```

```
(40960, 4) (40960,)
          ST   ST_conf
mu  0.262199  0.016207
r1  0.606906  0.020447
r2  0.195797  0.012946
r3  0.018003  0.002054
          S1   S1_conf
mu  0.187488  0.022584
r1  0.585542  0.029338
r2  0.145319  0.019381
r3  0.007185  0.006188
                S2   S2_conf
(mu, r1)  0.016683  0.032537
(mu, r2)  0.047791  0.028021
(mu, r3)  0.008010  0.026840
(r1, r2)  0.000445  0.044482
(r1, r3)  0.000261  0.045347
(r2, r3) -0.001168  0.025675
```



Appendix B

$C_{dB\_1} =$

(K1 K2 L^6 (-16 + r1 (9 - 5 r2) + 9 r2) - 2304 EI^2 (r1 + r2 + r1 r2) + 12 EI L^3 (8 K2 (-2 + r2) - 16 K1 (1 + r2) + K2 r1 (-16 + 9 r2) + K1 r1 (8 + 9 r2)))/(4 L^3 (K1 K2 L^3 (-4 + r1 r2) - 12 EI K1 (r1 + r2 + r1 r2) - 12 EI K2 (r1 + r2 + r1 r2)))

$C_{dB\_2} =$

(K1 K2 K3 L^9 μ^3 (4 (-7 + 4 r1) (-1 + r2) - μ (16 - 9 r2 + r1 (-9 + 5 r2)) (-4 + r3)) + 27648 EI^3 (-((1 + 2 r1) (-1 + r2) r3) + μ (r1 + r2 + r1 r2) (1 + 2 r3)) - 144 EI^2 L^3 (K2 ((8 + 7 r1) (-1 + r2) r3 + μ (8 (-2 + r2) + r1 (-16 + 9 r2)) (1 + 2 r3)) + K3 (64 μ^3 (1 + 2 r1) (-1 + r2) + 16 μ^4 (r1 + r2 + r1 r2) (-4 + r3) + (8 + 7 r1) (-1 + r2) r3 + 8 μ^2 (4 + 5 r1) (-1 + r2) (2 + r3) + μ (8 (-2 + r2) + r1 (-16 + 9 r2)) (1 + 2 r3)) + K1 (-((-32 + 17 r1) (-1 + r2) r3) + μ (-16 (1 + r2) + r1 (8 + 9 r2)) (1 + 2 r3))) + 12 EI L^6 (K2 K3 μ^3 (-4 (8 + 7 r1) (-1 + r2) + μ (8 (-2 + r2) + r1 (-16 + 9 r2)) (-4 + r3)) + K1 (K2 ((-7 + 4 r1) (-1 + r2) r3 + μ (16 - 9 r2 + r1 (-9 + 5 r2)) (1 + 2 r3)) + K3 (4 μ^3 (-32 + 17 r1) (-1 + r2) + μ^4 (-16 (1 + r2) + r1 (8 + 9 r2)) (-4 + r3) + (-7 + 4 r1) (-1 + r2) r3 + 2 μ^2 (-20 + 11 r1) (-1 + r2) (2 + r3) + μ (16 - 9 r2 + r1 (-9 + 5 r2)) (1 + 2 r3)))))/(4 K1 K2 K3 L^9 μ^3 (4 (-4 + r1) (-1 + r2) + μ (-4 + r1 r2) (-4 + r3)) + 576 EI^2 (K1 + K2 + K3) L^3 (-((1 + 2 r1) (-1 + r2) r3) + μ (r1 + r2 + r1 r2) (1 + 2 r3)) - 48 EI L^6 (K2 K3 μ^3 (-4 + 8 r1 (-1 + r2) + 4 r2 + μ r2 (-4 + r3) + μ r1 (1 + r2) (-4 + r3)) + K1 (K2 (-((-4 + r1) (-1 + r2) r3) + μ (-4 + r1 r2) (1 + 2 r3)) + K3 (4 μ^3 (1 + 2 r1) (-1 + r2) + μ^4 (r1 + r2 + r1 r2) (-4 + r3) - (-4 + r1) (-1 + r2) r3 + 2 μ^2 (2 + r1) (-1 + r2) (2 + r3) + μ (-4 + r1 r2) (1 + 2 r3)))))

$C_{fB\_1} =$

0.83368 ((L^3 (EI K2 r1 (12 + 12 r2) + EI K1 (r1 (12 + 12 r2) + 12. r2) + 12. EI K2 r2 + K1 K2 L^3 (4 - r1 r2)) (K1 K2 L^6 (2.66667 - 1.66667 r2 + r1 (-1.66667 + 1. r2)) + EI^2 (960. r2 + r1 (960. + 960. r2)) + EI L^3 (K2 (80. + 48. r1 - 32. r2 - 32. r1 r2) + K1 (80. - 32. r1 + 48. r2 - 32. r1 r2)))) / (EI^4 (779921. r2^2 + 779921. r1^2 (1. + 1. r2)^2 + r1 r2 (1.55984*10^6 + 1.55984*10^6 r2)) + K1^2 K2^2 L^12 (7.40299 - 9.13433 r2 + 2.86567 r2^2 + r1 (-9.13433 + 10.9851 r2 - 3.35821 r2^2) + r1^2 (2.86567 - 3.35821 r2 + 1. r2^2)) + EI K1 K2 L^9 (K2 (361.075 - 366.09 r2 + 94.0299 r2^2 + r1^2 (-156.358 + 189.851 r2 - 57.8507 r2^2) + r1 (3.58209 - 67.3433 r2 + 36.1791 r2^2)) + K1 (361.075 + 3.58209 r2 - 156.358 r2^2 + r1^2 (94.0299 + 36.1791 r2 - 57.8507 r2^2) + r1 (-366.09 - 67.3433 r2 + 189.851 r2^2))) + EI^2 L^6 (K1^2 (7221.49 + 7582.57 r2 + 2252.42 r2^2 + 962.866 r1^2 (1. + 1. r2)^2 + r1 (-5055.04 - 7978.03 r2 - 2922.99 r2^2)) + K2^2 (7221.49 - 5055.04 r2 + 962.866 r2^2 + r1^2 (2252.42 - 2922.99 r2 + 962.866 r2^2) + r1 (7582.57 - 7978.03 r2 + 1925.73 r2^2)) + K1 K2 (7221.49 + 6138.27 r2 - 5205.49 r2^2 + r1 (6138.27 - 2828.42 r2 - 1951.52 r2^2) + r1^2 (-5205.49 - 1951.52 r2 + 3253.97 r2^2))) + EI^3 L^3 (K2 ((129987. - 51994.7 r2) r2 + r1 (129987. + 155984. r2 - 103989. r2^2) + r1^2 (77992.1 + 25997.4 r2 - 51994.7 r2^2)) + K1 (-51994.7 r1^2 (1. + 1. r2)^2 + r2 (129987. + 77992.1 r2) + r1 (129987. + 155984. r2 + 25997.4 r2^2)))))^(1/2)

$C_{fB\_2} =$

$\sqrt{\int_0^L|y(x_1)| + \int_0^{\mu L}|y(x_2)|}$*(q^(1/2)*((0.010265982254684337511665003045737*L^4)/((0.000192901234567901234567901234567 9*L^9*q^2)/EI^2 + (0.00019290123456790123456790123456 79*L^9*μ^9*q^2)/EI^2 + (0.000062003968253968253968253968253968*L^7*q^2*(64.0*L^2*(8.0*k2*μ^4 - 8.0*k3*μ - 6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 - 32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 - 128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 + 8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 + 16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 - 2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 - 1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 + 12.0*k2*μ^2*r2*r3 - 32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3 + 6.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 - 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 + 4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3)^2 - 24.0*L^2*r1*(32.0*k2*μ + 32.0*k3*μ + 32.0*k2*r3 + 32.0*k3*r3 + 32.0*k1*μ^3 + 64.0*k2*μ^2 + 32.0*k2*μ^3 - 128.0*μ^3*r1 - 512.0*μ^3*r2 - 128.0*μ^4*r3 + 512.0*μ^3 + 512.0*μ^4 + 64.0*k1*μ^3*r1 + 32.0*k2*μ^2*r1 - 32.0*k1*μ^3*r2 + 32.0*k1*μ^4*r1 - 64.0*k2*μ^2*r2 + 64.0*k2*μ^3*r1 + 32.0*k1*μ^4*r2 + 32.0*k2*μ^2*r3 - 32.0*k2*μ^3*r2 + 32.0*k2*μ^4*r1 + 32.0*k2*μ^4*r2 + 128.0*μ^3*r1*r2 - 128.0*μ^4*r1*r2 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 - 32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1 + 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 + 2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 + 4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 + 4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 - 8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 + 8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 + 32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3 - 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 - 32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 - 8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 + 32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2 + 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 + 4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 + 2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 + 4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 - 4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 - 4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 - 16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 - 8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 + 4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 + 4.0*k2*k3*μ*r1*r2*r3)*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 - 24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 + 32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 + 2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 + 20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 + 8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 - 8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ + k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 -

2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 + 2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 +
2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 -
1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 + 4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 +
k1*k2*μ^2*r3 + k1*k2*μ^3*r2 - 1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 +
4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 + k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 +
8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 - 4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 -
1.0*k1*k2*μ^2*r2*r3 - 1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 -
1.0*k2*k3*μ^3*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 +
32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 +
32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 +
32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 +
16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3
- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 -
16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3
- 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3
+ 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 -
8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 +
2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) -
(0.00010850694444444444444444444444444444*L^4*q^2*(16.0*L^5*k1*(8.0*k2*μ^4 - 8.0*k3*μ -
6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 -
32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 -
128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 +
8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3
+ 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 +
16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 -
2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 -
1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 -
4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 +
k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 +
12.0*k2*μ^2*r2*r3 - 32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3
+ 6.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 -
8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 +
4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 +
2.0*k2*k3*μ^2*r2*r3 + 8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 -
4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3)^2 + 192.0*L^5*r1*(r1 -
1.0)*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 -
24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 +
32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 +
2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 +
20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 +
8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 -
8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ +

k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 - 2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 + 2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 + 2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 + 4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 + k1*k2*μ^2*r3 + k1*k2*μ^3*r2 - 1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 + 4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 + k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 + 8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 - 4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*μ^2*r2*r3 - 1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 - 1.0*k2*k3*μ^3*r2*r3)^2))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) + (0.000086805555555555555555555555555556*L^5*q^2*(144.0*L^4*r1^2*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 - 24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 + 32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 + 2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 + 20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 + 8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 - 8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ + k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 - 2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 + 2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 + 2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 + 4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 + k1*k2*μ^2*r3 + k1*k2*μ^3*r2 - 1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 + 4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 + k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 + 8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 - 4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*μ^2*r2*r3 - 1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 - 1.0*k2*k3*μ^3*r2*r3)^2 - 2.0*L^4*k1*(32.0*k2*μ + 32.0*k3*μ + 32.0*k2*r3 + 32.0*k3*r3 + 32.0*k1*μ^3 + 64.0*k2*μ^2 + 32.0*k2*μ^3 - 128.0*μ^3*r1 - 512.0*μ^3*r2 - 128.0*μ^4*r3 + 512.0*μ^3 + 512.0*μ^4 + 64.0*k1*μ^3*r1 + 32.0*k2*μ^2*r1 - 32.0*k1*μ^3*r2 + 32.0*k1*μ^4*r1 - 64.0*k2*μ^2*r2 + 64.0*k2*μ^3*r1 + 32.0*k1*μ^4*r2 + 32.0*k2*μ^2*r3 - 32.0*k2*μ^3*r2 + 32.0*k2*μ^4*r1 + 32.0*k2*μ^4*r2 + 128.0*μ^3*r1*r2 - 128.0*μ^4*r1*r2 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 - 32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1 + 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 + 2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 + 4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 +

4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 - 8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 +
8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 + 32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3
- 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 - 32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 -
8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 + 32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2
+ 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 + 4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 +
2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 + 4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 -
4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 - 4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 -
16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 - 8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 +
4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 + 4.0*k2*k3*μ*r1*r2*r3)*(8.0*k2*μ^4 - 8.0*k3*μ -
6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 -
32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 -
128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 +
8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3
+ 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 +
16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 -
2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 -
1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 -
4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 +
k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 +
12.0*k2*μ^2*r2*r3 - 32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3
+ 6.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 -
8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 +
4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 +
2.0*k2*k3*μ^2*r2*r3 + 8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 -
4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3) + 128.0*L^4*(r1 -
1.0)*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 -
24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 +
32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 +
2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 +
20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 +
8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 -
8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ +
k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 - 2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 +
2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 + 2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 -
1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 +
4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 + k1*k2*μ^2*r3 + k1*k2*μ^3*r2 -
1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 + 4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 +
k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 + 8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 -
4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*μ^2*r2*r3 -
1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 - 1.0*k2*k3*μ^3*r2*r3)*(8.0*k2*μ^4 - 8.0*k3*μ -
6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 -
32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 -
128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 +
8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3
+ 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 +
16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 -
2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 -

1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 + 12.0*k2*μ^2*r2*r3 - 32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3 + 6.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 - 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 + 4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) + (0.00014467592592592592592592592592593*L^3*μ^3*q^2*(64.0*L^6*μ^2*(r2 - 1.0)^2*(4.0*k2 + 4.0*k3 + k1*k2 + k1*k3 - 1.0*k2*k3 + 20.0*k2*μ - 2.0*k2*r1 - 2.0*k3*r1 + 8.0*k2*r3 + 8.0*k3*r3 + 16.0*k1*μ^3 - 16.0*k2*μ^2 - 16.0*k2*μ^3 + 16.0*k3*μ^2 + 4.0*k1*μ^5 + 20.0*k2*μ^4 + 4.0*k2*μ^5 - 32.0*μ^3*r1 - 16.0*μ^3*r3 - 16.0*μ^5*r1 - 32.0*μ^5*r3 + 64.0*μ^3 + 64.0*μ^5 - 1.0*k1*k2*μ^2 + k1*k3*μ^2 + k2*k3*μ^2 + 16.0*k1*μ^3*r1 + 4.0*k2*μ^2*r1 - 4.0*k3*μ^2*r1 - 4.0*k1*μ^3*r3 + 8.0*k1*μ^5*r1 + 10.0*k2*μ^4*r1 + 4.0*k2*μ^3*r3 + 8.0*k2*μ^5*r1 + 16.0*k3*μ^2*r3 - 2.0*k1*μ^5*r3 - 8.0*k2*μ^4*r3 - 2.0*k2*μ^5*r3 + 8.0*μ^3*r1*r3 + 8.0*μ^5*r1*r3 + 2.0*k1*k2*μ + 2.0*k2*k3*μ + k1*k2*r1 + k1*k3*r1 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 - 2.0*k2*k3*r3 - 8.0*k2*μ*r1 + 10.0*k2*μ*r3 - 4.0*k2*r1*r3 - 4.0*k3*r1*r3 + 4.0*k1*k2*μ*r1 + k1*k2*μ*r3 + k2*k3*μ*r1 + 4.0*k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 + 2.0*k1*k3*r1*r3 - 4.0*k2*μ*r1*r3 - 2.0*k1*k2*μ^2*r1 + 2.0*k1*k3*μ^2*r1 + 2.0*k2*k3*μ^2*r1 + k1*k3*μ^2*r3 + k2*k3*μ^2*r3 - 4.0*k1*μ^3*r1*r3 - 4.0*k3*μ^2*r1*r3 - 4.0*k1*μ^5*r1*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^5*r1*r3 + 2.0*k1*k2*μ*r1*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k1*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r1*r3)^2 + 24.0*L^6*k2*(16.0*k2*μ^2 - 4.0*k3*r3 - 16.0*k1*μ^3 - 4.0*k2*r3 - 16.0*k2*μ^4 + 16.0*k3*μ^3 + 32.0*μ^3*r1 + 64.0*μ^3*r2 - 32.0*μ^6*r3 - 64.0*μ^3 + 64.0*μ^6 - 16.0*k1*μ^3*r1 - 4.0*k2*μ^2*r1 + 16.0*k1*μ^3*r2 + 4.0*k2*μ^2*r2 + 8.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 - 8.0*k2*μ^4*r1 + 4.0*k1*μ^6*r1 + 16.0*k2*μ^4*r2 + 4.0*k1*μ^6*r2 + 4.0*k2*μ^4*r3 + 4.0*k2*μ^6*r1 + 16.0*k3*μ^3*r3 + 4.0*k2*μ^6*r2 - 32.0*μ^3*r1*r2 - 16.0*μ^6*r1*r2 - 1.0*k1*k2*r3 - 1.0*k1*k3*r3 + k2*k3*r3 + 2.0*k2*r1*r3 + 4.0*k2*r2*r3 + 2.0*k3*r1*r3 + 4.0*k3*r2*r3 - 2.0*k2*k3*μ*r3 - 1.0*k1*k2*r1*r3 + k1*k2*r2*r3 - 1.0*k1*k3*r1*r3 + k1*k3*r2*r3 - 1.0*k2*k3*r2*r3 - 2.0*k2*r1*r2*r3 - 2.0*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1 + 2.0*k1*k2*μ^2*r2 - 1.0*k1*k2*μ^3*r1 - 1.0*k1*k2*μ^3*r2 + k1*k3*μ^3*r1 + k1*k3*μ^3*r2 + k2*k3*μ^3*r1 + k2*k3*μ^3*r2 + 16.0*k1*μ^3*r1*r2 - 4.0*k2*μ^2*r1*r2 -

2.0*k2*μ^2*r1*r3 + 4.0*k2*μ^3*r1*r2 + 2.0*k2*μ^2*r2*r3 + 8.0*k2*μ^4*r1*r2 - 4.0*k3*μ^3*r1*r2 + 4.0*k1*μ^6*r1*r2 + 2.0*k2*μ^4*r1*r3 - 2.0*k1*μ^6*r1*r3 - 4.0*k2*μ^4*r2*r3 + 4.0*k2*μ^6*r1*r2 - 2.0*k1*μ^6*r2*r3 - 2.0*k2*μ^6*r1*r3 - 2.0*k2*μ^6*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 + k1*k2*r1*r2*r3 + k1*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1*r2 + k1*k2*μ^2*r1*r3 - 1.0*k1*k2*μ^3*r1*r2 + k1*k2*μ^2*r2*r3 + k1*k3*μ^3*r1*r2 + k1*k3*μ^3*r1*r3 + k2*k3*μ^3*r1*r2 + k1*k3*μ^3*r2*r3 + k2*k3*μ^3*r1*r3 + k2*k3*μ^3*r2*r3 - 2.0*k2*μ^2*r1*r2*r3 - 2.0*k2*μ^4*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 - 2.0*k1*μ^6*r1*r2*r3 - 2.0*k2*μ^6*r1*r2*r3 + k1*k2*μ^2*r1*r2*r3 + k1*k3*μ^3*r1*r2*r3 + k2*k3*μ^3*r1*r2*r3 + k2*k3*μ*r1*r2*r3)*(8.0*k1*μ^4 - 10.0*k3*r3 - 8.0*k1*μ^2 - 16.0*k1*μ^3 - 8.0*k3*μ + 16.0*k3*μ^2 + 8.0*k3*μ^3 + 16.0*μ^2*r1 + 32.0*μ^2*r2 + 64.0*μ^3*r1 - 16.0*μ^2*r3 + 160.0*μ^3*r2 - 160.0*μ^4*r2 + 16.0*μ^6*r1 - 64.0*μ^5*r3 - 32.0*μ^6*r2 - 16.0*μ^6*r3 - 32.0*μ^2 - 160.0*μ^3 + 160.0*μ^5 + 32.0*μ^6 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 - 32.0*k1*μ^3*r1 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^3*r2 - 24.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 + 4.0*k3*μ^3*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 + 32.0*k3*μ^2*r3 - 8.0*k3*μ^3*r2 + 8.0*k3*μ^3*r3 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 - 64.0*μ^3*r1*r2 + 16.0*μ^2*r2*r3 + 64.0*μ^4*r1*r2 - 40.0*μ^5*r1*r2 + 40.0*μ^4*r2*r3 - 16.0*μ^6*r1*r2 - 8.0*μ^6*r1*r3 + 16.0*μ^6*r2*r3 - 1.0*k1*k3*r3 + 2.0*k3*μ*r1 - 2.0*k3*μ*r2 + 4.0*k3*r1*r3 + 10.0*k3*r2*r3 - 1.0*k1*k3*μ*r1 - 1.0*k1*k3*μ*r2 + k1*k3*μ*r3 - 2.0*k1*k3*r1*r3 + k1*k3*r2*r3 + 2.0*k3*μ*r1*r2 - 20.0*k3*μ*r2*r3 - 4.0*k3*r1*r2*r3 + k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 32.0*k1*μ^3*r1*r2 + 4.0*k1*μ^2*r2*r3 - 32.0*k1*μ^4*r1*r2 - 4.0*k3*μ^2*r1*r2 + 10.0*k1*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r2 + 6.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 + 4.0*k3*μ^3*r1*r3 - 4.0*k1*μ^5*r2*r3 - 8.0*k3*μ^3*r2*r3 - 8.0*μ^2*r1*r2*r3 - 16.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k1*k3*μ*r1*r2 - 3.0*k1*k3*μ*r2*r3 + 2.0*k1*k3*r1*r2*r3 + 8.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + 2.0*k1*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 8.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 - 4.0*k1*k3*μ*r1*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) - (0.0023148148148148148148148148148148*L^9*q^2*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 - 24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 + 32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 + 2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 + 20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 + 8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 +

10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 - 8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 -
8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ + k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 -
2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 + 2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 +
2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 -
1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 + 4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 +
k1*k2*μ^2*r3 + k1*k2*μ^3*r2 - 1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 +
4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 + k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 +
8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 - 4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 -
1.0*k1*k2*μ^2*r2*r3 - 1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 -
1.0*k2*k3*μ^3*r2*r3)*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 +
32.0*k2*μ^2 + 16.0*k2*μ^3 - 896.0*μ^3*r1 - 256.0*μ^3*r2 - 1024.0*μ^4*r1 - 64.0*μ^4*r3 -
192.0*μ^6*r1 + 256.0*μ^3 + 256.0*μ^4 + 64.0*μ^3*r1^2 - 192.0*μ^4*r1^2 - 96.0*μ^6*r1^2 +
16.0*k1*μ^3*r1 - 12.0*k2*μ*r1^2 - 160.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 -
32.0*k2*μ^2*r2 - 32.0*k2*μ^3*r1 - 12.0*k3*μ*r1^2 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 -
16.0*k2*μ^3*r2 + 64.0*k2*μ^4*r1 - 48.0*k3*μ^3*r1 + 16.0*k2*μ^4*r2 + 4.0*k2*r1^2*r3 +
4.0*k3*r1^2*r3 + 896.0*μ^3*r1*r2 + 128.0*μ^4*r1*r2 + 256.0*μ^4*r1*r3 + 192.0*μ^6*r1*r2 +
96.0*μ^6*r1*r3 - 32.0*k1*μ^3*r1^2 - 112.0*k2*μ^2*r1^2 - 16.0*k1*μ^4*r1^2 - 200.0*k2*μ^3*r1^2 -
16.0*k2*μ^4*r1^2 - 24.0*k3*μ^3*r1^2 + 60.0*k2*μ^5*r1^2 - 64.0*μ^3*r1^2*r2 + 256.0*μ^4*r1^2*r2
+ 48.0*μ^4*r1^2*r3 + 96.0*μ^6*r1^2*r2 + 48.0*μ^6*r1^2*r3 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 -
64.0*k2*μ*r1 - 64.0*k3*μ*r1 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 56.0*k2*r1*r3 - 16.0*k2*r2*r3 -
56.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1
+ k2*k3*μ*r2 + k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 - 5.0*k2*k3*r1*r3 -
1.0*k2*k3*r2*r3 + 8.0*k2*μ*r1*r2 - 128.0*k2*μ*r1*r3 + 8.0*k3*μ*r1*r2 - 128.0*k3*μ*r1*r3 +
56.0*k2*r1*r2*r3 + 56.0*k3*r1*r2*r3 - 1.0*k1*k2*μ*r1^2 - 1.0*k1*k3*μ*r1^2 - 7.0*k2*k3*μ*r1^2 -
2.0*k1*k2*r1^2*r3 - 2.0*k1*k3*r1^2*r3 - 14.0*k2*k3*r1^2*r3 - 16.0*k1*μ^3*r1*r2 +
16.0*k2*μ*r1^2*r2 + 160.0*k2*μ^2*r1*r2 - 24.0*k2*μ*r1^2*r3 - 80.0*k2*μ^2*r1*r3 +
32.0*k2*μ^3*r1*r2 + 16.0*k3*μ*r1^2*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 -
144.0*k2*μ^4*r1*r2 - 24.0*k3*μ*r1^2*r3 + 48.0*k3*μ^3*r1*r2 - 4.0*k1*μ^4*r2*r3 -
16.0*k2*μ^4*r1*r3 + 60.0*k2*μ^5*r1*r2 - 48.0*k3*μ^3*r1*r3 - 4.0*k2*μ^4*r2*r3 - 4.0*k2*r1^2*r2*r3
- 4.0*k3*r1^2*r2*r3 - 32.0*μ^4*r1*r2*r3 - 96.0*μ^6*r1*r2*r3 + 6.0*k2*k3*μ^2*r1^2 +
32.0*k1*μ^3*r1^2*r2 + 112.0*k2*μ^2*r1^2*r2 - 16.0*k1*μ^4*r1^2*r2 - 56.0*k2*μ^2*r1^2*r3 +
200.0*k2*μ^3*r1^2*r2 + 4.0*k1*μ^4*r1^2*r3 - 208.0*k2*μ^4*r1^2*r2 + 24.0*k3*μ^3*r1^2*r2 +
4.0*k2*μ^4*r1^2*r3 + 60.0*k2*μ^5*r1^2*r2 - 24.0*k3*μ^3*r1^2*r3 - 24.0*k2*μ^5*r1^2*r3 -
64.0*μ^4*r1^2*r2*r3 - 48.0*μ^6*r1^2*r2*r3 + 6.0*k2*k3*μ^2*r1^2*r2 + 12.0*k2*k3*μ^2*r1^2*r3 +
56.0*k2*μ^2*r1^2*r2*r3 + 4.0*k1*μ^4*r1^2*r2*r3 + 52.0*k2*μ^4*r1^2*r2*r3 +
24.0*k3*μ^3*r1^2*r2*r3 - 24.0*k2*μ^5*r1^2*r2*r3 + 2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 - 6.0*k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 8.0*k2*k3*μ*r1*r3 +
2.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*r1*r2*r3 - 1.0*k1*k3*r1*r2*r3 + 5.0*k2*k3*r1*r2*r3 +
16.0*k2*μ*r1*r2*r3 + 16.0*k3*μ*r1*r2*r3 - 1.0*k1*k2*μ*r1^2*r2 - 2.0*k1*k2*μ*r1^2*r3 -
1.0*k1*k3*μ*r1^2*r2 - 2.0*k1*k3*μ*r1^2*r3 - 7.0*k2*k3*μ*r1^2*r2 + 6.0*k2*k3*μ^2*r1*r2 -
2.0*k2*k3*μ*r1^2*r3 + 2.0*k1*k2*r1^2*r2*r3 + 2.0*k1*k3*r1^2*r2*r3 + 14.0*k2*k3*r1^2*r2*r3 +
32.0*k2*μ*r1^2*r2*r3 + 80.0*k2*μ^2*r1*r2*r3 + 32.0*k3*μ*r1^2*r2*r3 + 36.0*k2*μ^4*r1*r2*r3 +
48.0*k3*μ^3*r1*r2*r3 - 24.0*k2*μ^5*r1*r2*r3 - 2.0*k1*k2*μ*r1^2*r2*r3 - 2.0*k1*k3*μ*r1^2*r2*r3 -
26.0*k2*k3*μ*r1^2*r2*r3 + 12.0*k2*k3*μ^2*r1*r2*r3 + 12.0*k2*k3*μ^2*r1^2*r2*r3 -
18.0*k2*k3*μ*r1*r2*r3))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 +
32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 +
32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 +

32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 +
16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3
- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 -
16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3
- 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3
+ 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 -
8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 +
2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) -
(0.00010850694444444444444444444444444444*L^4*μ^4*q^2*(16.0*L^5*k2*(8.0*k1*μ^4 - 10.0*k3*r3 -
8.0*k1*μ^2 - 16.0*k1*μ^3 - 8.0*k3*μ + 16.0*k3*μ^2 + 8.0*k3*μ^3 + 16.0*μ^2*r1 + 32.0*μ^2*r2 +
64.0*μ^3*r1 - 16.0*μ^2*r3 + 160.0*μ^3*r2 - 160.0*μ^4*r2 + 16.0*μ^6*r1 - 64.0*μ^5*r3 - 32.0*μ^6*r2
- 16.0*μ^6*r3 - 32.0*μ^2 - 160.0*μ^3 + 160.0*μ^5 + 32.0*μ^6 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 -
32.0*k1*μ^3*r1 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^3*r2 - 24.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 +
4.0*k3*μ^3*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 + 32.0*k3*μ^2*r3 - 8.0*k3*μ^3*r2 + 8.0*k3*μ^3*r3
- 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 - 64.0*μ^3*r1*r2 + 16.0*μ^2*r2*r3 + 64.0*μ^4*r1*r2 -
40.0*μ^5*r1*r2 + 40.0*μ^4*r2*r3 - 16.0*μ^6*r1*r2 - 8.0*μ^6*r1*r3 + 16.0*μ^6*r2*r3 - 1.0*k1*k3*r3
+ 2.0*k3*μ*r1 - 2.0*k3*μ*r2 + 4.0*k3*r1*r3 + 10.0*k3*r2*r3 - 1.0*k1*k3*μ*r1 - 1.0*k1*k3*μ*r2 +
k1*k3*μ*r3 - 2.0*k1*k3*r1*r3 + k1*k3*r2*r3 + 2.0*k3*μ*r1*r2 - 20.0*k3*μ*r2*r3 - 4.0*k3*r1*r2*r3 +
k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 32.0*k1*μ^3*r1*r2 +
4.0*k1*μ^2*r2*r3 - 32.0*k1*μ^4*r1*r2 - 4.0*k3*μ^2*r1*r2 + 10.0*k1*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r2 +
6.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 + 4.0*k3*μ^3*r1*r3 - 4.0*k1*μ^5*r2*r3 - 8.0*k3*μ^3*r2*r3 -
8.0*μ^2*r1*r2*r3 - 16.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k1*k3*μ*r1*r2 -
3.0*k1*k3*μ*r2*r3 + 2.0*k1*k3*r1*r2*r3 + 8.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 +
2.0*k1*k3*μ^2*r1*r3 + 2.0*k1*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 8.0*k1*μ^4*r1*r2*r3 -
8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 -
4.0*k1*k3*μ*r1*r2*r3)*(8.0*k2*μ + 10.0*k2*r3 - 8.0*k1*μ^2 + 8.0*k2*μ^2 + 8.0*k1*μ^4 +
8.0*k2*μ^3 + 16.0*k3*μ^2 + 8.0*k2*μ^4 + 16.0*μ^2*r1 + 32.0*μ^2*r2 - 16.0*μ^2*r3 - 32.0*μ^4*r1 -
128.0*μ^4*r2 - 32.0*μ^4*r3 - 64.0*μ^5*r3 - 32.0*μ^2 + 128.0*μ^4 + 160.0*μ^5 - 8.0*k1*μ^2*r1 +
8.0*k1*μ^2*r2 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^4*r1 - 8.0*k2*μ^2*r2 + 4.0*k2*μ^3*r1 - 8.0*k1*μ^4*r2 +
10.0*k1*μ^5*r1 + 4.0*k2*μ^2*r3 - 8.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 - 2.0*k1*μ^4*r3 +
10.0*k1*μ^5*r2 - 8.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 32.0*k3*μ^2*r3 - 2.0*k2*μ^4*r3 +
10.0*k2*μ^5*r2 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 + 16.0*μ^2*r2*r3 + 32.0*μ^4*r1*r2 +
8.0*μ^4*r1*r3 - 40.0*μ^5*r1*r2 + 32.0*μ^4*r2*r3 + k1*k2*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 +
16.0*k2*μ*r3 + 16.0*k3*μ*r3 - 4.0*k2*r1*r3 - 10.0*k2*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 +
k1*k3*μ*r3 + k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 - 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 +
4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 - 16.0*k3*μ*r2*r3 + 4.0*k2*r1*r2*r3 + k1*k3*μ^2*r1 +
k1*k3*μ^2*r2 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 +
4.0*k1*μ^2*r2*r3 - 16.0*k1*μ^4*r1*r2 - 4.0*k2*μ^3*r1*r2 - 4.0*k3*μ^2*r1*r2 - 4.0*k1*μ^4*r1*r3 +
10.0*k1*μ^5*r1*r2 - 4.0*k2*μ^2*r2*r3 - 16.0*k2*μ^4*r1*r2 + 2.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 -
4.0*k2*μ^4*r1*r3 + 10.0*k2*μ^5*r1*r2 - 4.0*k1*μ^5*r2*r3 + 2.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 -
4.0*k2*μ^5*r2*r3 - 8.0*μ^2*r1*r2*r3 - 8.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + k1*k2*μ*r1*r2 +
2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 - 1.0*k1*k3*μ*r2*r3 +

2.0*k2*k3*μ*r1*r3 - 1.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + k2*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r2*r3 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 4.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 + 4.0*k2*μ^4*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 - 2.0*k1*k3*μ*r1*r2*r3 - 2.0*k2*k3*μ*r1*r2*r3) + 192.0*L^5*μ*(r2 - 1.0)*(4.0*k2 + 4.0*k3 + k1*k2 + k1*k3 - 1.0*k2*k3 + 20.0*k2*μ - 2.0*k2*r1 - 2.0*k3*r1 + 8.0*k2*r3 + 8.0*k3*r3 + 16.0*k1*μ^3 - 16.0*k2*μ^2 - 16.0*k2*μ^3 + 16.0*k3*μ^2 + 4.0*k1*μ^5 + 20.0*k2*μ^4 + 4.0*k2*μ^5 - 32.0*μ^3*r1 - 16.0*μ^3*r3 - 16.0*μ^5*r1 - 32.0*μ^5*r3 + 64.0*μ^3 + 64.0*μ^5 - 1.0*k1*k2*μ^2 + k1*k3*μ^2 + k2*k3*μ^2 + 16.0*k1*μ^3*r1 + 4.0*k2*μ^2*r1 - 4.0*k3*μ^2*r1 - 4.0*k1*μ^3*r3 + 8.0*k1*μ^5*r1 + 10.0*k2*μ^4*r1 + 4.0*k2*μ^3*r3 + 8.0*k2*μ^5*r1 + 16.0*k3*μ^2*r3 - 2.0*k1*μ^5*r3 - 8.0*k2*μ^4*r3 - 2.0*k2*μ^5*r3 + 8.0*μ^3*r1*r3 + 8.0*μ^5*r1*r3 + 2.0*k1*k2*μ + 2.0*k2*k3*μ + k1*k2*r1 + k1*k3*r1 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 - 2.0*k2*k3*r3 - 8.0*k2*μ*r1 + 10.0*k2*μ*r3 - 4.0*k2*r1*r3 - 4.0*k3*r1*r3 + 4.0*k1*k2*μ*r1 + k1*k2*μ*r3 + k2*k3*μ*r1 + 4.0*k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 + 2.0*k1*k3*r1*r3 - 4.0*k2*μ*r1*r3 - 2.0*k1*k2*μ^2*r1 + 2.0*k1*k3*μ^2*r1 + 2.0*k2*k3*μ^2*r1 + k1*k3*μ^2*r3 + k2*k3*μ^2*r3 - 4.0*k1*μ^3*r1*r3 - 4.0*k3*μ^2*r1*r3 - 4.0*k1*μ^5*r1*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^5*r1*r3 + 2.0*k1*k2*μ*r1*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k1*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r1*r3)*(16.0*k2*μ^2 - 4.0*k3*r3 - 16.0*k1*μ^3 - 4.0*k2*r3 - 16.0*k2*μ^4 + 16.0*k3*μ^3 + 32.0*μ^3*r1 + 64.0*μ^3*r2 - 32.0*μ^6*r3 - 64.0*μ^3 + 64.0*μ^6 - 16.0*k1*μ^3*r1 - 4.0*k2*μ^2*r1 + 16.0*k1*μ^3*r2 + 4.0*k2*μ^2*r2 + 8.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 - 8.0*k2*μ^4*r1 + 4.0*k1*μ^6*r1 + 16.0*k2*μ^4*r2 + 4.0*k1*μ^6*r2 + 4.0*k2*μ^4*r3 + 4.0*k2*μ^6*r1 + 16.0*k3*μ^3*r3 + 4.0*k2*μ^6*r2 - 32.0*μ^3*r1*r2 - 16.0*μ^6*r1*r2 - 1.0*k1*k2*r3 - 1.0*k1*k3*r3 + k2*k3*r3 + 2.0*k2*r1*r3 + 4.0*k2*r2*r3 + 2.0*k3*r1*r3 + 4.0*k3*r2*r3 - 2.0*k2*k3*μ*r3 - 1.0*k1*k2*r1*r3 + k1*k2*r2*r3 - 1.0*k1*k3*r1*r3 + k1*k3*r2*r3 - 1.0*k2*k3*r2*r3 - 2.0*k2*r1*r2*r3 - 2.0*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1 + 2.0*k1*k2*μ^2*r2 - 1.0*k1*k2*μ^3*r1 - 1.0*k1*k2*μ^3*r2 + k1*k3*μ^3*r1 + k1*k3*μ^3*r2 + k2*k3*μ^3*r1 + k2*k3*μ^3*r2 + 16.0*k1*μ^3*r1*r2 - 4.0*k2*μ^2*r1*r2 - 2.0*k2*μ^2*r1*r3 + 4.0*k2*μ^3*r1*r2 + 2.0*k2*μ^2*r2*r3 + 8.0*k2*μ^4*r1*r2 - 4.0*k3*μ^3*r1*r2 + 4.0*k1*μ^6*r1*r2 + 2.0*k2*μ^4*r1*r3 - 2.0*k1*μ^6*r1*r3 - 4.0*k2*μ^4*r2*r3 + 4.0*k2*μ^6*r1*r2 - 2.0*k1*μ^6*r2*r3 - 2.0*k2*μ^6*r1*r3 - 2.0*k2*μ^6*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 + k1*k2*r1*r2*r3 + k1*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1*r2 + k1*k2*μ^2*r1*r3 - 1.0*k1*k2*μ^3*r1*r2 + k1*k2*μ^2*r2*r3 + k1*k3*μ^3*r1*r2 + k1*k3*μ^3*r1*r3 + k2*k3*μ^3*r1*r2 + k1*k3*μ^3*r2*r3 + k2*k3*μ^3*r1*r3 + k2*k3*μ^3*r2*r3 - 2.0*k2*μ^2*r1*r2*r3 - 2.0*k2*μ^4*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 - 2.0*k1*μ^6*r1*r2*r3 - 2.0*k2*μ^6*r1*r2*r3 + k1*k2*μ^2*r1*r2*r3 + k1*k3*μ^3*r1*r2*r3 + k2*k3*μ^3*r1*r2*r3 + k2*k3*μ*r1*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +

2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) - (0.000072337962962962962962962962963*L^6*μ^6*q^2*(192.0*L^3*(16.0*k2*μ^2 - 4.0*k3*r3 - 16.0*k1*μ^3 - 4.0*k2*r3 - 16.0*k2*μ^4 + 16.0*k3*μ^3 + 32.0*μ^3*r1 + 64.0*μ^3*r2 - 32.0*μ^6*r3 - 64.0*μ^3 + 64.0*μ^6 - 16.0*k1*μ^3*r1 - 4.0*k2*μ^2*r1 + 16.0*k1*μ^3*r2 + 4.0*k2*μ^2*r2 + 8.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 - 8.0*k2*μ^4*r1 + 4.0*k1*μ^6*r1 + 16.0*k2*μ^4*r2 + 4.0*k1*μ^6*r2 + 4.0*k2*μ^4*r3 + 4.0*k2*μ^6*r1 + 16.0*k3*μ^3*r3 + 4.0*k2*μ^6*r2 - 32.0*μ^3*r1*r2 - 16.0*μ^6*r1*r2 - 1.0*k1*k2*r3 - 1.0*k1*k3*r3 + k2*k3*r3 + 2.0*k2*r1*r3 + 4.0*k2*r2*r3 + 2.0*k3*r1*r3 + 4.0*k3*r2*r3 - 2.0*k2*k3*μ*r3 - 1.0*k1*k2*r1*r3 + k1*k2*r2*r3 - 1.0*k1*k3*r1*r3 + k1*k3*r2*r3 - 1.0*k2*k3*r2*r3 - 2.0*k2*r1*r2*r3 - 2.0*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1 + 2.0*k1*k2*μ^2*r2 - 1.0*k1*k2*μ^3*r1 - 1.0*k1*k2*μ^3*r2 + k1*k3*μ^3*r1 + k1*k3*μ^3*r2 + k2*k3*μ^3*r1 + k2*k3*μ^3*r2 + 16.0*k1*μ^3*r1*r2 - 4.0*k2*μ^2*r1*r2 - 2.0*k2*μ^2*r1*r3 + 4.0*k2*μ^3*r1*r2 + 2.0*k2*μ^2*r2*r3 + 8.0*k2*μ^4*r1*r2 - 4.0*k3*μ^3*r1*r2 + 4.0*k1*μ^6*r1*r2 + 2.0*k2*μ^4*r1*r3 - 2.0*k1*μ^6*r1*r3 - 4.0*k2*μ^4*r2*r3 + 4.0*k2*μ^6*r1*r2 - 2.0*k1*μ^6*r2*r3 - 2.0*k2*μ^6*r1*r3 - 2.0*k2*μ^6*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 + k1*k2*r1*r2*r3 + k1*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1*r2 + k1*k2*μ^2*r1*r3 - 1.0*k1*k2*μ^3*r1*r2 + k1*k2*μ^2*r2*r3 + k1*k3*μ^3*r1*r2 + k1*k3*μ^3*r1*r3 + k2*k3*μ^3*r1*r2 + k1*k3*μ^3*r2*r3 + k2*k3*μ^3*r1*r3 + k2*k3*μ^3*r2*r3 - 2.0*k2*μ^2*r1*r2*r3 - 2.0*k2*μ^4*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 - 2.0*k1*μ^6*r1*r2*r3 - 2.0*k2*μ^6*r1*r2*r3 + k1*k2*μ^2*r1*r2*r3 + k1*k3*μ^3*r1*r2*r3 + k2*k3*μ^3*r1*r2*r3 + k2*k3*μ*r1*r2*r3)*(8.0*k2*μ + 10.0*k2*r3 - 8.0*k1*μ^2 + 8.0*k2*μ^2 + 8.0*k1*μ^4 + 8.0*k2*μ^3 + 16.0*k3*μ^2 + 8.0*k2*μ^4 + 16.0*μ^2*r1 + 32.0*μ^2*r2 - 16.0*μ^2*r3 - 32.0*μ^4*r1 - 128.0*μ^4*r2 - 32.0*μ^4*r3 - 64.0*μ^5*r3 - 32.0*μ^2 + 128.0*μ^4 + 160.0*μ^5 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^4*r1 - 8.0*k2*μ^2*r2 + 4.0*k2*μ^3*r1 - 8.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 + 4.0*k2*μ^2*r3 - 8.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 - 8.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 32.0*k3*μ^2*r3 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 + 16.0*μ^2*r2*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 40.0*μ^5*r1*r2 + 32.0*μ^4*r2*r3 + k1*k2*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 + 16.0*k2*μ*r3 + 16.0*k3*μ*r3 - 4.0*k2*r1*r3 - 10.0*k2*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r3 + k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 - 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 - 16.0*k3*μ*r2*r3 + 4.0*k2*r1*r2*r3 + k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 4.0*k1*μ^2*r2*r3 - 16.0*k1*μ^4*r1*r2 - 4.0*k2*μ^3*r1*r2 - 4.0*k3*μ^2*r1*r2 - 4.0*k1*μ^4*r1*r3 + 10.0*k1*μ^5*r1*r2 - 4.0*k2*μ^2*r2*r3 - 16.0*k2*μ^4*r1*r2 + 2.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 - 4.0*k2*μ^4*r1*r3 + 10.0*k2*μ^5*r1*r2 - 4.0*k1*μ^5*r2*r3 + 2.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^2*r1*r2*r3 - 8.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 - 1.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 - 1.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + k2*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r2*r3 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 4.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 + 4.0*k2*μ^4*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 - 2.0*k1*k3*μ*r1*r2*r3 - 2.0*k2*k3*μ*r1*r2*r3) + 16.0*L^3*μ*(r2 - 1.0)*(4.0*k2 + 4.0*k3 + k1*k2 + k1*k3 - 1.0*k2*k3 + 20.0*k2*μ - 2.0*k2*r1 - 2.0*k3*r1 + 8.0*k2*r3 + 8.0*k3*r3 + 16.0*k1*μ^3 - 16.0*k2*μ^2 - 16.0*k2*μ^3 + 16.0*k3*μ^2 + 4.0*k1*μ^5 + 20.0*k2*μ^4 + 4.0*k2*μ^5 - 32.0*μ^3*r1 - 16.0*μ^3*r3 -

16.0*µ^5*r1 - 32.0*µ^5*r3 + 64.0*µ^3 + 64.0*µ^5 - 1.0*k1*k2*µ^2 + k1*k3*µ^2 + k2*k3*µ^2 +
16.0*k1*µ^3*r1 + 4.0*k2*µ^2*r1 - 4.0*k3*µ^2*r1 - 4.0*k1*µ^3*r3 + 8.0*k1*µ^5*r1 + 10.0*k2*µ^4*r1
+ 4.0*k2*µ^3*r3 + 8.0*k2*µ^5*r1 + 16.0*k3*µ^2*r3 - 2.0*k1*µ^5*r3 - 8.0*k2*µ^4*r3 - 2.0*k2*µ^5*r3
+ 8.0*µ^3*r1*r3 + 8.0*µ^5*r1*r3 + 2.0*k1*k2*µ + 2.0*k2*k3*µ + k1*k2*r1 + k1*k3*r1 +
2.0*k1*k2*r3 + 2.0*k1*k3*r3 - 2.0*k2*k3*r3 - 8.0*k2*µ*r1 + 10.0*k2*µ*r3 - 4.0*k2*r1*r3 -
4.0*k3*r1*r3 + 4.0*k1*k2*µ*r1 + k1*k2*µ*r3 + k2*k3*µ*r1 + 4.0*k2*k3*µ*r3 + 2.0*k1*k2*r1*r3 +
2.0*k1*k3*r1*r3 - 4.0*k2*µ*r1*r3 - 2.0*k1*k2*µ^2*r1 + 2.0*k1*k3*µ^2*r1 + 2.0*k2*k3*µ^2*r1 +
k1*k3*µ^2*r3 + k2*k3*µ^2*r3 - 4.0*k1*µ^3*r1*r3 - 4.0*k3*µ^2*r1*r3 - 4.0*k1*µ^5*r1*r3 -
4.0*k2*µ^4*r1*r3 - 4.0*k2*µ^5*r1*r3 + 2.0*k1*k2*µ*r1*r3 + 2.0*k2*k3*µ*r1*r3 +
2.0*k1*k3*µ^2*r1*r3 + 2.0*k2*k3*µ^2*r1*r3)*(32.0*k2*µ + 32.0*k3*µ + 32.0*k2*r3 + 32.0*k3*r3 +
32.0*k1*µ^3 + 64.0*k2*µ^2 + 32.0*k2*µ^3 - 128.0*µ^3*r1 - 512.0*µ^3*r2 - 128.0*µ^4*r3 + 512.0*µ^3
+ 512.0*µ^4 + 64.0*k1*µ^3*r1 + 32.0*k2*µ^2*r1 - 32.0*k1*µ^3*r2 + 32.0*k1*µ^4*r1 -
64.0*k2*µ^2*r2 + 64.0*k2*µ^3*r1 + 32.0*k1*µ^4*r2 + 32.0*k2*µ^2*r3 - 32.0*k2*µ^3*r2 +
32.0*k2*µ^4*r1 + 32.0*k2*µ^4*r2 + 128.0*µ^3*r1*r2 - 128.0*µ^4*r1*r2 + 2.0*k1*k2*r3 +
2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*µ*r3 + 64.0*k3*µ*r3 - 8.0*k2*r1*r3 - 32.0*k2*r2*r3 -
8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*µ*r1 + 2.0*k1*k2*µ*r2 + 2.0*k1*k3*µ*r1 +
2.0*k1*k3*µ*r2 + 2.0*k2*k3*µ*r1 + 2.0*k2*k3*µ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 +
4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 + 4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*µ*r1*r2 -
8.0*k3*µ*r1*r2 + 8.0*k2*r1*r2*r3 + 8.0*k3*r1*r2*r3 - 64.0*k1*µ^3*r1*r2 - 32.0*k2*µ^2*r1*r2 +
32.0*k1*µ^4*r1*r2 + 16.0*k2*µ^2*r1*r3 - 64.0*k2*µ^3*r1*r2 - 8.0*k1*µ^4*r1*r3 -
32.0*k2*µ^2*r2*r3 + 32.0*k2*µ^4*r1*r2 - 8.0*k1*µ^4*r2*r3 - 8.0*k2*µ^4*r1*r3 - 8.0*k2*µ^4*r2*r3 +
32.0*µ^4*r1*r2*r3 + 2.0*k1*k2*µ*r1*r2 + 4.0*k1*k2*µ*r1*r3 + 2.0*k1*k3*µ*r1*r2 +
4.0*k1*k2*µ*r2*r3 + 4.0*k1*k3*µ*r1*r3 + 2.0*k2*k3*µ*r1*r2 + 4.0*k1*k3*µ*r2*r3 +
4.0*k2*k3*µ*r1*r3 + 4.0*k2*k3*µ*r2*r3 - 4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 -
4.0*k2*k3*r1*r2*r3 - 16.0*k2*µ*r1*r2*r3 - 16.0*k3*µ*r1*r2*r3 - 16.0*k2*µ^2*r1*r2*r3 -
8.0*k1*µ^4*r1*r2*r3 - 8.0*k2*µ^4*r1*r2*r3 + 4.0*k1*k2*µ*r1*r2*r3 + 4.0*k1*k3*µ*r1*r2*r3 +
4.0*k2*k3*µ*r1*r2*r3)))/(EI^2*(16.0*k2*µ + 16.0*k3*µ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*µ^3 +
32.0*k2*µ^2 + 16.0*k2*µ^3 - 64.0*µ^3*r1 - 256.0*µ^3*r2 - 64.0*µ^4*r3 + 256.0*µ^3 + 256.0*µ^4 +
32.0*k1*µ^3*r1 + 16.0*k2*µ^2*r1 - 16.0*k1*µ^3*r2 + 16.0*k1*µ^4*r1 - 32.0*k2*µ^2*r2 +
32.0*k2*µ^3*r1 + 16.0*k1*µ^4*r2 + 16.0*k2*µ^2*r3 - 16.0*k2*µ^3*r2 + 16.0*k2*µ^4*r1 +
16.0*k2*µ^4*r2 + 64.0*µ^3*r1*r2 - 64.0*µ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*µ*r3 + 32.0*k3*µ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*µ*r1 + k1*k2*µ*r2 + k1*k3*µ*r1 + k1*k3*µ*r2 + k2*k3*µ*r1 + k2*k3*µ*r2 + 2.0*k1*k2*r1*r3
- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*µ*r1*r2 - 4.0*k3*µ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*µ^3*r1*r2 -
16.0*k2*µ^2*r1*r2 + 16.0*k1*µ^4*r1*r2 + 8.0*k2*µ^2*r1*r3 - 32.0*k2*µ^3*r1*r2 - 4.0*k1*µ^4*r1*r3
- 16.0*k2*µ^2*r2*r3 + 16.0*k2*µ^4*r1*r2 - 4.0*k1*µ^4*r2*r3 - 4.0*k2*µ^4*r1*r3 - 4.0*k2*µ^4*r2*r3
+ 16.0*µ^4*r1*r2*r3 + k1*k2*µ*r1*r2 + 2.0*k1*k2*µ*r1*r3 + k1*k3*µ*r1*r2 + 2.0*k1*k2*µ*r2*r3 +
2.0*k1*k3*µ*r1*r3 + k2*k3*µ*r1*r2 + 2.0*k1*k3*µ*r2*r3 + 2.0*k2*k3*µ*r1*r3 + 2.0*k2*k3*µ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*µ*r1*r2*r3 -
8.0*k3*µ*r1*r2*r3 - 8.0*k2*µ^2*r1*r2*r3 - 4.0*k1*µ^4*r1*r2*r3 - 4.0*k2*µ^4*r1*r2*r3 +
2.0*k1*k2*µ*r1*r2*r3 + 2.0*k1*k3*µ*r1*r2*r3 + 2.0*k2*k3*µ*r1*r2*r3)^2) +
(0.000868055555555555555555555555555556*L^9*q^2*(32.0*k2*µ + 32.0*k3*µ + 32.0*k2*r3 +
32.0*k3*r3 + 32.0*k1*µ^3 + 64.0*k2*µ^2 + 32.0*k2*µ^3 - 128.0*µ^3*r1 - 512.0*µ^3*r2 -
128.0*µ^4*r3 + 512.0*µ^3 + 512.0*µ^4 + 64.0*k1*µ^3*r1 + 32.0*k2*µ^2*r1 - 32.0*k1*µ^3*r2 +
32.0*k1*µ^4*r1 - 64.0*k2*µ^2*r2 + 64.0*k2*µ^3*r1 + 32.0*k1*µ^4*r2 + 32.0*k2*µ^2*r3 -
32.0*k2*µ^3*r2 + 32.0*k2*µ^4*r1 + 32.0*k2*µ^4*r2 + 128.0*µ^3*r1*r2 - 128.0*µ^4*r1*r2 +

2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 -
32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1
+ 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 + 2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 +
4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 + 4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 -
8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 + 8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 +
32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3 - 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 -
32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 - 8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 +
32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2 + 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 +
4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 + 2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 +
4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 - 4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 -
4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 - 16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 -
8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 + 4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 +
4.0*k2*k3*μ*r1*r2*r3)*(8.0*k2*μ^4 - 8.0*k3*μ - 6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 -
8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 - 32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 +
32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 - 128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 -
28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 + 8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 +
10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 +
32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 + 16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 -
1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 - 2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 -
16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 - 1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 -
2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 +
4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 +
16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 + 12.0*k2*μ^2*r2*r3 -
32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3 + 6.0*k2*μ^4*r2*r3
- 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 - 8.0*μ^6*r1*r2*r3 -
1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 + 4.0*k2*μ*r1*r2*r3 +
4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 +
8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 +
2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 +
16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3
+ 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 -
32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 +
16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 +
k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 -
16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 +
2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 -
1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 -
32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 -
32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3
- 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 +
k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 +
2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 -
2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 -
4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 +
2.0*k2*k3*μ*r1*r2*r3)^2) +
(0.00086805555555555555555555555555556*L^5*μ^5*q^2*(144.0*L^4*(16.0*k2*μ^2 - 4.0*k3*r3 -
16.0*k1*μ^3 - 4.0*k2*r3 - 16.0*k2*μ^4 + 16.0*k3*μ^3 + 32.0*μ^3*r1 + 64.0*μ^3*r2 - 32.0*μ^6*r3 -

64.0*μ^3 + 64.0*μ^6 - 16.0*k1*μ^3*r1 - 4.0*k2*μ^2*r1 + 16.0*k1*μ^3*r2 + 4.0*k2*μ^2*r2 +
8.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 - 8.0*k2*μ^4*r1 + 4.0*k1*μ^6*r1 + 16.0*k2*μ^4*r2 + 4.0*k1*μ^6*r2
+ 4.0*k2*μ^4*r3 + 4.0*k2*μ^6*r1 + 16.0*k3*μ^3*r3 + 4.0*k2*μ^6*r2 - 32.0*μ^3*r1*r2 -
16.0*μ^6*r1*r2 - 1.0*k1*k2*r3 - 1.0*k1*k3*r3 + k2*k3*r3 + 2.0*k2*r1*r3 + 4.0*k2*r2*r3 +
2.0*k3*r1*r3 + 4.0*k3*r2*r3 - 2.0*k2*k3*μ*r3 - 1.0*k1*k2*r1*r3 + k1*k2*r2*r3 - 1.0*k1*k3*r1*r3 +
k1*k3*r2*r3 - 1.0*k2*k3*r2*r3 - 2.0*k2*r1*r2*r3 - 2.0*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1 +
2.0*k1*k2*μ^2*r2 - 1.0*k1*k2*μ^3*r1 - 1.0*k1*k2*μ^3*r2 + k1*k3*μ^3*r1 + k1*k3*μ^3*r2 +
k2*k3*μ^3*r1 + k2*k3*μ^3*r2 + 16.0*k1*μ^3*r1*r2 - 4.0*k2*μ^2*r1*r2 - 2.0*k2*μ^2*r1*r3 +
4.0*k2*μ^3*r1*r2 + 2.0*k2*μ^2*r2*r3 + 8.0*k2*μ^4*r1*r2 - 4.0*k3*μ^3*r1*r2 + 4.0*k1*μ^6*r1*r2 +
2.0*k2*μ^4*r1*r3 - 2.0*k1*μ^6*r1*r3 - 4.0*k2*μ^4*r2*r3 + 4.0*k2*μ^6*r1*r2 - 2.0*k1*μ^6*r2*r3 -
2.0*k2*μ^6*r1*r3 - 2.0*k2*μ^6*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
+ k1*k2*r1*r2*r3 + k1*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1*r2 + k1*k2*μ^2*r1*r3 -
1.0*k1*k2*μ^3*r1*r2 + k1*k2*μ^2*r2*r3 + k1*k3*μ^3*r1*r2 + k1*k3*μ^3*r1*r3 + k2*k3*μ^3*r1*r2
+ k1*k3*μ^3*r2*r3 + k2*k3*μ^3*r1*r3 + k2*k3*μ^3*r2*r3 - 2.0*k2*μ^2*r1*r2*r3 -
2.0*k2*μ^4*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 - 2.0*k1*μ^6*r1*r2*r3 - 2.0*k2*μ^6*r1*r2*r3 +
k1*k2*μ^2*r1*r2*r3 + k1*k3*μ^3*r1*r2*r3 + k2*k3*μ^3*r1*r2*r3 + k2*k3*μ*r1*r2*r3)^2 +
2.0*L^4*k2*(32.0*k2*μ + 32.0*k3*μ + 32.0*k2*r3 + 32.0*k3*r3 + 32.0*k1*μ^3 + 64.0*k2*μ^2 +
32.0*k2*μ^3 - 128.0*μ^3*r1 - 512.0*μ^3*r2 - 128.0*μ^4*r3 + 512.0*μ^3 + 512.0*μ^4 +
64.0*k1*μ^3*r1 + 32.0*k2*μ^2*r1 - 32.0*k1*μ^3*r2 + 32.0*k1*μ^4*r1 - 64.0*k2*μ^2*r2 +
64.0*k2*μ^3*r1 + 32.0*k1*μ^4*r2 + 32.0*k2*μ^2*r3 - 32.0*k2*μ^3*r2 + 32.0*k2*μ^4*r1 +
32.0*k2*μ^4*r2 + 128.0*μ^3*r1*r2 - 128.0*μ^4*r1*r2 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3
+ 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 - 32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 +
2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1 + 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 +
2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 + 4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 +
4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 - 8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 +
8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 + 32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3
- 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 - 32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 -
8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 + 32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2
+ 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 + 4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 +
2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 + 4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 -
4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 - 4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 -
16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 - 8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 +
4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 + 4.0*k2*k3*μ*r1*r2*r3)*(8.0*k1*μ^4 - 10.0*k3*r3 -
8.0*k1*μ^2 - 16.0*k1*μ^3 - 8.0*k3*μ + 16.0*k3*μ^2 + 8.0*k3*μ^3 + 16.0*μ^2*r1 + 32.0*μ^2*r2 +
64.0*μ^3*r1 - 16.0*μ^2*r3 + 160.0*μ^3*r2 - 160.0*μ^4*r2 + 16.0*μ^6*r1 - 64.0*μ^5*r3 - 32.0*μ^6*r2
- 16.0*μ^6*r3 - 32.0*μ^2 - 160.0*μ^3 + 160.0*μ^5 + 32.0*μ^6 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 -
32.0*k1*μ^3*r1 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^3*r2 - 24.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 +
4.0*k3*μ^3*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 + 32.0*k3*μ^2*r3 - 8.0*k3*μ^3*r2 + 8.0*k3*μ^3*r3
- 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 - 64.0*μ^3*r1*r2 + 16.0*μ^2*r2*r3 + 64.0*μ^4*r1*r2 -
40.0*μ^5*r1*r2 + 40.0*μ^4*r2*r3 - 16.0*μ^6*r1*r2 - 8.0*μ^6*r1*r3 + 16.0*μ^6*r2*r3 - 1.0*k1*k3*r3
+ 2.0*k3*μ*r1 - 2.0*k3*μ*r2 + 4.0*k3*r1*r3 + 10.0*k3*r2*r3 - 1.0*k1*k3*μ*r1 - 1.0*k1*k3*μ*r2 +
k1*k3*μ*r3 - 2.0*k1*k3*r1*r3 + k1*k3*r2*r3 + 2.0*k3*μ*r1*r2 - 20.0*k3*μ*r2*r3 - 4.0*k3*r1*r2*r3 +
k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 32.0*k1*μ^3*r1*r2 +
4.0*k1*μ^2*r2*r3 - 32.0*k1*μ^4*r1*r2 - 4.0*k3*μ^2*r1*r2 + 10.0*k1*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r2 +
6.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 + 4.0*k3*μ^3*r1*r3 - 4.0*k1*μ^5*r2*r3 - 8.0*k3*μ^3*r2*r3 -
8.0*μ^2*r1*r2*r3 - 16.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k1*k3*μ*r1*r2 -
3.0*k1*k3*μ*r2*r3 + 2.0*k1*k3*r1*r2*r3 + 8.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 +

2.0*k1*k3*μ^2*r1*r3 + 2.0*k1*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 8.0*k1*μ^4*r1*r2*r3 -
8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 -
4.0*k1*k3*μ*r1*r2*r3) + 128.0*L^4*μ*(r2 - 1.0)*(4.0*k2 + 4.0*k3 + k1*k2 + k1*k3 - 1.0*k2*k3 +
20.0*k2*μ - 2.0*k2*r1 - 2.0*k3*r1 + 8.0*k2*r3 + 8.0*k3*r3 + 16.0*k1*μ^3 - 16.0*k2*μ^2 -
16.0*k2*μ^3 + 16.0*k3*μ^2 + 4.0*k1*μ^5 + 20.0*k2*μ^4 + 4.0*k2*μ^5 - 32.0*μ^3*r1 - 16.0*μ^3*r3 -
16.0*μ^5*r1 - 32.0*μ^5*r3 + 64.0*μ^3 + 64.0*μ^5 - 1.0*k1*k2*μ^2 + k1*k3*μ^2 + k2*k3*μ^2 +
16.0*k1*μ^3*r1 + 4.0*k2*μ^2*r1 - 4.0*k3*μ^2*r1 - 4.0*k1*μ^3*r3 + 8.0*k1*μ^5*r1 + 10.0*k2*μ^4*r1
+ 4.0*k2*μ^3*r3 + 8.0*k2*μ^5*r1 + 16.0*k3*μ^2*r3 - 2.0*k1*μ^5*r3 - 8.0*k2*μ^4*r3 - 2.0*k2*μ^5*r3
+ 8.0*μ^3*r1*r3 + 8.0*μ^5*r1*r3 + 2.0*k1*k2*μ + 2.0*k2*k3*μ + k1*k2*r1 + k1*k3*r1 +
2.0*k1*k2*r3 + 2.0*k1*k3*r3 - 2.0*k2*k3*r3 - 8.0*k2*μ*r1 + 10.0*k2*μ*r3 - 4.0*k2*r1*r3 -
4.0*k3*r1*r3 + 4.0*k1*k2*μ*r1 + k1*k2*μ*r3 + k2*k3*μ*r1 + 4.0*k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 +
2.0*k1*k3*r1*r3 - 4.0*k2*μ*r1*r3 - 2.0*k1*k2*μ^2*r1 + 2.0*k1*k3*μ^2*r1 + 2.0*k2*k3*μ^2*r1 +
k1*k3*μ^2*r3 + k2*k3*μ^2*r3 - 4.0*k1*μ^3*r1*r3 - 4.0*k3*μ^2*r1*r3 - 4.0*k1*μ^5*r1*r3 -
4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^5*r1*r3 + 2.0*k1*k2*μ*r1*r3 + 2.0*k2*k3*μ*r1*r3 +
2.0*k1*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r1*r3)*(8.0*k2*μ + 10.0*k2*r3 - 8.0*k1*μ^2 + 8.0*k2*μ^2 +
8.0*k1*μ^4 + 8.0*k2*μ^3 + 16.0*k3*μ^2 + 8.0*k2*μ^4 + 16.0*μ^2*r1 + 32.0*μ^2*r2 - 16.0*μ^2*r3 -
32.0*μ^4*r1 - 128.0*μ^4*r2 - 32.0*μ^4*r3 - 64.0*μ^5*r3 - 32.0*μ^2 + 128.0*μ^4 + 160.0*μ^5 -
8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^4*r1 - 8.0*k2*μ^2*r2 + 4.0*k2*μ^3*r1 -
8.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 + 4.0*k2*μ^2*r3 - 8.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 - 2.0*k1*μ^4*r3
+ 10.0*k1*μ^5*r2 - 8.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 32.0*k3*μ^2*r3 - 2.0*k2*μ^4*r3 +
10.0*k2*μ^5*r2 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 + 16.0*μ^2*r2*r3 + 32.0*μ^4*r1*r2 +
8.0*μ^4*r1*r3 - 40.0*μ^5*r1*r2 + 32.0*μ^4*r2*r3 + k1*k2*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 +
16.0*k2*μ*r3 + 16.0*k3*μ*r3 - 4.0*k2*r1*r3 - 10.0*k2*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 +
k1*k3*μ*r3 + k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 - 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 +
4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 - 16.0*k3*μ*r2*r3 + 4.0*k2*r1*r2*r3 + k1*k3*μ^2*r1 +
k1*k3*μ^2*r2 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 +
4.0*k1*μ^2*r2*r3 - 16.0*k1*μ^4*r1*r2 - 4.0*k2*μ^3*r1*r2 - 4.0*k3*μ^2*r1*r2 - 4.0*k1*μ^4*r1*r3 +
10.0*k1*μ^5*r1*r2 - 4.0*k2*μ^2*r2*r3 - 16.0*k2*μ^4*r1*r2 + 2.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 -
4.0*k2*μ^4*r1*r3 + 10.0*k2*μ^5*r1*r2 - 4.0*k1*μ^5*r2*r3 + 2.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 -
4.0*k2*μ^5*r2*r3 - 8.0*μ^2*r1*r2*r3 - 8.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + k1*k2*μ*r1*r2 +
2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 - 1.0*k1*k3*μ*r2*r3 +
2.0*k2*k3*μ*r1*r3 - 1.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 4.0*k2*μ*r1*r2*r3 +
4.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + k2*k3*μ^2*r1*r2 +
2.0*k1*k3*μ^2*r2*r3 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 +
4.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 + 4.0*k2*μ^4*r1*r2*r3 -
4.0*k2*μ^5*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 -
2.0*k1*k3*μ*r1*r2*r3 - 2.0*k2*k3*μ*r1*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 +
16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3
+ 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 -
32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 +
16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 +
k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 -
16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 +
2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 -
1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 -
32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 -
32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3

- 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) +
(0.000062003968253968253968253968253968*L^7*μ^7*q^2*(64.0*L^2*(8.0*k2*μ + 10.0*k2*r3 - 8.0*k1*μ^2 + 8.0*k2*μ^2 + 8.0*k1*μ^4 + 8.0*k2*μ^3 + 16.0*k3*μ^2 + 8.0*k2*μ^4 + 16.0*μ^2*r1 + 32.0*μ^2*r2 - 16.0*μ^2*r3 - 32.0*μ^4*r1 - 128.0*μ^4*r2 - 32.0*μ^4*r3 - 64.0*μ^5*r3 - 32.0*μ^2 + 128.0*μ^4 + 160.0*μ^5 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^4*r1 - 8.0*k2*μ^2*r2 + 4.0*k2*μ^3*r1 - 8.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 + 4.0*k2*μ^2*r3 - 8.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 - 8.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 32.0*k3*μ^2*r3 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 + 16.0*μ^2*r2*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 40.0*μ^5*r1*r2 + 32.0*μ^4*r2*r3 + k1*k2*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 + 16.0*k2*μ*r3 + 16.0*k3*μ*r3 - 4.0*k2*r1*r3 - 10.0*k2*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r3 + k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 - 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 - 16.0*k3*μ*r2*r3 + 4.0*k2*r1*r2*r3 + k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 4.0*k1*μ^2*r2*r3 - 16.0*k1*μ^4*r1*r2 - 4.0*k2*μ^3*r1*r2 - 4.0*k3*μ^2*r1*r2 - 4.0*k1*μ^4*r1*r3 + 10.0*k1*μ^5*r1*r2 - 4.0*k2*μ^2*r2*r3 - 16.0*k2*μ^4*r1*r2 + 2.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 - 4.0*k2*μ^4*r1*r3 + 10.0*k2*μ^5*r1*r2 - 4.0*k1*μ^5*r2*r3 + 2.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^2*r1*r2*r3 - 8.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 - 1.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 - 1.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + k2*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r2*r3 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 4.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 + 4.0*k2*μ^4*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 - 2.0*k1*k3*μ*r1*r2*r3 - 2.0*k2*k3*μ*r1*r2*r3)^2 + 24.0*L^2*(32.0*k2*μ + 32.0*k3*μ + 32.0*k2*r3 + 32.0*k3*r3 + 32.0*k1*μ^3 + 64.0*k2*μ^2 + 32.0*k2*μ^3 - 128.0*μ^3*r1 - 512.0*μ^3*r2 - 128.0*μ^4*r3 + 512.0*μ^3 + 512.0*μ^4 + 64.0*k1*μ^3*r1 + 32.0*k2*μ^2*r1 - 32.0*k1*μ^3*r2 + 32.0*k1*μ^4*r1 - 64.0*k2*μ^2*r2 + 64.0*k2*μ^3*r1 + 32.0*k1*μ^4*r2 + 32.0*k2*μ^2*r3 - 32.0*k2*μ^3*r2 + 32.0*k2*μ^4*r1 + 32.0*k2*μ^4*r2 + 128.0*μ^3*r1*r2 - 128.0*μ^4*r1*r2 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 - 32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1 + 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 + 2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 + 4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 + 4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 - 8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 + 8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 + 32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3 - 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 - 32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 - 8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 + 32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2 + 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 + 4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 + 2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 + 4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 - 4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 - 4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 - 16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 - 8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 + 4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 + 4.0*k2*k3*μ*r1*r2*r3)*(16.0*k2*μ^2 - 4.0*k3*r3 - 16.0*k1*μ^3 - 4.0*k2*r3 - 16.0*k2*μ^4 + 16.0*k3*μ^3 + 32.0*μ^3*r1 + 64.0*μ^3*r2 - 32.0*μ^6*r3 - 64.0*μ^3 + 64.0*μ^6 -

16.0*k1*μ^3*r1 - 4.0*k2*μ^2*r1 + 16.0*k1*μ^3*r2 + 4.0*k2*μ^2*r2 + 8.0*k2*μ^2*r3 -
16.0*k2*μ^3*r2 - 8.0*k2*μ^4*r1 + 4.0*k1*μ^6*r1 + 16.0*k2*μ^4*r2 + 4.0*k1*μ^6*r2 +
4.0*k2*μ^4*r3 + 4.0*k2*μ^6*r1 + 16.0*k3*μ^3*r3 + 4.0*k2*μ^6*r2 - 32.0*μ^3*r1*r2 -
16.0*μ^6*r1*r2 - 1.0*k1*k2*r3 - 1.0*k1*k3*r3 + k2*k3*r3 + 2.0*k2*r1*r3 + 4.0*k2*r2*r3 +
2.0*k3*r1*r3 + 4.0*k3*r2*r3 - 2.0*k2*k3*μ*r3 - 1.0*k1*k2*r1*r3 + k1*k2*r2*r3 - 1.0*k1*k3*r1*r3 +
k1*k3*r2*r3 - 1.0*k2*k3*r2*r3 - 2.0*k2*r1*r2*r3 - 2.0*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1 +
2.0*k1*k2*μ^2*r2 - 1.0*k1*k2*μ^3*r1 - 1.0*k1*k2*μ^3*r2 + k1*k3*μ^3*r1 + k1*k3*μ^3*r2 +
k2*k3*μ^3*r1 + k2*k3*μ^3*r2 + 16.0*k1*μ^3*r1*r2 - 4.0*k2*μ^2*r1*r2 - 2.0*k2*μ^2*r1*r3 +
4.0*k2*μ^3*r1*r2 + 2.0*k2*μ^2*r2*r3 + 8.0*k2*μ^4*r1*r2 - 4.0*k3*μ^3*r1*r2 + 4.0*k1*μ^6*r1*r2 +
2.0*k2*μ^4*r1*r3 - 2.0*k1*μ^6*r1*r3 - 4.0*k2*μ^4*r2*r3 + 4.0*k2*μ^6*r1*r2 - 2.0*k1*μ^6*r2*r3 -
2.0*k2*μ^6*r1*r3 - 2.0*k2*μ^6*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
+ k1*k2*r1*r2*r3 + k1*k3*r1*r2*r3 + 2.0*k1*k2*μ^2*r1*r2 + k1*k2*μ^2*r1*r3 -
1.0*k1*k2*μ^3*r1*r2 + k1*k2*μ^2*r2*r3 + k1*k3*μ^3*r1*r2 + k1*k3*μ^3*r1*r3 + k2*k3*μ^3*r1*r2
+ k1*k3*μ^3*r2*r3 + k2*k3*μ^3*r1*r3 + k2*k3*μ^3*r2*r3 - 2.0*k2*μ^2*r1*r2*r3 -
2.0*k2*μ^4*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 - 2.0*k1*μ^6*r1*r2*r3 - 2.0*k2*μ^6*r1*r2*r3 +
k1*k2*μ^2*r1*r2*r3 + k1*k3*μ^3*r1*r2*r3 + k2*k3*μ^3*r1*r2*r3 +
k2*k3*μ*r1*r2*r3)))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 +
32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 +
32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 +
32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 +
16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3
- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 -
16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3
- 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3
+ 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 -
8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 +
2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) +
(0.0004340277777777777777777777777777778*L^9*k1^2*q^2*(8.0*k2*μ^4 - 8.0*k3*μ - 6.0*k2*r3 -
6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 - 32.0*μ^4*r1 +
32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 - 128.0*μ^4 -
32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 + 8.0*k2*μ^3*r2 -
4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3 +
10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 + 16.0*μ^6*r1*r2
+ 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 - 2.0*k3*μ*r1 -
16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 - 1.0*k2*k3*μ*r1 -
1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 +
2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 + k2*k3*μ^2*r1 +
k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 + 12.0*k2*μ^2*r2*r3 -
32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3 + 6.0*k2*μ^4*r2*r3
- 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 - 8.0*μ^6*r1*r2*r3 -
1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 + 4.0*k2*μ*r1*r2*r3 +
4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 +

8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 - 4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3)^2)/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) + (0.0011574074074074074074074074074074*L^9*q^2*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 - 24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 + 32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 + 2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 + 20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 + 8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 - 8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ + k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 - 2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 + 2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 + 2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 + 4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 + k1*k2*μ^2*r3 + k1*k2*μ^3*r2 - 1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 + 4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 + k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 + 8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 - 4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*μ^2*r2*r3 - 1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 - 1.0*k2*k3*μ^3*r2*r3)*(128.0*k1*μ^3 - 32.0*k3*μ - 16.0*k2*r3 - 16.0*k3*r3 - 32.0*k2*μ - 160.0*k2*μ^2 + 128.0*k1*μ^4 - 192.0*k2*μ^3 + 64.0*k2*μ^4 - 64.0*k3*μ^3 + 32.0*k1*μ^6 + 160.0*k2*μ^5 + 32.0*k2*μ^6 + 512.0*μ^3*r1 + 256.0*μ^3*r2 + 1024.0*μ^4*r1 + 256.0*μ^4*r2 + 128.0*μ^4*r3 + 512.0*μ^6*r1 + 256.0*μ^6*r2 + 128.0*μ^6*r3 - 256.0*μ^3 - 512.0*μ^4 - 256.0*μ^6 - 256.0*μ^3*r1^2 - 512.0*μ^4*r1^2 - 256.0*μ^6*r1^2 + 16.0*k1*k2*μ^2 - 8.0*k1*k2*μ^3 + 8.0*k1*k3*μ^3 + 16.0*k2*k3*μ^2 + 8.0*k2*k3*μ^3 - 544.0*k1*μ^3*r1 - 32.0*k2*μ*r1^2 + 320.0*k2*μ^2*r1 - 128.0*k1*μ^3*r2 - 640.0*k1*μ^4*r1 + 160.0*k2*μ^2*r2 + 384.0*k2*μ^3*r1 - 32.0*k3*μ*r1^2 - 80.0*k2*μ^2*r3 + 192.0*k2*μ^3*r2 - 128.0*k2*μ^4*r1 + 128.0*k3*μ^3*r1 - 32.0*k1*μ^4*r3 - 160.0*k1*μ^6*r1 - 320.0*k2*μ^4*r2 - 320.0*k2*μ^5*r1 + 64.0*k3*μ^3*r2 - 32.0*k1*μ^6*r2 - 16.0*k2*μ^4*r3 + 80.0*k2*μ^5*r2 - 64.0*k2*μ^6*r1 - 64.0*k3*μ^3*r3 - 16.0*k1*μ^6*r3 - 64.0*k2*μ^5*r3 - 32.0*k2*μ^6*r2 - 16.0*k2*μ^6*r3 - 16.0*k2*r1^2*r3 - 16.0*k3*r1^2*r3 - 512.0*μ^3*r1*r2 - 512.0*μ^4*r1*r2 - 256.0*μ^4*r1*r3 - 64.0*μ^4*r2*r3 - 512.0*μ^6*r1*r2 - 256.0*μ^6*r1*r3 - 128.0*μ^6*r2*r3 + 128.0*k1*μ^3*r1^2 - 160.0*k2*μ^2*r1^2 + 32.0*k1*μ^4*r1^2 - 192.0*k2*μ^3*r1^2 + 64.0*k2*μ^4*r1^2 - 64.0*k3*μ^3*r1^2 - 16.0*k1*μ^6*r1^2 + 160.0*k2*μ^5*r1^2 + 32.0*k2*μ^6*r1^2 + 256.0*μ^3*r1^2*r2 + 256.0*μ^4*r1^2*r2 +

128.0*μ^4*r1^2*r3 + 256.0*μ^6*r1^2*r2 + 128.0*μ^6*r1^2*r3 + 8.0*k1*k2*μ + 8.0*k1*k3*μ -
8.0*k2*k3*μ + 8.0*k1*k2*r3 + 8.0*k1*k3*r3 - 16.0*k2*k3*r3 + 64.0*k2*μ*r1 + 16.0*k2*μ*r2 +
64.0*k3*μ*r1 - 64.0*k2*μ*r3 + 16.0*k3*μ*r2 - 64.0*k3*μ*r3 + 32.0*k2*r1*r3 + 16.0*k2*r2*r3 +
32.0*k3*r1*r3 + 16.0*k3*r2*r3 - 40.0*k1*k2*μ*r1 - 40.0*k1*k3*μ*r1 + 16.0*k1*k2*μ*r3 +
16.0*k2*k3*μ*r1 + 16.0*k1*k3*μ*r3 - 8.0*k2*k3*μ*r2 + 8.0*k2*k3*μ*r3 - 34.0*k1*k2*r1*r3 -
8.0*k1*k2*r2*r3 - 34.0*k1*k3*r1*r3 - 8.0*k1*k3*r2*r3 + 32.0*k2*k3*r1*r3 + 16.0*k2*k3*r2*r3 -
32.0*k2*μ*r1*r2 + 128.0*k2*μ*r1*r3 - 32.0*k3*μ*r1*r2 + 32.0*k2*μ*r2*r3 + 128.0*k3*μ*r1*r3 +
32.0*k3*μ*r2*r3 - 32.0*k2*r1*r2*r3 - 32.0*k3*r1*r2*r3 + 2.0*k1*k2*μ*r1^2 - 104.0*k1*k2*μ^2*r1 -
16.0*k1*k2*μ^2*r2 - 8.0*k1*k2*μ^3*r1 + 2.0*k1*k3*μ*r1^2 + 8.0*k1*k2*μ^2*r3 + 8.0*k1*k2*μ^3*r2
+ 24.0*k1*k2*μ^4*r1 - 40.0*k1*k3*μ^3*r1 - 8.0*k2*k3*μ*r1^2 - 32.0*k2*k3*μ^2*r1 -
8.0*k1*k3*μ^3*r2 + 8.0*k2*k3*μ^2*r2 - 16.0*k2*k3*μ^3*r1 + 8.0*k1*k3*μ^3*r3 +
32.0*k2*k3*μ^2*r3 - 8.0*k2*k3*μ^3*r2 + 8.0*k2*k3*μ^3*r3 + 8.0*k1*k2*r1^2*r3 +
8.0*k1*k3*r1^2*r3 - 16.0*k2*k3*r1^2*r3 + 544.0*k1*μ^3*r1*r2 + 16.0*k2*μ*r1^2*r2 -
320.0*k2*μ^2*r1*r2 + 96.0*k1*μ^4*r1*r2 - 64.0*k2*μ*r1^2*r3 + 160.0*k2*μ^2*r1*r3 -
384.0*k2*μ^3*r1*r2 + 16.0*k3*μ*r1^2*r2 + 160.0*k1*μ^4*r1*r3 + 80.0*k2*μ^2*r2*r3 +
640.0*k2*μ^4*r1*r2 - 64.0*k3*μ*r1^2*r3 - 128.0*k3*μ^3*r1*r2 + 160.0*k1*μ^6*r1*r2 +
32.0*k2*μ^4*r1*r3 - 160.0*k2*μ^5*r1*r2 + 128.0*k3*μ^3*r1*r3 + 80.0*k1*μ^6*r1*r3 +
80.0*k2*μ^4*r2*r3 + 128.0*k2*μ^5*r1*r3 + 64.0*k2*μ^6*r1*r2 + 64.0*k3*μ^3*r2*r3 +
16.0*k1*μ^6*r2*r3 - 32.0*k2*μ^5*r2*r3 + 32.0*k2*μ^6*r1*r3 + 16.0*k2*μ^6*r2*r3 +
16.0*k2*r1^2*r2*r3 + 16.0*k3*r1^2*r2*r3 + 128.0*μ^4*r1*r2*r3 + 256.0*μ^6*r1*r2*r3 -
32.0*k1*k2*μ^2*r1^2 - 92.0*k1*k2*μ^3*r1^2 - 4.0*k1*k3*μ^3*r1^2 + 16.0*k2*k3*μ^2*r1^2 +
30.0*k1*k2*μ^5*r1^2 + 8.0*k2*k3*μ^3*r1^2 - 128.0*k1*μ^3*r1^2*r2 + 160.0*k2*μ^2*r1^2*r2 +
96.0*k1*μ^4*r1^2*r2 - 80.0*k2*μ^2*r1^2*r3 + 192.0*k2*μ^3*r1^2*r2 - 8.0*k1*μ^4*r1^2*r3 -
320.0*k2*μ^4*r1^2*r2 + 64.0*k3*μ^3*r1^2*r2 + 16.0*k1*μ^6*r1^2*r2 - 16.0*k2*μ^4*r1^2*r3 +
80.0*k2*μ^5*r1^2*r2 - 64.0*k3*μ^3*r1^2*r3 + 8.0*k1*μ^6*r1^2*r3 - 64.0*k2*μ^5*r1^2*r3 -
32.0*k2*μ^6*r1^2*r2 - 16.0*k2*μ^6*r1^2*r3 - 64.0*μ^4*r1^2*r2*r3 - 128.0*μ^6*r1^2*r2*r3 +
3.0*k1*k2*k3*μ^2*r1^2 + 32.0*k1*k2*μ^2*r1^2*r2 - 16.0*k1*k2*μ^2*r1^2*r3 +
92.0*k1*k2*μ^3*r1^2*r2 - 96.0*k1*k2*μ^4*r1^2*r2 + 4.0*k1*k3*μ^3*r1^2*r2 +
8.0*k2*k3*μ^2*r1^2*r2 + 30.0*k1*k2*μ^5*r1^2*r2 - 4.0*k1*k3*μ^3*r1^2*r3 +
32.0*k2*k3*μ^2*r1^2*r3 - 8.0*k2*k3*μ^3*r1^2*r2 - 12.0*k1*k2*μ^5*r1^2*r3 +
8.0*k2*k3*μ^3*r1^2*r3 + 80.0*k2*μ^2*r1^2*r2*r3 - 24.0*k1*μ^4*r1^2*r2*r3 +
80.0*k2*μ^4*r1^2*r2*r3 + 64.0*k3*μ^3*r1^2*r2*r3 - 8.0*k1*μ^6*r1^2*r2*r3 -
32.0*k2*μ^5*r1^2*r2*r3 + 16.0*k2*μ^6*r1^2*r2*r3 - 3.0*k1*k2*k3*r1*r3 + 6.0*k1*k2*μ*r1*r2 -
80.0*k1*k2*μ*r1*r3 + 6.0*k1*k3*μ*r1*r2 - 80.0*k1*k3*μ*r1*r3 + 16.0*k2*k3*μ*r1*r2 -
16.0*k2*k3*μ*r1*r3 - 40.0*k2*k3*μ*r2*r3 + 34.0*k1*k2*r1*r2*r3 + 34.0*k1*k3*r1*r2*r3 -
32.0*k2*k3*r1*r2*r3 - 64.0*k2*μ*r1*r2*r3 - 64.0*k3*μ*r1*r2*r3 - 3.0*k1*k2*k3*μ*r1^2 -
6.0*k1*k2*k3*r1^2*r3 + 6.0*k1*k2*μ*r1^2*r2 + 104.0*k1*k2*μ^2*r1*r2 + 4.0*k1*k2*μ*r1^2*r3 -
52.0*k1*k2*μ^2*r1*r3 + 8.0*k1*k2*μ^3*r1*r2 + 6.0*k1*k3*μ*r1^2*r2 - 8.0*k1*k2*μ^2*r2*r3 -
72.0*k1*k2*μ^4*r1*r2 + 4.0*k1*k3*μ*r1^2*r3 + 40.0*k1*k3*μ^3*r1*r2 - 8.0*k2*k3*μ*r1^2*r2 -
16.0*k2*k3*μ^2*r1*r2 - 6.0*k1*k2*μ^4*r1*r3 + 30.0*k1*k2*μ^5*r1*r2 - 40.0*k1*k3*μ^3*r1*r3 +
8.0*k2*k3*μ*r1^2*r3 - 64.0*k2*k3*μ^2*r1*r3 + 16.0*k2*k3*μ^3*r1*r2 - 8.0*k1*k3*μ^3*r2*r3 +
16.0*k2*k3*μ^2*r2*r3 - 16.0*k2*k3*μ^3*r1*r3 - 8.0*k2*k3*μ^3*r2*r3 - 8.0*k1*k2*r1^2*r2*r3 -
8.0*k1*k3*r1^2*r2*r3 + 16.0*k2*k3*r1^2*r2*r3 + 32.0*k2*μ*r1^2*r2*r3 - 160.0*k2*μ^2*r1*r2*r3 -
24.0*k1*μ^4*r1*r2*r3 + 32.0*k3*μ*r1^2*r2*r3 - 160.0*k2*μ^4*r1*r2*r3 - 128.0*k3*μ^3*r1*r2*r3 -
80.0*k1*μ^6*r1*r2*r3 + 64.0*k2*μ^5*r1*r2*r3 - 32.0*k2*μ^6*r1*r2*r3 - 3.0*k1*k2*k3*μ*r1^2*r2 +
3.0*k1*k2*k3*μ^2*r1*r2 + 6.0*k1*k2*k3*r1^2*r2*r3 + 12.0*k1*k2*μ*r1^2*r2*r3 +
52.0*k1*k2*μ^2*r1*r2*r3 + 12.0*k1*k3*μ*r1^2*r2*r3 + 18.0*k1*k2*μ^4*r1*r2*r3 +

40.0*k1*k3*μ^3*r1*r2*r3 - 40.0*k2*k3*μ*r1^2*r2*r3 - 32.0*k2*k3*μ^2*r1*r2*r3 -
12.0*k1*k2*μ^5*r1*r2*r3 + 16.0*k2*k3*μ^3*r1*r2*r3 + 3.0*k1*k2*k3*μ^2*r1^2*r2 +
6.0*k1*k2*k3*μ^2*r1^2*r3 + 16.0*k1*k2*μ^2*r1^2*r2*r3 + 24.0*k1*k2*μ^4*r1^2*r2*r3 +
4.0*k1*k3*μ^3*r1^2*r2*r3 + 16.0*k2*k3*μ^2*r1^2*r2*r3 - 12.0*k1*k2*μ^5*r1^2*r2*r3 -
8.0*k2*k3*μ^3*r1^2*r2*r3 - 3.0*k1*k2*k3*μ*r1*r2 + 3.0*k1*k2*k3*μ*r1*r3 +
3.0*k1*k2*k3*r1*r2*r3 + 12.0*k1*k2*μ*r1*r2*r3 + 12.0*k1*k3*μ*r1*r2*r3 + 80.0*k2*k3*μ*r1*r2*r3
+ 6.0*k1*k2*k3*μ^2*r1^2*r2*r3 - 9.0*k1*k2*k3*μ*r1*r2*r3 - 12.0*k1*k2*k3*μ*r1^2*r2*r3 +
6.0*k1*k2*k3*μ^2*r1*r2*r3))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 +
16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 +
256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2
+ 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 +
16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3
- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 -
16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3
- 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3
+ 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 -
8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 +
2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) +
(0.000434027777777777777777777777777778*L^9*k2^2*μ*q^2*(8.0*k1*μ^4 - 10.0*k3*r3 -
8.0*k1*μ^2 - 16.0*k1*μ^3 - 8.0*k3*μ + 16.0*k3*μ^2 + 8.0*k3*μ^3 + 16.0*μ^2*r1 + 32.0*μ^2*r2 +
64.0*μ^3*r1 - 16.0*μ^2*r3 + 160.0*μ^3*r2 - 160.0*μ^4*r2 + 16.0*μ^6*r1 - 64.0*μ^5*r3 - 32.0*μ^6*r2
- 16.0*μ^6*r3 - 32.0*μ^2 - 160.0*μ^3 + 160.0*μ^5 + 32.0*μ^6 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 -
32.0*k1*μ^3*r1 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^3*r2 - 24.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 +
4.0*k3*μ^3*r1 - 2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 + 32.0*k3*μ^2*r3 - 8.0*k3*μ^3*r2 + 8.0*k3*μ^3*r3
- 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 - 64.0*μ^3*r1*r2 + 16.0*μ^2*r2*r3 + 64.0*μ^4*r1*r2 -
40.0*μ^5*r1*r2 + 40.0*μ^4*r2*r3 - 16.0*μ^6*r1*r2 - 8.0*μ^6*r1*r3 + 16.0*μ^6*r2*r3 - 1.0*k1*k3*r3
+ 2.0*k3*μ*r1 - 2.0*k3*μ*r2 + 4.0*k3*r1*r3 + 10.0*k3*r2*r3 - 1.0*k1*k3*μ*r1 - 1.0*k1*k3*μ*r2 +
k1*k3*μ*r3 - 2.0*k1*k3*r1*r3 + k1*k3*r2*r3 + 2.0*k3*μ*r1*r2 - 20.0*k3*μ*r2*r3 - 4.0*k3*r1*r2*r3 +
k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 32.0*k1*μ^3*r1*r2 +
4.0*k1*μ^2*r2*r3 - 32.0*k1*μ^4*r1*r2 - 4.0*k3*μ^2*r1*r2 + 10.0*k1*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r2 +
6.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 + 4.0*k3*μ^3*r1*r3 - 4.0*k1*μ^5*r2*r3 - 8.0*k3*μ^3*r2*r3 -
8.0*μ^2*r1*r2*r3 - 16.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k1*k3*μ*r1*r2 -
3.0*k1*k3*μ*r2*r3 + 2.0*k1*k3*r1*r2*r3 + 8.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 +
2.0*k1*k3*μ^2*r1*r3 + 2.0*k1*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 8.0*k1*μ^4*r1*r2*r3 -
8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 -
4.0*k1*k3*μ*r1*r2*r3)^2)/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 +
32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 +
32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 +
32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 +
16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 +
32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 +
k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3

- 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 -
4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 -
16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3
- 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3
+ 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 +
2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3
- 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 -
8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 +
2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) -
(0.000868055555555555555555555555555556*L^9*μ^8*q^2*(32.0*k2*μ + 32.0*k3*μ + 32.0*k2*r3 +
32.0*k3*r3 + 32.0*k1*μ^3 + 64.0*k2*μ^2 + 32.0*k2*μ^3 - 128.0*μ^3*r1 - 512.0*μ^3*r2 -
128.0*μ^4*r3 + 512.0*μ^3 + 512.0*μ^4 + 64.0*k1*μ^3*r1 + 32.0*k2*μ^2*r1 - 32.0*k1*μ^3*r2 +
32.0*k1*μ^4*r1 - 64.0*k2*μ^2*r2 + 64.0*k2*μ^3*r1 + 32.0*k1*μ^4*r2 + 32.0*k2*μ^2*r3 -
32.0*k2*μ^3*r2 + 32.0*k2*μ^4*r1 + 32.0*k2*μ^4*r2 + 128.0*μ^3*r1*r2 - 128.0*μ^4*r1*r2 +
2.0*k1*k2*r3 + 2.0*k1*k3*r3 + 2.0*k2*k3*r3 + 64.0*k2*μ*r3 + 64.0*k3*μ*r3 - 8.0*k2*r1*r3 -
32.0*k2*r2*r3 - 8.0*k3*r1*r3 - 32.0*k3*r2*r3 + 2.0*k1*k2*μ*r1 + 2.0*k1*k2*μ*r2 + 2.0*k1*k3*μ*r1
+ 2.0*k1*k3*μ*r2 + 2.0*k2*k3*μ*r1 + 2.0*k2*k3*μ*r2 + 4.0*k1*k2*r1*r3 - 2.0*k1*k2*r2*r3 +
4.0*k1*k3*r1*r3 - 2.0*k1*k3*r2*r3 + 4.0*k2*k3*r1*r3 - 2.0*k2*k3*r2*r3 - 8.0*k2*μ*r1*r2 -
8.0*k3*μ*r1*r2 + 8.0*k2*r1*r2*r3 + 8.0*k3*r1*r2*r3 - 64.0*k1*μ^3*r1*r2 - 32.0*k2*μ^2*r1*r2 +
32.0*k1*μ^4*r1*r2 + 16.0*k2*μ^2*r1*r3 - 64.0*k2*μ^3*r1*r2 - 8.0*k1*μ^4*r1*r3 -
32.0*k2*μ^2*r2*r3 + 32.0*k2*μ^4*r1*r2 - 8.0*k1*μ^4*r2*r3 - 8.0*k2*μ^4*r1*r3 - 8.0*k2*μ^4*r2*r3 +
32.0*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2 + 4.0*k1*k2*μ*r1*r3 + 2.0*k1*k3*μ*r1*r2 +
4.0*k1*k2*μ*r2*r3 + 4.0*k1*k3*μ*r1*r3 + 2.0*k2*k3*μ*r1*r2 + 4.0*k1*k3*μ*r2*r3 +
4.0*k2*k3*μ*r1*r3 + 4.0*k2*k3*μ*r2*r3 - 4.0*k1*k2*r1*r2*r3 - 4.0*k1*k3*r1*r2*r3 -
4.0*k2*k3*r1*r2*r3 - 16.0*k2*μ*r1*r2*r3 - 16.0*k3*μ*r1*r2*r3 - 16.0*k2*μ^2*r1*r2*r3 -
8.0*k1*μ^4*r1*r2*r3 - 8.0*k2*μ^4*r1*r2*r3 + 4.0*k1*k2*μ*r1*r2*r3 + 4.0*k1*k3*μ*r1*r2*r3 +
4.0*k2*k3*μ*r1*r2*r3)*(8.0*k2*μ + 10.0*k2*r3 - 8.0*k1*μ^2 + 8.0*k2*μ^2 + 8.0*k1*μ^4 +
8.0*k2*μ^3 + 16.0*k3*μ^2 + 8.0*k2*μ^4 + 16.0*μ^2*r1 + 32.0*μ^2*r2 - 16.0*μ^2*r3 - 32.0*μ^4*r1 -
128.0*μ^4*r2 - 32.0*μ^4*r3 - 64.0*μ^5*r3 - 32.0*μ^2 + 128.0*μ^4 + 160.0*μ^5 - 8.0*k1*μ^2*r1 +
8.0*k1*μ^2*r2 - 4.0*k1*μ^2*r3 + 16.0*k1*μ^4*r1 - 8.0*k2*μ^2*r2 + 4.0*k2*μ^3*r1 - 8.0*k1*μ^4*r2 +
10.0*k1*μ^5*r1 + 4.0*k2*μ^2*r3 - 8.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 - 2.0*k1*μ^4*r3 +
10.0*k1*μ^5*r2 - 8.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 32.0*k3*μ^2*r3 - 2.0*k2*μ^4*r3 +
10.0*k2*μ^5*r2 - 16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 + 16.0*μ^2*r2*r3 + 32.0*μ^4*r1*r2 +
8.0*μ^4*r1*r3 - 40.0*μ^5*r1*r2 + 32.0*μ^4*r2*r3 + k1*k2*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 +
16.0*k2*μ*r3 + 16.0*k3*μ*r3 - 4.0*k2*r1*r3 - 10.0*k2*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 +
k1*k3*μ*r3 + k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 - 2.0*k2*μ*r1*r2 - 4.0*k2*μ*r1*r3 +
4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 - 16.0*k3*μ*r2*r3 + 4.0*k2*r1*r2*r3 + k1*k3*μ^2*r1 +
k1*k3*μ^2*r2 + k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 +
4.0*k1*μ^2*r2*r3 - 16.0*k1*μ^4*r1*r2 - 4.0*k2*μ^3*r1*r2 - 4.0*k3*μ^2*r1*r2 - 4.0*k1*μ^4*r1*r3 +
10.0*k1*μ^5*r1*r2 - 4.0*k2*μ^2*r2*r3 - 16.0*k2*μ^4*r1*r2 + 2.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 -
4.0*k2*μ^4*r1*r3 + 10.0*k2*μ^5*r1*r2 - 4.0*k1*μ^5*r2*r3 + 2.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 -
4.0*k2*μ^5*r2*r3 - 8.0*μ^2*r1*r2*r3 - 8.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + k1*k2*μ*r1*r2 +
2.0*k1*k2*μ*r1*r3 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 - 1.0*k1*k3*μ*r2*r3 +
2.0*k2*k3*μ*r1*r3 - 1.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 4.0*k2*μ*r1*r2*r3 +
4.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 + 2.0*k1*k3*μ^2*r1*r3 + k2*k3*μ^2*r1*r2 +
2.0*k1*k3*μ^2*r2*r3 + 2.0*k2*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 +
4.0*k1*μ^4*r1*r2*r3 - 8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 + 4.0*k2*μ^4*r1*r2*r3 -

4.0*k2*μ^5*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 -
2.0*k1*k3*μ*r1*r2*r3 - 2.0*k2*k3*μ*r1*r2*r3))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 +
16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3
+ 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 -
32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 +
16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 +
k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 -
16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 +
2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 -
1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 -
32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 -
32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3
- 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 +
k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 +
2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 -
2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 -
4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 +
2.0*k2*k3*μ*r1*r2*r3)^2) - (0.0034722222222222222222222222222222*L^9*k1*q^2*(r1 -
1.0)*(16.0*k1*μ^3 - 4.0*k3*μ - 2.0*k2*r3 - 2.0*k3*r3 - 4.0*k2*μ - 20.0*k2*μ^2 + 16.0*k1*μ^4 -
24.0*k2*μ^3 + 8.0*k2*μ^4 - 8.0*k3*μ^3 + 4.0*k1*μ^6 + 20.0*k2*μ^5 + 4.0*k2*μ^6 + 32.0*μ^3*r2 +
32.0*μ^4*r2 + 16.0*μ^4*r3 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 32.0*μ^3 - 64.0*μ^4 - 32.0*μ^6 +
2.0*k1*k2*μ^2 - 1.0*k1*k2*μ^3 + k1*k3*μ^3 + 2.0*k2*k3*μ^2 + k2*k3*μ^3 - 16.0*k1*μ^3*r2 +
20.0*k2*μ^2*r2 - 10.0*k2*μ^2*r3 + 24.0*k2*μ^3*r2 - 4.0*k1*μ^4*r3 - 40.0*k2*μ^4*r2 +
8.0*k3*μ^3*r2 - 4.0*k1*μ^6*r2 - 2.0*k2*μ^4*r3 + 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 - 2.0*k1*μ^6*r3 -
8.0*k2*μ^5*r3 - 4.0*k2*μ^6*r2 - 2.0*k2*μ^6*r3 - 8.0*μ^4*r2*r3 - 16.0*μ^6*r2*r3 + k1*k2*μ +
k1*k3*μ - 1.0*k2*k3*μ + k1*k2*r3 + k1*k3*r3 - 2.0*k2*k3*r3 + 2.0*k2*μ*r2 - 8.0*k2*μ*r3 +
2.0*k3*μ*r2 - 8.0*k3*μ*r3 + 2.0*k2*r2*r3 + 2.0*k3*r2*r3 + 2.0*k1*k2*μ*r3 + 2.0*k1*k3*μ*r3 -
1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 1.0*k1*k2*r2*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r2*r3 +
4.0*k2*μ*r2*r3 + 4.0*k3*μ*r2*r3 - 2.0*k1*k2*μ^2*r2 + k1*k2*μ^2*r3 + k1*k2*μ^3*r2 -
1.0*k1*k3*μ^3*r2 + k2*k3*μ^2*r2 + k1*k3*μ^3*r3 + 4.0*k2*k3*μ^2*r3 - 1.0*k2*k3*μ^3*r2 +
k2*k3*μ^3*r3 + 10.0*k2*μ^2*r2*r3 + 10.0*k2*μ^4*r2*r3 + 8.0*k3*μ^3*r2*r3 + 2.0*k1*μ^6*r2*r3 -
4.0*k2*μ^5*r2*r3 + 2.0*k2*μ^6*r2*r3 - 5.0*k2*k3*μ*r2*r3 - 1.0*k1*k2*μ^2*r2*r3 -
1.0*k1*k3*μ^3*r2*r3 + 2.0*k2*k3*μ^2*r2*r3 - 1.0*k2*k3*μ^3*r2*r3)*(8.0*k2*μ^4 - 8.0*k3*μ -
6.0*k2*r3 - 6.0*k3*r3 - 24.0*k2*μ^2 - 8.0*k2*μ^3 - 8.0*k2*μ - 8.0*k3*μ^3 + 96.0*μ^3*r2 -
32.0*μ^4*r1 + 32.0*μ^4*r2 + 32.0*μ^4*r3 - 16.0*μ^6*r1 + 32.0*μ^6*r2 + 16.0*μ^6*r3 - 96.0*μ^3 -
128.0*μ^4 - 32.0*μ^6 - 16.0*k2*μ^2*r1 + 24.0*k2*μ^2*r2 - 28.0*k2*μ^3*r1 - 12.0*k2*μ^2*r3 +
8.0*k2*μ^3*r2 - 4.0*k3*μ^3*r1 - 24.0*k2*μ^4*r2 + 10.0*k2*μ^5*r1 + 8.0*k3*μ^3*r2 - 2.0*k2*μ^4*r3
+ 10.0*k2*μ^5*r2 - 8.0*k3*μ^3*r3 + 32.0*μ^4*r1*r2 + 8.0*μ^4*r1*r3 - 8.0*μ^4*r2*r3 +
16.0*μ^6*r1*r2 + 8.0*μ^6*r1*r3 - 16.0*μ^6*r2*r3 - 1.0*k2*k3*r3 - 2.0*k2*μ*r1 + 2.0*k2*μ*r2 -
2.0*k3*μ*r1 - 16.0*k2*μ*r3 + 2.0*k3*μ*r2 - 16.0*k3*μ*r3 + 6.0*k2*r2*r3 + 6.0*k3*r2*r3 -
1.0*k2*k3*μ*r1 - 1.0*k2*k3*μ*r2 + k2*k3*μ*r3 - 2.0*k2*k3*r1*r3 + k2*k3*r2*r3 + 2.0*k2*μ*r1*r2 -
4.0*k2*μ*r1*r3 + 2.0*k3*μ*r1*r2 + 4.0*k2*μ*r2*r3 - 4.0*k3*μ*r1*r3 + 4.0*k3*μ*r2*r3 +
k2*k3*μ^2*r1 + k2*k3*μ^2*r2 + 16.0*k2*μ^2*r1*r2 - 8.0*k2*μ^2*r1*r3 + 28.0*k2*μ^3*r1*r2 +
12.0*k2*μ^2*r2*r3 - 32.0*k2*μ^4*r1*r2 + 4.0*k3*μ^3*r1*r2 + 10.0*k2*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r3
+ 6.0*k2*μ^4*r2*r3 - 4.0*k2*μ^5*r1*r3 + 8.0*k3*μ^3*r2*r3 - 4.0*k2*μ^5*r2*r3 - 8.0*μ^4*r1*r2*r3 -
8.0*μ^6*r1*r2*r3 - 1.0*k2*k3*μ*r1*r2 - 3.0*k2*k3*μ*r2*r3 + 2.0*k2*k3*r1*r2*r3 +
4.0*k2*μ*r1*r2*r3 + 4.0*k3*μ*r1*r2*r3 + k2*k3*μ^2*r1*r2 + 2.0*k2*k3*μ^2*r1*r3 +

2.0*k2*k3*μ^2*r2*r3 + 8.0*k2*μ^2*r1*r2*r3 + 8.0*k2*μ^4*r1*r2*r3 + 4.0*k3*μ^3*r1*r2*r3 -
4.0*k2*μ^5*r1*r2*r3 + 2.0*k2*k3*μ^2*r1*r2*r3 - 4.0*k2*k3*μ*r1*r2*r3))/(EI^2*(16.0*k2*μ +
16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 -
256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 -
16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 +
16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 -
64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 -
16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 +
k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 -
1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 +
4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 +
8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2
- 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 +
2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 +
2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 -
2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 -
8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 +
2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2) -
(0.0034722222222222222222222222222222*L^9*k2*μ^3*q^2*(r2 - 1.0)*(4.0*k2 + 4.0*k3 + k1*k2 +
k1*k3 - 1.0*k2*k3 + 20.0*k2*μ - 2.0*k2*r1 - 2.0*k3*r1 + 8.0*k2*r3 + 8.0*k3*r3 + 16.0*k1*μ^3 -
16.0*k2*μ^2 - 16.0*k2*μ^3 + 16.0*k3*μ^2 + 4.0*k1*μ^5 + 20.0*k2*μ^4 + 4.0*k2*μ^5 - 32.0*μ^3*r1 -
16.0*μ^3*r3 - 16.0*μ^5*r1 - 32.0*μ^5*r3 + 64.0*μ^3 + 64.0*μ^5 - 1.0*k1*k2*μ^2 + k1*k3*μ^2 +
k2*k3*μ^2 + 16.0*k1*μ^3*r1 + 4.0*k2*μ^2*r1 - 4.0*k3*μ^2*r1 - 4.0*k1*μ^3*r3 + 8.0*k1*μ^5*r1 +
10.0*k2*μ^4*r1 + 4.0*k2*μ^3*r3 + 8.0*k2*μ^5*r1 + 16.0*k3*μ^2*r3 - 2.0*k1*μ^5*r3 - 8.0*k2*μ^4*r3
- 2.0*k2*μ^5*r3 + 8.0*μ^3*r1*r3 + 8.0*μ^5*r1*r3 + 2.0*k1*k2*μ + 2.0*k2*k3*μ + k1*k2*r1 +
k1*k3*r1 + 2.0*k1*k2*r3 + 2.0*k1*k3*r3 - 2.0*k2*k3*r3 - 8.0*k2*μ*r1 + 10.0*k2*μ*r3 - 4.0*k2*r1*r3
- 4.0*k3*r1*r3 + 4.0*k1*k2*μ*r1 + k1*k2*μ*r3 + k2*k3*μ*r1 + 4.0*k2*k3*μ*r3 + 2.0*k1*k2*r1*r3 +
2.0*k1*k3*r1*r3 - 4.0*k2*μ*r1*r3 - 2.0*k1*k2*μ^2*r1 + 2.0*k1*k3*μ^2*r1 + 2.0*k2*k3*μ^2*r1 +
k1*k3*μ^2*r3 + k2*k3*μ^2*r3 - 4.0*k1*μ^3*r1*r3 - 4.0*k3*μ^2*r1*r3 - 4.0*k1*μ^5*r1*r3 -
4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^5*r1*r3 + 2.0*k1*k2*μ*r1*r3 + 2.0*k2*k3*μ*r1*r3 +
2.0*k1*k3*μ^2*r1*r3 + 2.0*k2*k3*μ^2*r1*r3)*(8.0*k1*μ^4 - 10.0*k3*r3 - 8.0*k1*μ^2 - 16.0*k1*μ^3 -
8.0*k3*μ + 16.0*k3*μ^2 + 8.0*k3*μ^3 + 16.0*μ^2*r1 + 32.0*μ^2*r2 + 64.0*μ^3*r1 - 16.0*μ^2*r3 +
160.0*μ^3*r2 - 160.0*μ^4*r2 + 16.0*μ^6*r1 - 64.0*μ^5*r3 - 32.0*μ^6*r2 - 16.0*μ^6*r3 - 32.0*μ^2 -
160.0*μ^3 + 160.0*μ^5 + 32.0*μ^6 - 8.0*k1*μ^2*r1 + 8.0*k1*μ^2*r2 - 32.0*k1*μ^3*r1 -
4.0*k1*μ^2*r3 + 16.0*k1*μ^3*r2 - 24.0*k1*μ^4*r2 + 10.0*k1*μ^5*r1 + 4.0*k3*μ^3*r1 -
2.0*k1*μ^4*r3 + 10.0*k1*μ^5*r2 + 32.0*k3*μ^2*r3 - 8.0*k3*μ^3*r2 + 8.0*k3*μ^3*r3 -
16.0*μ^2*r1*r2 + 8.0*μ^2*r1*r3 - 64.0*μ^3*r1*r2 + 16.0*μ^2*r2*r3 + 64.0*μ^4*r1*r2 -
40.0*μ^5*r1*r2 + 40.0*μ^4*r2*r3 - 16.0*μ^6*r1*r2 - 8.0*μ^6*r1*r3 + 16.0*μ^6*r2*r3 - 1.0*k1*k3*r3
+ 2.0*k3*μ*r1 - 2.0*k3*μ*r2 + 4.0*k3*r1*r3 + 10.0*k3*r2*r3 - 1.0*k1*k3*μ*r1 - 1.0*k1*k3*μ*r2 +
k1*k3*μ*r3 - 2.0*k1*k3*r1*r3 + k1*k3*r2*r3 + 2.0*k3*μ*r1*r2 - 20.0*k3*μ*r2*r3 - 4.0*k3*r1*r2*r3 +
k1*k3*μ^2*r1 + k1*k3*μ^2*r2 + 8.0*k1*μ^2*r1*r2 - 4.0*k1*μ^2*r1*r3 + 32.0*k1*μ^3*r1*r2 +
4.0*k1*μ^2*r2*r3 - 32.0*k1*μ^4*r1*r2 - 4.0*k3*μ^2*r1*r2 + 10.0*k1*μ^5*r1*r2 - 4.0*k3*μ^3*r1*r2 +
6.0*k1*μ^4*r2*r3 - 4.0*k1*μ^5*r1*r3 + 4.0*k3*μ^3*r1*r3 - 4.0*k1*μ^5*r2*r3 - 8.0*k3*μ^3*r2*r3 -
8.0*μ^2*r1*r2*r3 - 16.0*μ^4*r1*r2*r3 + 16.0*μ^5*r1*r2*r3 + 8.0*μ^6*r1*r2*r3 - 1.0*k1*k3*μ*r1*r2 -
3.0*k1*k3*μ*r2*r3 + 2.0*k1*k3*r1*r2*r3 + 8.0*k3*μ*r1*r2*r3 + k1*k3*μ^2*r1*r2 +
2.0*k1*k3*μ^2*r1*r3 + 2.0*k1*k3*μ^2*r2*r3 + 4.0*k1*μ^2*r1*r2*r3 + 8.0*k1*μ^4*r1*r2*r3 -
8.0*k3*μ^2*r1*r2*r3 - 4.0*k1*μ^5*r1*r2*r3 - 4.0*k3*μ^3*r1*r2*r3 + 2.0*k1*k3*μ^2*r1*r2*r3 -

4.0*k1*k3*μ*r1*r2*r3))/(EI^2*(16.0*k2*μ + 16.0*k3*μ + 16.0*k2*r3 + 16.0*k3*r3 + 16.0*k1*μ^3 + 32.0*k2*μ^2 + 16.0*k2*μ^3 - 64.0*μ^3*r1 - 256.0*μ^3*r2 - 64.0*μ^4*r3 + 256.0*μ^3 + 256.0*μ^4 + 32.0*k1*μ^3*r1 + 16.0*k2*μ^2*r1 - 16.0*k1*μ^3*r2 + 16.0*k1*μ^4*r1 - 32.0*k2*μ^2*r2 + 32.0*k2*μ^3*r1 + 16.0*k1*μ^4*r2 + 16.0*k2*μ^2*r3 - 16.0*k2*μ^3*r2 + 16.0*k2*μ^4*r1 + 16.0*k2*μ^4*r2 + 64.0*μ^3*r1*r2 - 64.0*μ^4*r1*r2 + k1*k2*r3 + k1*k3*r3 + k2*k3*r3 + 32.0*k2*μ*r3 + 32.0*k3*μ*r3 - 4.0*k2*r1*r3 - 16.0*k2*r2*r3 - 4.0*k3*r1*r3 - 16.0*k3*r2*r3 + k1*k2*μ*r1 + k1*k2*μ*r2 + k1*k3*μ*r1 + k1*k3*μ*r2 + k2*k3*μ*r1 + k2*k3*μ*r2 + 2.0*k1*k2*r1*r3 - 1.0*k1*k2*r2*r3 + 2.0*k1*k3*r1*r3 - 1.0*k1*k3*r2*r3 + 2.0*k2*k3*r1*r3 - 1.0*k2*k3*r2*r3 - 4.0*k2*μ*r1*r2 - 4.0*k3*μ*r1*r2 + 4.0*k2*r1*r2*r3 + 4.0*k3*r1*r2*r3 - 32.0*k1*μ^3*r1*r2 - 16.0*k2*μ^2*r1*r2 + 16.0*k1*μ^4*r1*r2 + 8.0*k2*μ^2*r1*r3 - 32.0*k2*μ^3*r1*r2 - 4.0*k1*μ^4*r1*r3 - 16.0*k2*μ^2*r2*r3 + 16.0*k2*μ^4*r1*r2 - 4.0*k1*μ^4*r2*r3 - 4.0*k2*μ^4*r1*r3 - 4.0*k2*μ^4*r2*r3 + 16.0*μ^4*r1*r2*r3 + k1*k2*μ*r1*r2 + 2.0*k1*k2*μ*r1*r3 + k1*k3*μ*r1*r2 + 2.0*k1*k2*μ*r2*r3 + 2.0*k1*k3*μ*r1*r3 + k2*k3*μ*r1*r2 + 2.0*k1*k3*μ*r2*r3 + 2.0*k2*k3*μ*r1*r3 + 2.0*k2*k3*μ*r2*r3 - 2.0*k1*k2*r1*r2*r3 - 2.0*k1*k3*r1*r2*r3 - 2.0*k2*k3*r1*r2*r3 - 8.0*k2*μ*r1*r2*r3 - 8.0*k3*μ*r1*r2*r3 - 8.0*k2*μ^2*r1*r2*r3 - 4.0*k1*μ^4*r1*r2*r3 - 4.0*k2*μ^4*r1*r2*r3 + 2.0*k1*k2*μ*r1*r2*r3 + 2.0*k1*k3*μ*r1*r2*r3 + 2.0*k2*k3*μ*r1*r2*r3)^2)))^(1/2))/EI^(1/2)

$C_{dB\_2}(subs.in\ k, r_3 = r_1) =$

(4 μ^3 (16 (-7 + 4 r1) (-1 + r2) - 4 k1 (8 + 7 r1) (-1 + r2) - 4 μ (-4 + r1) (16 - 9 r2 + r1 (-9 + 5 r2)) + k1 μ (-4 + r1) (8 (-2 + r2) + r1 (-16 + 9 r2))) + k3 (-r1 (28 - 16 r1 + k1 (8 + 7 r1)) (-1 + r2) - μ (1 + 2 r1) (-64 + 4 r1 (9 - 5 r2) + 8 k1 (-2 + r2) + 36 r2 + k1 r1 (-16 + 9 r2))) + k2 (-8 μ^2 (2 + r1) (20 - 11 r1 + k1 (4 + 5 r1)) (-1 + r2) - 16 μ^3 (32 - 17 r1 + k1 (4 + 8 r1)) (-1 + r2) - r1 (28 + k3 (32 - 17 r1) - 16 r1 + k1 (8 + 7 r1 + k3 (4 + 8 r1))) (-1 + r2) - 4 μ^4 (-4 + r1) (4 (4 + (4 + k1) r2) + r1 (-8 - 9 r2 + 4 k1 (1 + r2))) + μ (1 + 2 r1) (4 k1 (4 + (-2 + k3) r2) + k1 r1 (16 - 9 r2 + 4 k3 (1 + r2)) + 4 (16 - 9 r2 + r1 (-9 + 5 r2)) + k3 (16 (1 + r2) - r1 (8 + 9 r2))))))/(4 (-4 μ^3 (-4 (-4 + r1) (-4 - 4 μ + 4 r2 + μ r1 r2) + k1 (-4 + 4 r2 - 4 μ r2 + μ r1^2 (1 + r2) - r1 (8 + 4 μ - 8 r2 + 3 μ r2))) + k3 (-r1 (16 + k1 - 4 r1 + 2 k1 r1) (-1 + r2) + μ (1 + 2 r1) (16 - 4 r1 r2 + k1 (r1 + r2 + r1 r2))) + k2 (-8 μ^2 (2 + r1)^2 (-1 + r2) - 16 μ^3 (1 + 2 r1) (-1 + r2) - r1 (16 + k1 + k3 - 4 r1 + 2 k1 r1 + 2 k3 r1) (-1 + r2) - 4 μ^4 (-4 + r1) (r1 + r2 + r1 r2) + μ (1 + 2 r1) (16 - 4 r1 r2 + k1 (r1 + r2 + r1 r2) + k3 (r1 + r2 + r1 r2)))))

Sensitivity analysis codes and results

```
# CdB2
from SALib.sample import saltelli
from SALib.analyze import sobol
import numpy as np
from SALib.plotting.bar import plot as barplot
import matplotlib.pyplot as plot
import math
#Define the model inputs
problem_CdB2 = {
    'num_vars': 7,
    'names': ['mu', 'r1', 'r2', 'r3', 'k1', 'k2','k3'],
    'bounds': [[0, 1], [0, 1], [0, 1], [0, 1], [0, 1], [0, 1], [0, 1]]}
def evaluate_CdB2(X):
    return np.array([(64*x[5]*x[0] + 64*x[6]*x[0] + 28*x[5]*x[3] + 28*x[6]*x[3] + 128*x[4]*x[0]**3 +
320*x[5]*x[0]**2 + 256*x[4]*x[0]**4 + 512*x[5]*x[0]**3 + 256*x[5]*x[0]**4 - 256*x[0]**3*x[1] -
448*x[0]**3*x[2] - 576*x[0]**4*x[1] - 576*x[0]**4*x[2] - 256*x[0]**4*x[3] + 448*x[0]**3 +
1024*x[0]**4 + 64*x[4]*x[5]*x[0]**2 + 64*x[4]*x[5]*x[0]**3 + 112*x[4]*x[0]**3*x[1] -
176*x[5]*x[0]**2*x[1] - 128*x[4]*x[0]**3*x[2] + 256*x[4]*x[0]**4*x[1] - 320*x[5]*x[0]**2*x[2] -
272*x[5]*x[0]**3*x[1] - 128*x[4]*x[0]**4*x[2] + 160*x[5]*x[0]**2*x[3] - 512*x[5]*x[0]**3*x[2] -
128*x[5]*x[0]**4*x[1] - 64*x[4]*x[0]**4*x[3] + 256*x[5]*x[0]**4*x[2] - 64*x[5]*x[0]**4*x[3] +
256*x[0]**3*x[1]*x[2] + 320*x[0]**4*x[1]*x[2] + 144*x[0]**4*x[1]*x[3] + 144*x[0]**4*x[2]*x[3] +
16*x[4]*x[5]*x[0] + 16*x[4]*x[6]*x[0] + 16*x[5]*x[6]*x[0] + 8*x[4]*x[5]*x[3] + 8*x[4]*x[6]*x[3] +
32*x[5]*x[6]*x[3] - 36*x[5]*x[0]*x[1] - 36*x[5]*x[0]*x[2] - 36*x[6]*x[0]*x[1] + 128*x[5]*x[0]*x[3] -
36*x[6]*x[0]*x[2] + 128*x[6]*x[0]*x[3] - 16*x[5]*x[1]*x[3] - 28*x[5]*x[2]*x[3] - 16*x[6]*x[1]*x[3] -
28*x[6]*x[2]*x[3] + 4*x[4]*x[5]*x[6]*x[3] + 16*x[4]*x[5]*x[0]*x[1] - 8*x[4]*x[5]*x[0]*x[2] +
16*x[4]*x[6]*x[0]*x[1] + 32*x[4]*x[5]*x[0]*x[3] - 8*x[4]*x[6]*x[0]*x[2] - 8*x[5]*x[6]*x[0]*x[1] +
32*x[4]*x[6]*x[0]*x[3] + 16*x[5]*x[6]*x[0]*x[2] + 32*x[5]*x[6]*x[0]*x[3] + 7*x[4]*x[5]*x[1]*x[3] -
8*x[4]*x[5]*x[2]*x[3] + 7*x[4]*x[6]*x[1]*x[3] - 8*x[4]*x[6]*x[2]*x[3] - 17*x[5]*x[6]*x[1]*x[3] -
32*x[5]*x[6]*x[2]*x[3] + 20*x[5]*x[0]*x[1]*x[2] - 72*x[5]*x[0]*x[1]*x[3] + 20*x[6]*x[0]*x[1]*x[2] -
72*x[5]*x[0]*x[2]*x[3] - 72*x[6]*x[0]*x[1]*x[3] - 72*x[6]*x[0]*x[2]*x[3] + 16*x[5]*x[1]*x[2]*x[3] +
16*x[6]*x[1]*x[2]*x[3] + 80*x[4]*x[5]*x[0]**2*x[1] - 64*x[4]*x[5]*x[0]**2*x[2] +
128*x[4]*x[5]*x[0]**3*x[1] + 32*x[4]*x[5]*x[0]**2*x[3] - 64*x[4]*x[5]*x[0]**3*x[2] +
64*x[4]*x[5]*x[0]**4*x[1] + 64*x[4]*x[5]*x[0]**4*x[2] - 112*x[4]*x[0]**3*x[1]*x[2] +
176*x[5]*x[0]**2*x[1]*x[2] - 144*x[4]*x[0]**4*x[1]*x[2] - 88*x[5]*x[0]**2*x[1]*x[3] +
272*x[5]*x[0]**3*x[1]*x[2] - 64*x[4]*x[0]**4*x[1]*x[3] - 160*x[5]*x[0]**2*x[2]*x[3] -
144*x[5]*x[0]**4*x[1]*x[2] + 32*x[4]*x[0]**4*x[2]*x[3] + 32*x[5]*x[0]**4*x[1]*x[3] -
64*x[5]*x[0]**4*x[2]*x[3] - 80*x[0]**4*x[1]*x[2]*x[3] + 4*x[4]*x[5]*x[6]*x[0]*x[1] +
4*x[4]*x[5]*x[6]*x[0]*x[2] + 8*x[4]*x[5]*x[6]*x[1]*x[3] - 4*x[4]*x[5]*x[6]*x[2]*x[3] -
9*x[4]*x[5]*x[0]*x[1]*x[2] + 32*x[4]*x[5]*x[0]*x[1]*x[3] - 9*x[4]*x[6]*x[0]*x[1]*x[2] -
16*x[4]*x[5]*x[0]*x[2]*x[3] + 32*x[4]*x[6]*x[0]*x[1]*x[3] - 9*x[5]*x[6]*x[0]*x[1]*x[2] -
16*x[4]*x[6]*x[0]*x[2]*x[3] - 16*x[5]*x[6]*x[0]*x[1]*x[3] + 32*x[5]*x[6]*x[0]*x[2]*x[3] -
```

```
7*x[4]*x[5]*x[1]*x[2]*x[3] - 7*x[4]*x[6]*x[1]*x[2]*x[3] + 17*x[5]*x[6]*x[1]*x[2]*x[3] +
40*x[5]*x[0]*x[1]*x[2]*x[3] + 40*x[6]*x[0]*x[1]*x[2]*x[3] - 80*x[4]*x[5]*x[0]**2*x[1]*x[2] +
40*x[4]*x[5]*x[0]**2*x[1]*x[3] - 128*x[4]*x[5]*x[0]**3*x[1]*x[2] - 32*x[4]*x[5]*x[0]**2*x[2]*x[3]
+ 64*x[4]*x[5]*x[0]**4*x[1]*x[2] - 16*x[4]*x[5]*x[0]**4*x[1]*x[3] - 16*x[4]*x[5]*x[0]**4*x[2]*x[3]
+ 88*x[5]*x[0]**2*x[1]*x[2]*x[3] + 36*x[4]*x[0]**4*x[1]*x[2]*x[3] +
36*x[5]*x[0]**4*x[1]*x[2]*x[3] - 40*x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] -
16*x[4]*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4*x[4]*x[5]*x[6]*x[0]*x[1]*x[2] +
8*x[4]*x[5]*x[6]*x[0]*x[1]*x[3] + 8*x[4]*x[5]*x[6]*x[0]*x[2]*x[3] - 8*x[4]*x[5]*x[6]*x[1]*x[2]*x[3]
- 18*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 18*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
18*x[5]*x[6]*x[0]*x[1]*x[2]*x[3] + 8*x[4]*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])/(4*(16*x[5]*x[0] +
16*x[6]*x[0] + 16*x[5]*x[3] + 16*x[6]*x[3] + 16*x[4]*x[0]**3 + 32*x[5]*x[0]**2 + 16*x[5]*x[0]**3 -
64*x[0]**3*x[1] - 256*x[0]**3*x[2] - 64*x[0]**4*x[3] + 256*x[0]**3 + 256*x[0]**4 +
32*x[4]*x[0]**3*x[1] + 16*x[5]*x[0]**2*x[1] - 16*x[4]*x[0]**3*x[2] + 16*x[4]*x[0]**4*x[1] -
32*x[5]*x[0]**2*x[2] + 32*x[5]*x[0]**3*x[1] + 16*x[4]*x[0]**4*x[2] + 16*x[5]*x[0]**2*x[3] -
16*x[5]*x[0]**3*x[2] + 16*x[5]*x[0]**4*x[1] + 16*x[5]*x[0]**4*x[2] + 64*x[0]**3*x[1]*x[2] -
64*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32*x[5]*x[0]*x[3] +
32*x[6]*x[0]*x[3] - 4*x[5]*x[1]*x[3] - 16*x[5]*x[2]*x[3] - 4*x[6]*x[1]*x[3] - 16*x[6]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] +
x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2*x[4]*x[5]*x[1]*x[3] - x[4]*x[5]*x[2]*x[3] +
2*x[4]*x[6]*x[1]*x[3] - x[4]*x[6]*x[2]*x[3] + 2*x[5]*x[6]*x[1]*x[3] - x[5]*x[6]*x[2]*x[3] -
4*x[5]*x[0]*x[1]*x[2] - 4*x[6]*x[0]*x[1]*x[2] + 4*x[5]*x[1]*x[2]*x[3] + 4*x[6]*x[1]*x[2]*x[3] -
32*x[4]*x[0]**3*x[1]*x[2] - 16*x[5]*x[0]**2*x[1]*x[2] + 16*x[4]*x[0]**4*x[1]*x[2] +
8*x[5]*x[0]**2*x[1]*x[3] - 32*x[5]*x[0]**3*x[1]*x[2] - 4*x[4]*x[0]**4*x[1]*x[3] -
16*x[5]*x[0]**2*x[2]*x[3] + 16*x[5]*x[0]**4*x[1]*x[2] - 4*x[4]*x[0]**4*x[2]*x[3] -
4*x[5]*x[0]**4*x[1]*x[3] - 4*x[5]*x[0]**4*x[2]*x[3] + 16*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2*x[4]*x[5]*x[0]*x[2]*x[3] + 2*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2*x[4]*x[6]*x[0]*x[2]*x[3] + 2*x[5]*x[6]*x[0]*x[1]*x[3] + 2*x[5]*x[6]*x[0]*x[2]*x[3] -
2*x[4]*x[5]*x[1]*x[2]*x[3] - 2*x[4]*x[6]*x[1]*x[2]*x[3] - 2*x[5]*x[6]*x[1]*x[2]*x[3] -
8*x[5]*x[0]*x[1]*x[2]*x[3] - 8*x[6]*x[0]*x[1]*x[2]*x[3] - 8*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
2*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])) for x in X])

# Generate samples

param_values_CdB2 = saltelli.sample(problem_CdB2, 4096)

# Run model

Y_CdB2 = evaluate_CdB2(param_values_CdB2)

# Generate the dimension of sample data

print(param_values_CdB2.shape, Y_CdB2.shape)

# Perform analysis

Si_CdB2 = sobol.analyze(problem_CdB2, Y_CdB2, print_to_console=True)

# Print the first-order, second-order, total-order sensitivity indices

print()

# Generate sensitivity plot
```

Si_df_CdB2 = Si_CdB2.to_df()

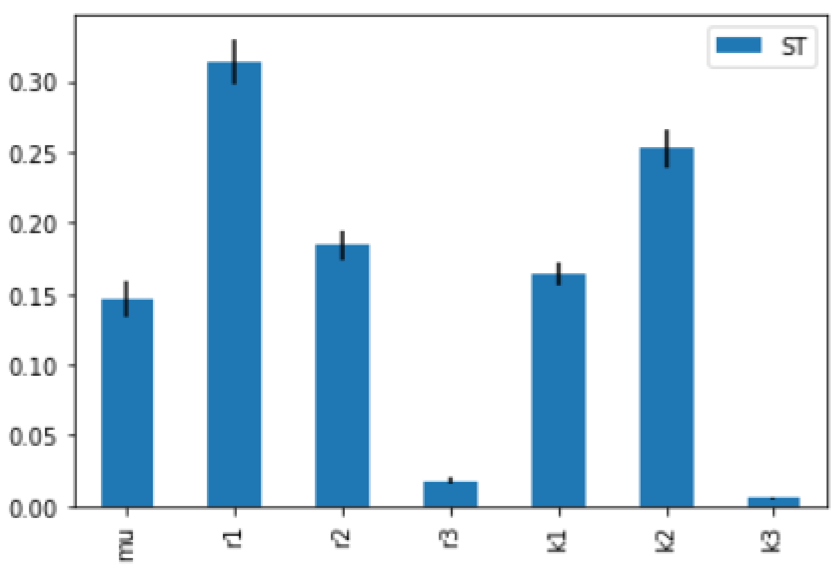
barplot(Si_df_CdB2[0])

plot.show()

	ST	ST_conf
mu	0.146601	0.012901
r1	0.314118	0.015970
r2	0.183917	0.010487
r3	0.017500	0.002472
k1	0.163445	0.008345
k2	0.252903	0.013334
k3	0.005918	0.000872

	S1	S1_conf
mu	0.102010	0.014511
r1	0.279641	0.021940
r2	0.157406	0.019993
r3	0.003897	0.006156
k1	0.152080	0.017160
k2	0.214722	0.020774
k3	0.000470	0.002833

	S2	S2_conf
(mu, r1)	0.004483	0.019485
(mu, r2)	0.007522	0.020342
(mu, r3)	0.006735	0.019870
(mu, k1)	-0.000408	0.020418
(mu, k2)	0.020382	0.020304
(mu, k3)	0.002635	0.019603
(r1, r2)	0.012044	0.034343
(r1, r3)	0.002957	0.031594
(r1, k1)	0.012106	0.033223
(r1, k2)	0.014015	0.035317
(r1, k3)	0.003212	0.031704
(r2, r3)	0.003702	0.024634
(r2, k1)	0.003996	0.026649
(r2, k2)	0.003580	0.026753
(r2, k3)	0.002612	0.024665
(r3, k1)	0.003058	0.008197
(r3, k2)	0.003679	0.008217
(r3, k3)	0.001723	0.008320
(k1, k2)	-0.000106	0.024527
(k1, k3)	-0.001884	0.022162
(k2, k3)	0.004667	0.029125



```
# CfB2
from SALib.sample import saltelli
from SALib.analyze import sobol
import numpy as np
from SALib.plotting.bar import plot as barplot
import matplotlib.pyplot as plot
import math
#Define the model inputs
problem_CfB2 = {
    'num_vars': 7,
    'names': ['mu', 'r1', 'r2', 'r3', 'k1', 'k2','k3'],
    'bounds': [[0, 1], [0, 1], [0, 1], [0, 1], [0, 1], [0, 1], [0, 1]]}
def evaluate_CfB2(X):
    return np.array([((((0.0000053468657576480924539921890863214*(256.0*x[5]*x[0] +
256.0*x[6]*x[0] + 96.0*x[5]*x[3] + 96.0*x[6]*x[3] + 1536.0*x[4]*x[0]**3 + 1792.0*x[5]*x[0]**2 +
2560.0*x[4]*x[0]**4 + 3776.0*x[5]*x[0]**3 + 960.0*x[5]*x[0]**4 + 960.0*x[6]*x[0]**3 +
560.0*x[4]*x[0]**6 - 3120.0*x[5]*x[0]**5 - 2240.0*x[5]*x[0]**6 + 1280.0*x[6]*x[0]**5 -
400.0*x[5]*x[0]**7 + 1040.0*x[6]*x[0]**6 - 1024.0*x[0]**3*x[1] - 1536.0*x[0]**3*x[2] -
2560.0*x[0]**4*x[1] - 2560.0*x[0]**4*x[2] - 1024.0*x[0]**4*x[3] - 1120.0*x[0]**6*x[1] -
2240.0*x[0]**6*x[2] - 1440.0*x[0]**6*x[3] - 320.0*x[0]**8*x[3] - 320.0*x[0]**9*x[3] +
1536.0*x[0]**3 + 4096.0*x[0]**4 + 2240.0*x[0]**6 + 320.0*x[0]**9 + 640.0*x[4]*x[5]*x[0]**2 +
560.0*x[4]*x[5]*x[0]**3 - 160.0*x[4]*x[5]*x[0]**4 + 240.0*x[4]*x[6]*x[0]**3 -
320.0*x[5]*x[6]*x[0]**2 - 80.0*x[4]*x[5]*x[0]**5 - 240.0*x[5]*x[6]*x[0]**3 +
80.0*x[4]*x[6]*x[0]**5 + 160.0*x[5]*x[6]*x[0]**4 + 80.0*x[5]*x[6]*x[0]**5 +
512.0*x[4]*x[0]**3*x[1] - 1024.0*x[5]*x[0]**2*x[1] - 1536.0*x[4]*x[0]**3*x[2] +
1536.0*x[4]*x[0]**4*x[1] - 1792.0*x[5]*x[0]**2*x[2] - 1888.0*x[5]*x[0]**3*x[1] -
1024.0*x[4]*x[0]**4*x[2] + 896.0*x[5]*x[0]**2*x[3] - 3776.0*x[5]*x[0]**3*x[2] -
384.0*x[5]*x[0]**4*x[1] - 480.0*x[6]*x[0]**3*x[1] - 640.0*x[4]*x[0]**4*x[3] +
560.0*x[4]*x[0]**6*x[1] + 160.0*x[5]*x[0]**3*x[3] + 3136.0*x[5]*x[0]**4*x[2] +
780.0*x[5]*x[0]**5*x[1] - 960.0*x[6]*x[0]**3*x[2] - 560.0*x[4]*x[0]**6*x[2] +
260.0*x[5]*x[0]**4*x[3] + 820.0*x[5]*x[0]**5*x[2] + 800.0*x[6]*x[0]**3*x[3] -
320.0*x[6]*x[0]**5*x[1] - 360.0*x[4]*x[0]**6*x[3] + 1360.0*x[5]*x[0]**5*x[3] +
560.0*x[5]*x[0]**6*x[2] - 200.0*x[5]*x[0]**7*x[1] - 1280.0*x[6]*x[0]**5*x[2] +
20.0*x[4]*x[0]**9*x[1] + 840.0*x[5]*x[0]**6*x[3] + 400.0*x[5]*x[0]**7*x[2] +
400.0*x[6]*x[0]**5*x[3] - 20.0*x[4]*x[0]**8*x[3] + 20.0*x[4]*x[0]**9*x[2] + 80.0*x[5]*x[0]**7*x[3]
+ 20.0*x[5]*x[0]**9*x[1] + 160.0*x[6]*x[0]**6*x[3] - 20.0*x[5]*x[0]**8*x[3] +
20.0*x[5]*x[0]**9*x[2] + 1024.0*x[0]**3*x[1]*x[2] + 1536.0*x[0]**4*x[1]*x[2] +
640.0*x[0]**4*x[1]*x[3] + 640.0*x[0]**4*x[2]*x[3] + 1120.0*x[0]**6*x[1]*x[2] +
720.0*x[0]**6*x[1]*x[3] + 1440.0*x[0]**6*x[2]*x[3] + 80.0*x[0]**8*x[1]*x[3] -
80.0*x[0]**9*x[1]*x[2] + 320.0*x[0]**8*x[2]*x[3] + 160.0*x[4]*x[5]*x[0] + 160.0*x[4]*x[6]*x[0] +
160.0*x[5]*x[6]*x[0] + 96.0*x[4]*x[5]*x[3] + 96.0*x[4]*x[6]*x[3] + 256.0*x[5]*x[6]*x[3] -
160.0*x[5]*x[0]*x[1] - 160.0*x[5]*x[0]*x[2] - 160.0*x[6]*x[0]*x[1] + 512.0*x[5]*x[0]*x[3] -
160.0*x[6]*x[0]*x[2] + 512.0*x[6]*x[0]*x[3] - 64.0*x[5]*x[1]*x[3] - 96.0*x[5]*x[2]*x[3] -
```

64.0*x[6]*x[1]*x[3] - 96.0*x[6]*x[2]*x[3] + 40.0*x[4]*x[5]*x[6]*x[3] + 96.0*x[4]*x[5]*x[0]*x[1] - 64.0*x[4]*x[5]*x[0]*x[2] + 96.0*x[4]*x[6]*x[0]*x[1] + 320.0*x[4]*x[5]*x[0]*x[3] - 64.0*x[4]*x[6]*x[0]*x[2] - 64.0*x[5]*x[6]*x[0]*x[1] + 320.0*x[4]*x[6]*x[0]*x[3] + 96.0*x[5]*x[6]*x[0]*x[2] - 80.0*x[5]*x[6]*x[0]*x[3] + 32.0*x[4]*x[5]*x[1]*x[3] - 96.0*x[4]*x[5]*x[2]*x[3] + 32.0*x[4]*x[6]*x[1]*x[3] - 96.0*x[4]*x[6]*x[2]*x[3] - 128.0*x[5]*x[6]*x[1]*x[3] - 256.0*x[5]*x[6]*x[2]*x[3] + 96.0*x[5]*x[0]*x[1]*x[2] - 320.0*x[5]*x[0]*x[1]*x[3] + 96.0*x[6]*x[0]*x[1]*x[2] - 320.0*x[5]*x[0]*x[2]*x[3] - 320.0*x[6]*x[0]*x[1]*x[3] - 320.0*x[6]*x[0]*x[2]*x[3] + 64.0*x[5]*x[1]*x[2]*x[3] + 64.0*x[6]*x[1]*x[2]*x[3] + 640.0*x[4]*x[5]*x[0]**2*x[1] - 640.0*x[4]*x[5]*x[0]**2*x[2] + 1200.0*x[4]*x[5]*x[0]**3*x[1] + 320.0*x[4]*x[5]*x[0]**2*x[3] - 560.0*x[4]*x[5]*x[0]**3*x[2] + 320.0*x[4]*x[5]*x[0]**4*x[1] + 240.0*x[4]*x[6]*x[0]**3*x[1] + 80.0*x[5]*x[6]*x[0]**2*x[1] + 40.0*x[4]*x[5]*x[0]**3*x[3] + 800.0*x[4]*x[5]*x[0]**4*x[2] - 390.0*x[4]*x[5]*x[0]**5*x[1] - 240.0*x[4]*x[6]*x[0]**3*x[2] - 80.0*x[5]*x[6]*x[0]**2*x[2] + 90.0*x[4]*x[5]*x[0]**4*x[3] - 150.0*x[4]*x[5]*x[0]**5*x[2] - 105.0*x[4]*x[5]*x[0]**6*x[1] + 200.0*x[4]*x[6]*x[0]**3*x[3] + 160.0*x[4]*x[6]*x[0]**5*x[1] - 640.0*x[5]*x[6]*x[0]**2*x[3] + 240.0*x[5]*x[6]*x[0]**3*x[2] + 80.0*x[5]*x[6]*x[0]**4*x[1] + 15.0*x[4]*x[5]*x[0]**5*x[3] - 105.0*x[4]*x[5]*x[0]**6*x[2] - 80.0*x[4]*x[6]*x[0]**5*x[2] + 65.0*x[4]*x[6]*x[0]**6*x[1] - 200.0*x[5]*x[6]*x[0]**3*x[3] - 160.0*x[5]*x[6]*x[0]**4*x[2] + 160.0*x[5]*x[6]*x[0]**5*x[1] + 25.0*x[4]*x[6]*x[0]**5*x[3] + 65.0*x[4]*x[6]*x[0]**6*x[2] + 80.0*x[5]*x[6]*x[0]**4*x[3] - 80.0*x[5]*x[6]*x[0]**5*x[2] + 65.0*x[5]*x[6]*x[0]**6*x[1] + 25.0*x[5]*x[6]*x[0]**5*x[3] + 65.0*x[5]*x[6]*x[0]**6*x[2] - 512.0*x[4]*x[0]**3*x[1]*x[2] + 1024.0*x[5]*x[0]**2*x[1]*x[2] - 1024.0*x[4]*x[0]**4*x[1]*x[2] - 512.0*x[5]*x[0]**2*x[1]*x[3] + 1888.0*x[5]*x[0]**3*x[1]*x[2] - 384.0*x[4]*x[0]**4*x[1]*x[3] - 896.0*x[5]*x[0]**2*x[2]*x[3] - 80.0*x[5]*x[0]**3*x[1]*x[3] - 1664.0*x[5]*x[0]**4*x[1]*x[2] + 480.0*x[6]*x[0]**3*x[1]*x[2] + 256.0*x[4]*x[0]**4*x[2]*x[3] - 560.0*x[4]*x[0]**6*x[1]*x[2] - 160.0*x[5]*x[0]**3*x[2]*x[3] - 104.0*x[5]*x[0]**4*x[1]*x[3] + 140.0*x[5]*x[0]**5*x[1]*x[2] - 400.0*x[6]*x[0]**3*x[1]*x[3] - 360.0*x[4]*x[0]**6*x[1]*x[3] - 1284.0*x[5]*x[0]**4*x[2]*x[3] - 340.0*x[5]*x[0]**5*x[1]*x[3] + 420.0*x[5]*x[0]**6*x[1]*x[2] - 800.0*x[6]*x[0]**3*x[2]*x[3] + 320.0*x[6]*x[0]**5*x[1]*x[2] + 360.0*x[4]*x[0]**6*x[2]*x[3] + 40.0*x[5]*x[0]**5*x[2]*x[3] + 200.0*x[5]*x[0]**7*x[1]*x[2] - 100.0*x[6]*x[0]**5*x[1]*x[3] - 260.0*x[6]*x[0]**6*x[1]*x[2] - 40.0*x[4]*x[0]**8*x[1]*x[3] + 20.0*x[4]*x[0]**9*x[1]*x[2] - 360.0*x[5]*x[0]**6*x[2]*x[3] + 40.0*x[5]*x[0]**7*x[1]*x[3] - 400.0*x[6]*x[0]**5*x[2]*x[3] + 20.0*x[4]*x[0]**8*x[2]*x[3] - 20.0*x[4]*x[0]**9*x[1]*x[3] - 80.0*x[5]*x[0]**7*x[2]*x[3] - 40.0*x[5]*x[0]**8*x[1]*x[3] + 20.0*x[5]*x[0]**9*x[1]*x[2] - 20.0*x[4]*x[0]**9*x[2]*x[3] + 20.0*x[5]*x[0]**8*x[2]*x[3] - 20.0*x[5]*x[0]**9*x[1]*x[3] - 20.0*x[5]*x[0]**9*x[2]*x[3] - 384.0*x[0]**4*x[1]*x[2]*x[3] - 720.0*x[0]**6*x[1]*x[2]*x[3] - 80.0*x[0]**8*x[1]*x[2]*x[3] + 80.0*x[0]**9*x[1]*x[2]*x[3] + 40.0*x[4]*x[5]*x[6]*x[0]*x[1] + 40.0*x[4]*x[5]*x[6]*x[0]*x[2] - 40.0*x[4]*x[5]*x[6]*x[0]*x[3] + 80.0*x[4]*x[5]*x[6]*x[1]*x[3] - 40.0*x[4]*x[5]*x[6]*x[2]*x[3] - 64.0*x[4]*x[5]*x[0]*x[1]*x[2] + 192.0*x[4]*x[5]*x[0]*x[1]*x[3] - 64.0*x[4]*x[6]*x[0]*x[1]*x[2] - 128.0*x[4]*x[5]*x[0]*x[2]*x[3] + 192.0*x[4]*x[6]*x[0]*x[1]*x[3] - 64.0*x[5]*x[6]*x[0]*x[1]*x[2] - 128.0*x[4]*x[6]*x[0]*x[2]*x[3] + 32.0*x[5]*x[6]*x[0]*x[1]*x[3] + 592.0*x[5]*x[6]*x[0]*x[2]*x[3] - 32.0*x[4]*x[5]*x[1]*x[2]*x[3] - 32.0*x[4]*x[6]*x[1]*x[2]*x[3] + 128.0*x[5]*x[6]*x[1]*x[2]*x[3] + 192.0*x[5]*x[0]*x[1]*x[2]*x[3] + 192.0*x[6]*x[0]*x[1]*x[2]*x[3] - 40.0*x[4]*x[5]*x[6]*x[0]**2*x[1]-40.0*x[4]*x[5]*x[6]*x[0]**2*x[2] -640.0*x[4]*x[5]*x[0]**2*x[1]*x[2]+320.0*x[4]*x[5]*x[0]**2*x[1]*x[3]- 1200.0*x[4]*x[5]*x[0]**3*x[1]*x[2]-320.0*x[4]*x[5]*x[0]**2*x[2]*x[3]+ 40.0*x[4]*x[5]*x[0]**3*x[1]*x[3] + 960.0*x[4]*x[5]*x[0]**4*x[1]*x[2] - 240.0*x[4]*x[6]*x[0]**3*x[1]*x[2] + 80.0*x[5]*x[6]*x[0]**2*x[1]*x[2] - 40.0*x[4]*x[5]*x[0]**3*x[2]*x[3] + 20.0*x[4]*x[5]*x[0]**4*x[1]*x[3] - 70.0*x[4]*x[5]*x[0]**5*x[1]*x[2] + 200.0*x[4]*x[6]*x[0]**3*x[1]*x[3] + 160.0*x[5]*x[6]*x[0]**2*x[1]*x[3] - 250.0*x[4]*x[5]*x[0]**4*x[2]*x[3] + 170.0*x[4]*x[5]*x[0]**5*x[1]*x[3] - 105.0*x[4]*x[5]*x[0]**6*x[1]*x[2] - 200.0*x[4]*x[6]*x[0]**3*x[2]*x[3] - 160.0*x[4]*x[6]*x[0]**5*x[1]*x[2] -

160.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 80.0*x[5]*x[6]*x[0]**4*x[1]*x[2] +
125.0*x[4]*x[5]*x[0]**5*x[2]*x[3] + 30.0*x[4]*x[5]*x[0]**6*x[1]*x[3] +
50.0*x[4]*x[6]*x[0]**5*x[1]*x[3] + 65.0*x[4]*x[6]*x[0]**6*x[1]*x[2] +
200.0*x[5]*x[6]*x[0]**3*x[2]*x[3] + 40.0*x[5]*x[6]*x[0]**4*x[1]*x[3] -
160.0*x[5]*x[6]*x[0]**5*x[1]*x[2] + 30.0*x[4]*x[5]*x[0]**6*x[2]*x[3] -
25.0*x[4]*x[6]*x[0]**5*x[2]*x[3] + 10.0*x[4]*x[6]*x[0]**6*x[1]*x[3] -
80.0*x[5]*x[6]*x[0]**4*x[2]*x[3] + 50.0*x[5]*x[6]*x[0]**5*x[1]*x[3] +
65.0*x[5]*x[6]*x[0]**6*x[1]*x[2] + 10.0*x[4]*x[6]*x[0]**6*x[2]*x[3] -
25.0*x[5]*x[6]*x[0]**5*x[2]*x[3] + 10.0*x[5]*x[6]*x[0]**6*x[1]*x[3] +
10.0*x[5]*x[6]*x[0]**6*x[2]*x[3] + 512.0*x[5]*x[0]**2*x[1]*x[2]*x[3] +
256.0*x[4]*x[0]**4*x[1]*x[2]*x[3] + 80.0*x[5]*x[0]**3*x[1]*x[2]*x[3] +
616.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 400.0*x[6]*x[0]**3*x[1]*x[2]*x[3] +
360.0*x[4]*x[0]**6*x[1]*x[2]*x[3] - 220.0*x[5]*x[0]**5*x[1]*x[2]*x[3] -
120.0*x[5]*x[0]**6*x[1]*x[2]*x[3] + 100.0*x[6]*x[0]**5*x[1]*x[2]*x[3] +
40.0*x[4]*x[0]**8*x[1]*x[2]*x[3] - 40.0*x[5]*x[0]**7*x[1]*x[2]*x[3] -
40.0*x[6]*x[0]**6*x[1]*x[2]*x[3] - 20.0*x[4]*x[0]**9*x[1]*x[2]*x[3] +
40.0*x[5]*x[0]**8*x[1]*x[2]*x[3] - 20.0*x[5]*x[0]**9*x[1]*x[2]*x[3] -
40.0*x[4]*x[5]*x[6]*x[0]**2*x[1]*x[2] - 80.0*x[4]*x[5]*x[6]*x[0]**2*x[1]*x[3] -
80.0*x[4]*x[5]*x[6]*x[0]**2*x[2]*x[3] - 320.0*x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] -
40.0*x[4]*x[5]*x[0]**3*x[1]*x[2]*x[3] - 340.0*x[4]*x[5]*x[0]**4*x[1]*x[2]*x[3] -
200.0*x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] + 160.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +
110.0*x[4]*x[5]*x[0]**5*x[1]*x[2]*x[3] + 30.0*x[4]*x[5]*x[0]**6*x[1]*x[2]*x[3] -
50.0*x[4]*x[6]*x[0]**5*x[1]*x[2]*x[3] - 40.0*x[5]*x[6]*x[0]**4*x[1]*x[2]*x[3] +
10.0*x[4]*x[6]*x[0]**6*x[1]*x[2]*x[3] - 50.0*x[5]*x[6]*x[0]**5*x[1]*x[2]*x[3] +
10.0*x[5]*x[6]*x[0]**6*x[1]*x[2]*x[3] + 40.0*x[4]*x[5]*x[6]*x[0]*x[1]*x[2] +
120.0*x[4]*x[5]*x[6]*x[0]*x[2]*x[3] - 80.0*x[4]*x[5]*x[6]*x[1]*x[2]*x[3] -
128.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 128.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
288.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3] + 160.0*x[4]*x[5]*x[6]*x[0]*x[1]*x[2]*x[3] -
80.0*x[4]*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3]))/(((0.000192901234567901234567901234567901234567 9) +
(0.000192901234567901234567901234567 9*x[0]**9) +
(0.0000620039682539682539682539682539683968*(64.0*(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3]
- 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 +
96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] +
32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 -
16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] +
8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] +
8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] +
32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] +
8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] +
2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] -
16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] -
1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] +
2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] +
4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]
+ x[5]*x[6]*x[0]**2*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] +
28.0*x[5]*x[0]**3*x[1]*x[2] + 12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] +
4.0*x[6]*x[0]**3*x[1]*x[2] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] +
6.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] -
1.0*x[5]*x[6]*x[0]*x[1]*x[2] - 3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] +
4.0*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] +

2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] +
8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2 -
24.0*x[1]*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 +
64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] - 512.0*x[0]**3*x[2] -
128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1] +
32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2] +
64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] - 32.0*x[5]*x[0]**3*x[2]
+ 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] + 128.0*x[0]**3*x[1]*x[2] -
128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] + 2.0*x[5]*x[6]*x[3] +
64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] - 32.0*x[5]*x[2]*x[3] -
8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] + 2.0*x[4]*x[5]*x[0]*x[2] +
2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] + 2.0*x[5]*x[6]*x[0]*x[1] +
2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] - 2.0*x[4]*x[5]*x[2]*x[3] +
4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] + 4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3]
- 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2] + 8.0*x[5]*x[1]*x[2]*x[3] +
8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] - 32.0*x[5]*x[0]**2*x[1]*x[2] +
32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] - 64.0*x[5]*x[0]**3*x[1]*x[2] -
8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] + 32.0*x[5]*x[0]**4*x[1]*x[2] -
8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] - 8.0*x[5]*x[0]**4*x[2]*x[3] +
32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[1]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[3] +
4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] - 16.0*x[6]*x[0]*x[1]*x[2]*x[3] -
16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(16.0*x[4]*x[0]**3 -
4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 -
24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 +
4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] +
16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 -
1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 -
16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] -
4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] -
2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] -
8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] -
16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] -
8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] +
2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] +
4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] +
x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] +
x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] +
x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] +
8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] +
2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] -
1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] -
1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +

16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -
16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] +
x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] -
1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2]
+ 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] -
16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] -
32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] +
16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] -
2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) -
(0.0001085069444444444444444444444444*(16.0*x[4]*(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] -
6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] -
8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] -
16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 -
32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] -
12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] +
10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] -
8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] +
16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] -
2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] +
2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] -
1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] +
x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] +
4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]
+ x[5]*x[6]*x[0]**2*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] +
28.0*x[5]*x[0]**3*x[1]*x[2] + 12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] +
4.0*x[6]*x[0]**3*x[1]*x[2] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] +
6.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] -
1.0*x[5]*x[6]*x[0]*x[1]*x[2] - 3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] +
4.0*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] +
8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2 + 192.0*x[1]*(x[1] -
1.0)*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] -
20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 +
4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] +

16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])**2))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) + (0.0000868055555555555555555555555555556*(144.0*x[1]**2*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] -

8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])**2 - 2.0*x[4]*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 + 64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] - 512.0*x[0]**3*x[2] - 128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1] + 32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2] + 64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] - 32.0*x[5]*x[0]**3*x[2] + 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] + 128.0*x[0]**3*x[1]*x[2] - 128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] + 2.0*x[5]*x[6]*x[3] + 64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] - 32.0*x[5]*x[2]*x[3] - 8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] + 2.0*x[4]*x[5]*x[0]*x[2] + 2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] + 2.0*x[5]*x[6]*x[0]*x[1] + 2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] - 2.0*x[4]*x[5]*x[2]*x[3] + 4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] + 4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2] + 8.0*x[5]*x[1]*x[2]*x[3] + 8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] - 32.0*x[5]*x[0]**2*x[1]*x[2] + 32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] - 64.0*x[5]*x[0]**3*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] + 32.0*x[5]*x[0]**4*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] - 8.0*x[5]*x[0]**4*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[3] + 4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] - 16.0*x[6]*x[0]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3]-8.0*x[5]*x[0]**4*x[1]*x[2]*x[3]+ 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[0]*x[2]*x[3] -

4.0*x[6]*x[0]*x[1]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] + 28.0*x[5]*x[0]**3*x[1]*x[2] + 12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] + 4.0*x[6]*x[0]**3*x[1]*x[2] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] + 6.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[2] - 3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]) + 128.0*(x[1] - 1.0)*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])*(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] + 28.0*x[5]*x[0]**3*x[1]*x[2] + 12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] + 4.0*x[6]*x[0]**3*x[1]*x[2] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] + 6.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -

8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[2] -
3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] +
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +
16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -
16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] +
x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] -
1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2]
+ 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] -
16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] -
32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] +
16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] -
2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) +
(0.00014467592592592592592592592593*x[0]**3 *(64.0*x[0]**2*(x[2] - 1.0)**2*(4.0*x[5] +
4.0*x[6] + x[4]*x[5] + x[4]*x[6] - 1.0*x[5]*x[6] + 20.0*x[5]*x[0] - 2.0*x[5]*x[1] - 2.0*x[6]*x[1] +
8.0*x[5]*x[3] + 8.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 - 16.0*x[5]*x[0]**2 - 16.0*x[5]*x[0]**3 +
16.0*x[6]*x[0]**2 + 4.0*x[4]*x[0]**5 + 20.0*x[5]*x[0]**4 + 4.0*x[5]*x[0]**5 - 32.0*x[0]**3*x[1] -
16.0*x[0]**3*x[3] - 16.0*x[0]**5*x[1] - 32.0*x[0]**5*x[3] + 64.0*x[0]**3 + 64.0*x[0]**5 -
1.0*x[4]*x[5]*x[0]**2 + x[4]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**2 + 16.0*x[4]*x[0]**3*x[1] +
4.0*x[5]*x[0]**2*x[1] - 4.0*x[6]*x[0]**2*x[1] - 4.0*x[4]*x[0]**3*x[3] + 8.0*x[4]*x[0]**5*x[1] +
10.0*x[5]*x[0]**4*x[1] + 4.0*x[5]*x[0]**3*x[3] + 8.0*x[5]*x[0]**5*x[1] + 16.0*x[6]*x[0]**2*x[3] -
2.0*x[4]*x[0]**5*x[3] - 8.0*x[5]*x[0]**4*x[3] - 2.0*x[5]*x[0]**5*x[3] + 8.0*x[0]**3*x[1]*x[3] +
8.0*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0] + 2.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[1] + x[4]*x[6]*x[1] +
2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] - 8.0*x[5]*x[0]*x[1] +
10.0*x[5]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 4.0*x[6]*x[1]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[3] + x[5]*x[6]*x[0]*x[1] + 4.0*x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] +
2.0*x[4]*x[6]*x[1]*x[3] - 4.0*x[5]*x[0]*x[1]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[1] +
2.0*x[4]*x[6]*x[0]**2*x[1] + 2.0*x[5]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[3] +
x[5]*x[6]*x[0]**2*x[3] - 4.0*x[4]*x[0]**3*x[1]*x[3] - 4.0*x[6]*x[0]**2*x[1]*x[3] -
4.0*x[4]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[3])**2 + 24.0*x[5]*(16.0*x[5]*x[0]**2 - 4.0*x[6]*x[3] -
16.0*x[4]*x[0]**3 - 4.0*x[5]*x[3] - 16.0*x[5]*x[0]**4 + 16.0*x[6]*x[0]**3 + 32.0*x[0]**3*x[1] +
64.0*x[0]**3*x[2] - 32.0*x[0]**6*x[3] - 64.0*x[0]**3 + 64.0*x[0]**6 - 16.0*x[4]*x[0]**3*x[1] -

4.0*x[5]*x[0]**2*x[1] + 16.0*x[4]*x[0]**3*x[2] + 4.0*x[5]*x[0]**2*x[2] + 8.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] - 8.0*x[5]*x[0]**4*x[1] + 4.0*x[4]*x[0]**6*x[1] + 16.0*x[5]*x[0]**4*x[2] + 4.0*x[4]*x[0]**6*x[2] + 4.0*x[5]*x[0]**4*x[3] + 4.0*x[5]*x[0]**6*x[1] + 16.0*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[0]**6*x[2] - 32.0*x[0]**3*x[1]*x[2] - 16.0*x[0]**6*x[1]*x[2] - 1.0*x[4]*x[5]*x[3] - 1.0*x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 2.0*x[5]*x[1]*x[3] + 4.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[1]*x[3] + 4.0*x[6]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[1]*x[3] + x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 2.0*x[5]*x[1]*x[2]*x[3] - 2.0*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[5]*x[0]**2*x[2] - 1.0*x[4]*x[5]*x[0]**3*x[1] - 1.0*x[4]*x[5]*x[0]**3*x[2] + x[4]*x[6]*x[0]**3*x[1] + x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[1] + x[5]*x[6]*x[0]**3*x[2] + 16.0*x[4]*x[0]**3*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[1]*x[2] - 2.0*x[5]*x[0]**2*x[1]*x[3] + 4.0*x[5]*x[0]**3*x[1]*x[2] + 2.0*x[5]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**6*x[1]*x[2] + 2.0*x[5]*x[0]**4*x[1]*x[3] - 2.0*x[4]*x[0]**6*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 4.0*x[5]*x[0]**6*x[1]*x[2] - 2.0*x[4]*x[0]**6*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[3] - 2.0*x[5]*x[0]**6*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] + x[4]*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[1]*x[3] - 1.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[2]*x[3] - 2.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 2.0*x[4]*x[0]**6*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[4]*x[0]**4 - 10.0*x[6]*x[3] - 8.0*x[4]*x[0]**2 - 16.0*x[4]*x[0]**3 - 8.0*x[6]*x[0] + 16.0*x[6]*x[0]**2 + 8.0*x[6]*x[0]**3 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] + 64.0*x[0]**3*x[1] - 16.0*x[0]**2*x[3] + 160.0*x[0]**3*x[2] - 160.0*x[0]**4*x[2] + 16.0*x[0]**6*x[1] - 64.0*x[0]**5*x[3] - 32.0*x[0]**6*x[2] - 16.0*x[0]**6*x[3] - 32.0*x[0]**2 - 160.0*x[0]**3 + 160.0*x[0]**5 + 32.0*x[0]**6 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 32.0*x[4]*x[0]**3*x[1] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**3*x[2] - 24.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] + 4.0*x[6]*x[0]**3*x[1] - 2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] + 32.0*x[6]*x[0]**2*x[3] - 8.0*x[6]*x[0]**3*x[2] + 8.0*x[6]*x[0]**3*x[3] - 16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] - 64.0*x[0]**3*x[1]*x[2] + 16.0*x[0]**2*x[2]*x[3] + 64.0*x[0]**4*x[1]*x[2] - 40.0*x[0]**5*x[1]*x[2] + 40.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[1]*x[2] - 8.0*x[0]**6*x[1]*x[3] + 16.0*x[0]**6*x[2]*x[3] - 1.0*x[4]*x[6]*x[3] + 2.0*x[6]*x[0]*x[1] - 2.0*x[6]*x[0]*x[2] + 4.0*x[6]*x[1]*x[3] + 10.0*x[6]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1] - 1.0*x[4]*x[6]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] - 2.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] - 20.0*x[6]*x[0]*x[2]*x[3] - 4.0*x[6]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 32.0*x[4]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 32.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] + 6.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[2]*x[3] - 8.0*x[6]*x[0]**3*x[2]*x[3] - 8.0*x[0]**2*x[1]*x[2]*x[3] - 16.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1]*x[2] - 3.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[2]*x[3] + 8.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +

256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) - (0.00231481481481481481481481481481481*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])*(16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 896.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 1024.0*x[0]**4*x[1] - 64.0*x[0]**4*x[3] - 192.0*x[0]**6*x[1] + 256.0*x[0]**3 + 256.0*x[0]**4 + 64.0*x[0]**3*x[1]**2 - 192.0*x[0]**4*x[1]**2 - 96.0*x[0]**6*x[1]**2 + 16.0*x[4]*x[0]**3*x[1] - 12.0*x[5]*x[0]*x[1]**2 - 160.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] - 32.0*x[5]*x[0]**3*x[1] - 12.0*x[6]*x[0]*x[1]**2 + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 64.0*x[5]*x[0]**4*x[1] - 48.0*x[6]*x[0]**3*x[1] + 16.0*x[5]*x[0]**4*x[2]

+ 4.0*x[5]*x[1]**2*x[3] + 4.0*x[6]*x[1]**2*x[3] + 896.0*x[0]**3*x[1]*x[2] +
128.0*x[0]**4*x[1]*x[2] + 256.0*x[0]**4*x[1]*x[3] + 192.0*x[0]**6*x[1]*x[2] +
96.0*x[0]**6*x[1]*x[3] - 32.0*x[4]*x[0]**3*x[1]**2 - 112.0*x[5]*x[0]**2*x[1]**2 -
16.0*x[4]*x[0]**4*x[1]**2 - 200.0*x[5]*x[0]**3*x[1]**2 - 16.0*x[5]*x[0]**4*x[1]**2 -
24.0*x[6]*x[0]**3*x[1]**2 + 60.0*x[5]*x[0]**5*x[1]**2 - 64.0*x[0]**3*x[1]**2*x[2] +
256.0*x[0]**4*x[1]**2*x[2] + 48.0*x[0]**4*x[1]**2*x[3] + 96.0*x[0]**6*x[1]**2*x[2] +
48.0*x[0]**6*x[1]**2*x[3] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] - 64.0*x[5]*x[0]*x[1] -
64.0*x[6]*x[0]*x[1] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 56.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 56.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] - 5.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] + 8.0*x[5]*x[0]*x[1]*x[2]
- 128.0*x[5]*x[0]*x[1]*x[3] + 8.0*x[6]*x[0]*x[1]*x[2] - 128.0*x[6]*x[0]*x[1]*x[3] +
56.0*x[5]*x[1]*x[2]*x[3] + 56.0*x[6]*x[1]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]*x[1]**2 -
1.0*x[4]*x[6]*x[0]*x[1]**2 - 7.0*x[5]*x[6]*x[0]*x[1]**2 - 2.0*x[4]*x[5]*x[1]**2*x[3] -
2.0*x[4]*x[6]*x[1]**2*x[3] - 14.0*x[5]*x[6]*x[1]**2*x[3] - 16.0*x[4]*x[0]**3*x[1]*x[2] +
16.0*x[5]*x[0]*x[1]**2*x[2] + 160.0*x[5]*x[0]**2*x[1]*x[2] - 24.0*x[5]*x[0]*x[1]**2*x[3] -
80.0*x[5]*x[0]**2*x[1]*x[3] + 32.0*x[5]*x[0]**3*x[1]*x[2] + 16.0*x[6]*x[0]*x[1]**2*x[2] -
4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] - 144.0*x[5]*x[0]**4*x[1]*x[2] -
24.0*x[6]*x[0]*x[1]**2*x[3] + 48.0*x[6]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
16.0*x[5]*x[0]**4*x[1]*x[3] + 60.0*x[5]*x[0]**5*x[1]*x[2] - 48.0*x[6]*x[0]**3*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[1]**2*x[2]*x[3] - 4.0*x[6]*x[1]**2*x[2]*x[3] -
32.0*x[0]**4*x[1]*x[2]*x[3] - 96.0*x[0]**6*x[1]*x[2]*x[3] + 6.0*x[5]*x[6]*x[0]**2*x[1]**2 +
32.0*x[4]*x[0]**3*x[1]**2*x[2] + 112.0*x[5]*x[0]**2*x[1]**2*x[2] -16.0*x[4]*x[0]**4*x[1]**2*x[2]
- 56.0*x[5]*x[0]**2*x[1]**2*x[3]+200.0*x[5]*x[0]**3*x[1]**2*x[2]+ 4.0*x[4]*x[0]**4*x[1]**2*x[3]
- 208.0*x[5]*x[0]**4*x[1]**2*x[2] + 24.0*x[6]*x[0]**3*x[1]**2*x[2]+4.0*x[5]*x[0]**4*x[1]**2*x[3]
+ 60.0*x[5]*x[0]**5*x[1]**2*x[2] - 24.0*x[6]*x[0]**3*x[1]**2*x[3]-24.0*x[5]*x[0]**5*x[1]**2*x[3]
-64.0*x[0]**4*x[1]**2*x[2]*x[3]-48.0*x[0]**6*x[1]**2*x[2]*x[3]+
6.0*x[5]*x[6]*x[0]**2*x[1]**2*x[2]+ 12.0*x[5]*x[6]*x[0]**2*x[1]**2*x[3] +
56.0*x[5]*x[0]**2*x[1]**2*x[2]*x[3] + 4.0*x[4]*x[0]**4*x[1]**2*x[2]*x[3] +
52.0*x[5]*x[0]**4*x[1]**2*x[2]*x[3] + 24.0*x[6]*x[0]**3*x[1]**2*x[2]*x[3] -
24.0*x[5]*x[0]**5*x[1]**2*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3]
+ 2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 6.0*x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
8.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[1]*x[2]*x[3] -
1.0*x[4]*x[6]*x[1]*x[2]*x[3] + 5.0*x[5]*x[6]*x[1]*x[2]*x[3] + 16.0*x[5]*x[0]*x[1]*x[2]*x[3] +
16.0*x[6]*x[0]*x[1]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]*x[1]**2*x[2] - 2.0*x[4]*x[5]*x[0]*x[1]**2*x[3] -
1.0*x[4]*x[6]*x[0]*x[1]**2*x[2] - 2.0*x[4]*x[6]*x[0]*x[1]**2*x[3] - 7.0*x[5]*x[6]*x[0]*x[1]**2*x[2]
+ 6.0*x[5]*x[6]*x[0]**2*x[1]*x[2]-2.0*x[5]*x[6]*x[0]*x[1]**2*x[3]+2.0*x[4]*x[5]*x[1]**2*x[2]*x[3]
+2.0*x[4]*x[6]*x[1]**2*x[2]*x[3]+14.0*x[5]*x[6]*x[1]**2*x[2]*x[3]
+32.0*x[5]*x[0]*x[1]**2*x[2]*x[3]+ 80.0*x[5]*x[0]**2*x[1]*x[2]*x[3]
+32.0*x[6]*x[0]*x[1]**2*x[2]*x[3]+ 36.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
48.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 24.0*x[5]*x[0]**5*x[1]*x[2]*x[3] -
2.0*x[4]*x[5]*x[0]*x[1]**2*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]**2*x[2]*x[3] -
26.0*x[5]*x[6]*x[0]*x[1]**2*x[2]*x[3] + 12.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +
12.0*x[5]*x[6]*x[0]**2*x[1]**2*x[2]*x[3] - 18.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]))/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +

```
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) - (0.000108506944444444444444444444444*x[0]**4*
(16.0*x[5]*(8.0*x[4]*x[0]**4 - 10.0*x[6]*x[3] - 8.0*x[4]*x[0]**2 - 16.0*x[4]*x[0]**3 - 8.0*x[6]*x[0]
+ 16.0*x[6]*x[0]**2 + 8.0*x[6]*x[0]**3 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] + 64.0*x[0]**3*x[1]
- 16.0*x[0]**2*x[3] + 160.0*x[0]**3*x[2] - 160.0*x[0]**4*x[2] + 16.0*x[0]**6*x[1] -
64.0*x[0]**5*x[3] - 32.0*x[0]**6*x[2] - 16.0*x[0]**6*x[3] - 32.0*x[0]**2 - 160.0*x[0]**3 +
160.0*x[0]**5 + 32.0*x[0]**6 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] -
32.0*x[4]*x[0]**3*x[1] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**3*x[2] - 24.0*x[4]*x[0]**4*x[2] +
10.0*x[4]*x[0]**5*x[1] + 4.0*x[6]*x[0]**3*x[1] - 2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] +
32.0*x[6]*x[0]**2*x[3] - 8.0*x[6]*x[0]**3*x[2] + 8.0*x[6]*x[0]**3*x[3] - 16.0*x[0]**2*x[1]*x[2] +
8.0*x[0]**2*x[1]*x[3] - 64.0*x[0]**3*x[1]*x[2] + 16.0*x[0]**2*x[2]*x[3] + 64.0*x[0]**4*x[1]*x[2] -
40.0*x[0]**5*x[1]*x[2] + 40.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[1]*x[2] - 8.0*x[0]**6*x[1]*x[3] +
16.0*x[0]**6*x[2]*x[3] - 1.0*x[4]*x[6]*x[3] + 2.0*x[6]*x[0]*x[1] - 2.0*x[6]*x[0]*x[2] +
4.0*x[6]*x[1]*x[3] + 10.0*x[6]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1] - 1.0*x[4]*x[6]*x[0]*x[2] +
x[4]*x[6]*x[0]*x[3] - 2.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] -
20.0*x[6]*x[0]*x[2]*x[3] - 4.0*x[6]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] +
x[4]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] +
32.0*x[4]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 32.0*x[4]*x[0]**4*x[1]*x[2] -
4.0*x[6]*x[0]**2*x[1]*x[2] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] +
6.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[3] -
4.0*x[4]*x[0]**5*x[2]*x[3] - 8.0*x[6]*x[0]**3*x[2]*x[3] - 8.0*x[0]**2*x[1]*x[2]*x[3] -
16.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] -
1.0*x[4]*x[6]*x[0]*x[1]*x[2] - 3.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[2]*x[3] +
8.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] +
8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3]
- 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[5]*x[0] + 10.0*x[5]*x[3] - 8.0*x[4]*x[0]**2 +
8.0*x[5]*x[0]**2 + 8.0*x[4]*x[0]**4 + 8.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 + 8.0*x[5]*x[0]**4 +
16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] - 16.0*x[0]**2*x[3] - 32.0*x[0]**4*x[1] - 128.0*x[0]**4*x[2]
- 32.0*x[0]**4*x[3] - 64.0*x[0]**5*x[3] - 32.0*x[0]**2 + 128.0*x[0]**4 + 160.0*x[0]**5 -
8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**4*x[1] -
8.0*x[5]*x[0]**2*x[2] + 4.0*x[5]*x[0]**3*x[1] - 8.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] +
4.0*x[5]*x[0]**2*x[3] - 8.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] - 2.0*x[4]*x[0]**4*x[3] +
10.0*x[4]*x[0]**5*x[2] - 8.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 32.0*x[6]*x[0]**2*x[3] -
```

2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] + 16.0*x[0]**2*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 40.0*x[0]**5*x[1]*x[2] + 32.0*x[0]**4*x[2]*x[3] + x[4]*x[5]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] + 16.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 10.0*x[5]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] + x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] - 16.0*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[2] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 16.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[2]*x[3] - 16.0*x[5]*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[4]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] - 8.0*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 1.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] + 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]) + 192.0*x[0]*(x[2] - 1.0)*(4.0*x[5] + 4.0*x[6] + x[4]*x[5] + x[4]*x[6] - 1.0*x[5]*x[6] + 20.0*x[5]*x[0] - 2.0*x[5]*x[1] - 2.0*x[6]*x[1] + 8.0*x[5]*x[3] + 8.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 - 16.0*x[5]*x[0]**2 - 16.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 + 4.0*x[4]*x[0]**5 + 20.0*x[5]*x[0]**4 + 4.0*x[5]*x[0]**5 - 32.0*x[0]**3*x[1] - 16.0*x[0]**3*x[3] - 16.0*x[0]**5*x[1] - 32.0*x[0]**5*x[3] + 64.0*x[0]**3 + 64.0*x[0]**5 - 1.0*x[4]*x[5]*x[0]**2 + x[4]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**2 + 16.0*x[4]*x[0]**3*x[1] + 4.0*x[5]*x[0]**2*x[1] - 4.0*x[6]*x[0]**2*x[1] - 4.0*x[4]*x[0]**3*x[3] + 8.0*x[4]*x[0]**5*x[1] + 10.0*x[5]*x[0]**4*x[1] + 4.0*x[5]*x[0]**3*x[3] + 8.0*x[5]*x[0]**5*x[1] + 16.0*x[6]*x[0]**2*x[3] - 2.0*x[4]*x[0]**5*x[3] - 8.0*x[5]*x[0]**4*x[3] - 2.0*x[5]*x[0]**5*x[3] + 8.0*x[0]**3*x[1]*x[3] + 8.0*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0] + 2.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[1] + x[4]*x[6]*x[1] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] - 8.0*x[5]*x[0]*x[1] + 10.0*x[5]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 4.0*x[6]*x[1]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[3] + x[5]*x[6]*x[0]*x[1] + 4.0*x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 4.0*x[5]*x[0]*x[1]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[6]*x[0]**2*x[1] + 2.0*x[5]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[3] + x[5]*x[6]*x[0]**2*x[3] - 4.0*x[4]*x[0]**3*x[1]*x[3] - 4.0*x[6]*x[0]**2*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3])*(16.0*x[5]*x[0]**2 - 4.0*x[6]*x[3] - 16.0*x[4]*x[0]**3 - 4.0*x[5]*x[3] - 16.0*x[5]*x[0]**4 + 16.0*x[6]*x[0]**3 + 32.0*x[0]**3*x[1] + 64.0*x[0]**3*x[2] - 32.0*x[0]**6*x[3] - 64.0*x[0]**3 + 64.0*x[0]**6 - 16.0*x[4]*x[0]**3*x[1] - 4.0*x[5]*x[0]**2*x[1] + 16.0*x[4]*x[0]**3*x[2] + 4.0*x[5]*x[0]**2*x[2] + 8.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] - 8.0*x[5]*x[0]**4*x[1] + 4.0*x[4]*x[0]**6*x[1] + 16.0*x[5]*x[0]**4*x[2] + 4.0*x[4]*x[0]**6*x[2] + 4.0*x[5]*x[0]**4*x[3] + 4.0*x[5]*x[0]**6*x[1] + 16.0*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[0]**6*x[2] - 32.0*x[0]**3*x[1]*x[2] - 16.0*x[0]**6*x[1]*x[2] - 1.0*x[4]*x[5]*x[3] - 1.0*x[4]*x[6]*x[3] +

x[5]*x[6]*x[3] + 2.0*x[5]*x[1]*x[3] + 4.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[1]*x[3] + 4.0*x[6]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[1]*x[3] + x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 2.0*x[5]*x[1]*x[2]*x[3] - 2.0*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[5]*x[0]**2*x[2] - 1.0*x[4]*x[5]*x[0]**3*x[1] - 1.0*x[4]*x[5]*x[0]**3*x[2] + x[4]*x[6]*x[0]**3*x[1] + x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[1] + x[5]*x[6]*x[0]**3*x[2] + 16.0*x[4]*x[0]**3*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[1]*x[2] - 2.0*x[5]*x[0]**2*x[1]*x[3] + 4.0*x[5]*x[0]**3*x[1]*x[2] + 2.0*x[5]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**6*x[1]*x[2] + 2.0*x[5]*x[0]**4*x[1]*x[3] - 2.0*x[4]*x[0]**6*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 4.0*x[5]*x[0]**6*x[1]*x[2] - 2.0*x[4]*x[0]**6*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[3] - 2.0*x[5]*x[0]**6*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] + x[4]*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[1]*x[3] - 1.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[2]*x[3] - 2.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 2.0*x[4]*x[0]**6*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) - (0.0000723379629629629629629629629629629630*x[0]**6*(192.0*(16.0*x[5]*x[0]**2 - 4.0*x[6]*x[3] - 16.0*x[4]*x[0]**3 - 4.0*x[5]*x[3] - 16.0*x[5]*x[0]**4 + 16.0*x[6]*x[0]**3 + 32.0*x[0]**3*x[1] + 64.0*x[0]**3*x[2] - 32.0*x[0]**6*x[3] - 64.0*x[0]**3 + 64.0*x[0]**6 - 16.0*x[4]*x[0]**3*x[1] - 4.0*x[5]*x[0]**2*x[1] + 16.0*x[4]*x[0]**3*x[2] + 4.0*x[5]*x[0]**2*x[2] + 8.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] - 8.0*x[5]*x[0]**4*x[1] + 4.0*x[4]*x[0]**6*x[1] + 16.0*x[5]*x[0]**4*x[2] + 4.0*x[4]*x[0]**6*x[2] + 4.0*x[5]*x[0]**4*x[3] + 4.0*x[5]*x[0]**6*x[1] + 16.0*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[0]**6*x[2] - 32.0*x[0]**3*x[1]*x[2] - 16.0*x[0]**6*x[1]*x[2] - 1.0*x[4]*x[5]*x[3] - 1.0*x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 2.0*x[5]*x[1]*x[3] + 4.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[1]*x[3] +

4.0*x[6]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[1]*x[3] + x[4]*x[5]*x[2]*x[3] -
1.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 2.0*x[5]*x[1]*x[2]*x[3] -
2.0*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[5]*x[0]**2*x[2] -
1.0*x[4]*x[5]*x[0]**3*x[1] - 1.0*x[4]*x[5]*x[0]**3*x[2] + x[4]*x[6]*x[0]**3*x[1] +
x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[1] + x[5]*x[6]*x[0]**3*x[2] +
16.0*x[4]*x[0]**3*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[1]*x[2] - 2.0*x[5]*x[0]**2*x[1]*x[3] +
4.0*x[5]*x[0]**3*x[1]*x[2] + 2.0*x[5]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2] -
4.0*x[6]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**6*x[1]*x[2] + 2.0*x[5]*x[0]**4*x[1]*x[3] -
2.0*x[4]*x[0]**6*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 4.0*x[5]*x[0]**6*x[1]*x[2] -
2.0*x[4]*x[0]**6*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[3] - 2.0*x[5]*x[0]**6*x[2]*x[3] +
8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] +
x[4]*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1]*x[2] +
x[4]*x[5]*x[0]**2*x[1]*x[3] - 1.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[2]*x[3] +
x[4]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2] +
x[4]*x[6]*x[0]**3*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[2]*x[3] -
2.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3]
- 2.0*x[4]*x[0]**6*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] +
x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[5]*x[0] + 10.0*x[5]*x[3] -
8.0*x[4]*x[0]**2 + 8.0*x[5]*x[0]**2 + 8.0*x[4]*x[0]**4 + 8.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 +
8.0*x[5]*x[0]**4 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] - 16.0*x[0]**2*x[3] - 32.0*x[0]**4*x[1] -
128.0*x[0]**4*x[2] - 32.0*x[0]**4*x[3] - 64.0*x[0]**5*x[3] - 32.0*x[0]**2 + 128.0*x[0]**4 +
160.0*x[0]**5 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 4.0*x[4]*x[0]**2*x[3] +
16.0*x[4]*x[0]**4*x[1] - 8.0*x[5]*x[0]**2*x[2] + 4.0*x[5]*x[0]**3*x[1] - 8.0*x[4]*x[0]**4*x[2] +
10.0*x[4]*x[0]**5*x[1] + 4.0*x[5]*x[0]**2*x[3] - 8.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] -
2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] - 8.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] +
32.0*x[6]*x[0]**2*x[3] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 16.0*x[0]**2*x[1]*x[2] +
8.0*x[0]**2*x[1]*x[3] + 16.0*x[0]**2*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] -
40.0*x[0]**5*x[1]*x[2] + 32.0*x[0]**4*x[2]*x[3] + x[4]*x[5]*x[3] - 2.0*x[5]*x[0]*x[1] +
2.0*x[5]*x[0]*x[2] + 16.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
10.0*x[5]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] +
x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 2.0*x[5]*x[0]*x[1]*x[2] -
4.0*x[5]*x[0]*x[1]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] -
16.0*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] +
x[4]*x[6]*x[0]**2*x[2] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] +
8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 4.0*x[4]*x[0]**2*x[2]*x[3] -
16.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] -
4.0*x[4]*x[0]**4*x[1]*x[3] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[2]*x[3] -
16.0*x[5]*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[4]*x[0]**5*x[2]*x[3] +
2.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 1.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] -
1.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] +
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3]
+ 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +

2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]) + 16.0*x[0]*(x[2] - 1.0)*(4.0*x[5] + 4.0*x[6] + x[4]*x[5] +
x[4]*x[6] - 1.0*x[5]*x[6] + 20.0*x[5]*x[0] - 2.0*x[5]*x[1] - 2.0*x[6]*x[1] + 8.0*x[5]*x[3] +
8.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 - 16.0*x[5]*x[0]**2 - 16.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 +
4.0*x[4]*x[0]**5 + 20.0*x[5]*x[0]**4 + 4.0*x[5]*x[0]**5 - 32.0*x[0]**3*x[1] - 16.0*x[0]**3*x[3] -
16.0*x[0]**5*x[1] - 32.0*x[0]**5*x[3] + 64.0*x[0]**3 + 64.0*x[0]**5 - 1.0*x[4]*x[5]*x[0]**2 +
x[4]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**2 + 16.0*x[4]*x[0]**3*x[1] + 4.0*x[5]*x[0]**2*x[1] -
4.0*x[6]*x[0]**2*x[1] - 4.0*x[4]*x[0]**3*x[3] + 8.0*x[4]*x[0]**5*x[1] + 10.0*x[5]*x[0]**4*x[1] +
4.0*x[5]*x[0]**3*x[3] + 8.0*x[5]*x[0]**5*x[1] + 16.0*x[6]*x[0]**2*x[3] - 2.0*x[4]*x[0]**5*x[3] -
8.0*x[5]*x[0]**4*x[3] - 2.0*x[5]*x[0]**5*x[3] + 8.0*x[0]**3*x[1]*x[3] + 8.0*x[0]**5*x[1]*x[3] +
2.0*x[4]*x[5]*x[0] + 2.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[1] + x[4]*x[6]*x[1] + 2.0*x[4]*x[5]*x[3] +
2.0*x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] - 8.0*x[5]*x[0]*x[1] + 10.0*x[5]*x[0]*x[3] -
4.0*x[5]*x[1]*x[3] - 4.0*x[6]*x[1]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[3] +
x[5]*x[6]*x[0]*x[1] + 4.0*x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
4.0*x[5]*x[0]*x[1]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[6]*x[0]**2*x[1] +
2.0*x[5]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[3] + x[5]*x[6]*x[0]**2*x[3] -
4.0*x[4]*x[0]**3*x[1]*x[3] - 4.0*x[6]*x[0]**2*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3])
*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 +
64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] - 512.0*x[0]**3*x[2] -
128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1] +
32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2] +
64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] - 32.0*x[5]*x[0]**3*x[2]
+ 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] + 128.0*x[0]**3*x[1]*x[2] -
128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] + 2.0*x[5]*x[6]*x[3] +
64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] - 32.0*x[5]*x[2]*x[3] -
8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] + 2.0*x[4]*x[5]*x[0]*x[2] +
2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] + 2.0*x[5]*x[6]*x[0]*x[1] +
2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] - 2.0*x[4]*x[5]*x[2]*x[3] +
4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] + 4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3]
- 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2] + 8.0*x[5]*x[1]*x[2]*x[3] +
8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] - 32.0*x[5]*x[0]**2*x[1]*x[2] +
32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] - 64.0*x[5]*x[0]**3*x[1]*x[2] -
8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] + 32.0*x[5]*x[0]**4*x[1]*x[2] -
8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] - 8.0*x[5]*x[0]**4*x[2]*x[3] +
32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[1]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[3] +
4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] - 16.0*x[6]*x[0]*x[1]*x[2]*x[3] -
16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -

16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) + (0.000868055555555555555555555555556*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 + 64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] - 512.0*x[0]**3*x[2] - 128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1] + 32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2] + 64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] - 32.0*x[5]*x[0]**3*x[2] + 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] + 128.0*x[0]**3*x[1]*x[2] - 128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] + 2.0*x[5]*x[6]*x[3] + 64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] - 32.0*x[5]*x[2]*x[3] - 8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] + 2.0*x[4]*x[5]*x[0]*x[2] + 2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] + 2.0*x[5]*x[6]*x[0]*x[1] + 2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] - 2.0*x[4]*x[5]*x[2]*x[3] + 4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] + 4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2] + 8.0*x[5]*x[1]*x[2]*x[3] + 8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] - 32.0*x[5]*x[0]**2*x[1]*x[2] + 32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] - 64.0*x[5]*x[0]**3*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] + 32.0*x[5]*x[0]**4*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] - 8.0*x[5]*x[0]**4*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[3] + 4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] - 16.0*x[6]*x[0]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3]-8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] +

6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] +
x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] -
4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[0]*x[2]*x[3] -
4.0*x[6]*x[0]*x[1]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2]
+ 16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] + 28.0*x[5]*x[0]**3*x[1]*x[2] +
12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] + 4.0*x[6]*x[0]**3*x[1]*x[2] +
10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] + 6.0*x[5]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[2] -
3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] +
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +
16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -
16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] +
x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] -
1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2]
+ 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] -
16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] -
32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] +
16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] -
2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3]+2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3]
+2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) +
(0.0000868055555555555555555555555555556*x[0]**5*(144.0*(16.0*x[5]*x[0]**2 - 4.0*x[6]*x[3] -
16.0*x[4]*x[0]**3 - 4.0*x[5]*x[3] - 16.0*x[5]*x[0]**4 + 16.0*x[6]*x[0]**3 + 32.0*x[0]**3*x[1] +
64.0*x[0]**3*x[2] - 32.0*x[0]**6*x[3] - 64.0*x[0]**3 + 64.0*x[0]**6 - 16.0*x[4]*x[0]**3*x[1] -
4.0*x[5]*x[0]**2*x[1] + 16.0*x[4]*x[0]**3*x[2] + 4.0*x[5]*x[0]**2*x[2] + 8.0*x[5]*x[0]**2*x[3] -
16.0*x[5]*x[0]**3*x[2] - 8.0*x[5]*x[0]**4*x[1] + 4.0*x[4]*x[0]**6*x[1] + 16.0*x[5]*x[0]**4*x[2] +
4.0*x[4]*x[0]**6*x[2] + 4.0*x[5]*x[0]**4*x[3] + 4.0*x[5]*x[0]**6*x[1] + 16.0*x[6]*x[0]**3*x[3] +
4.0*x[5]*x[0]**6*x[2] - 32.0*x[0]**3*x[1]*x[2] - 16.0*x[0]**6*x[1]*x[2] - 1.0*x[4]*x[5]*x[3] -
1.0*x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 2.0*x[5]*x[1]*x[3] + 4.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[1]*x[3] +
4.0*x[6]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[1]*x[3] + x[4]*x[5]*x[2]*x[3] -
1.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 2.0*x[5]*x[1]*x[2]*x[3] -
2.0*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[5]*x[0]**2*x[2] -
1.0*x[4]*x[5]*x[0]**3*x[1] - 1.0*x[4]*x[5]*x[0]**3*x[2] + x[4]*x[6]*x[0]**3*x[1] +
x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[1] + x[5]*x[6]*x[0]**3*x[2] +

16.0*x[4]*x[0]**3*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[1]*x[2] - 2.0*x[5]*x[0]**2*x[1]*x[3] +
4.0*x[5]*x[0]**3*x[1]*x[2] + 2.0*x[5]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2] -
4.0*x[6]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**6*x[1]*x[2] + 2.0*x[5]*x[0]**4*x[1]*x[3] -
2.0*x[4]*x[0]**6*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 4.0*x[5]*x[0]**6*x[1]*x[2] -
2.0*x[4]*x[0]**6*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[3] - 2.0*x[5]*x[0]**6*x[2]*x[3] +
8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] +
x[4]*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1]*x[2] +
x[4]*x[5]*x[0]**2*x[1]*x[3] - 1.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[2]*x[3] +
x[4]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2] +
x[4]*x[6]*x[0]**3*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[2]*x[3] -
2.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3]
-2.0*x[4]*x[0]**6*x[1]*x[2]*x[3]-2.0*x[5]*x[0]**6*x[1]*x[2]*x[3]+x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3]
+x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3]+x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3]+x[5]*x[6]*x[0]*x[1]*x[2]*x[3])
**2 + 2.0*x[5]*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] +
32.0*x[4]*x[0]**3 + 64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] -
512.0*x[0]**3*x[2] - 128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1]
+ 32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2]
+ 64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] -
32.0*x[5]*x[0]**3*x[2] + 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] +
128.0*x[0]**3*x[1]*x[2] - 128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] +
2.0*x[5]*x[6]*x[3] + 64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] -
32.0*x[5]*x[2]*x[3] - 8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] +
2.0*x[4]*x[5]*x[0]*x[2] + 2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] +
2.0*x[5]*x[6]*x[0]*x[1] + 2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] -
2.0*x[4]*x[5]*x[2]*x[3] + 4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] +
4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2]
+ 8.0*x[5]*x[1]*x[2]*x[3] + 8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] -
32.0*x[5]*x[0]**2*x[1]*x[2] + 32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] -
64.0*x[5]*x[0]**3*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] +
32.0*x[5]*x[0]**4*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] -
8.0*x[5]*x[0]**4*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] +
4.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] +
4.0*x[5]*x[6]*x[0]*x[1]*x[3] + 4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] -
4.0*x[4]*x[6]*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] -
16.0*x[6]*x[0]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3]+
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3]+ 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[4]*x[0]**4 -
10.0*x[6]*x[3] - 8.0*x[4]*x[0]**2 - 16.0*x[4]*x[0]**3 - 8.0*x[6]*x[0] + 16.0*x[6]*x[0]**2 +
8.0*x[6]*x[0]**3 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] + 64.0*x[0]**3*x[1] - 16.0*x[0]**2*x[3] +
160.0*x[0]**3*x[2] - 160.0*x[0]**4*x[2] + 16.0*x[0]**6*x[1] - 64.0*x[0]**5*x[3] - 32.0*x[0]**6*x[2]
- 16.0*x[0]**6*x[3] - 32.0*x[0]**2 - 160.0*x[0]**3 + 160.0*x[0]**5 + 32.0*x[0]**6 -
8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 32.0*x[4]*x[0]**3*x[1] - 4.0*x[4]*x[0]**2*x[3] +
16.0*x[4]*x[0]**3*x[2] - 24.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] + 4.0*x[6]*x[0]**3*x[1] -
2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] + 32.0*x[6]*x[0]**2*x[3] - 8.0*x[6]*x[0]**3*x[2] +
8.0*x[6]*x[0]**3*x[3] - 16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] - 64.0*x[0]**3*x[1]*x[2] +
16.0*x[0]**2*x[2]*x[3] + 64.0*x[0]**4*x[1]*x[2] - 40.0*x[0]**5*x[1]*x[2] + 40.0*x[0]**4*x[2]*x[3] -
16.0*x[0]**6*x[1]*x[2] - 8.0*x[0]**6*x[1]*x[3] + 16.0*x[0]**6*x[2]*x[3] - 1.0*x[4]*x[6]*x[3] +
2.0*x[6]*x[0]*x[1] - 2.0*x[6]*x[0]*x[2] + 4.0*x[6]*x[1]*x[3] + 10.0*x[6]*x[2]*x[3] -
1.0*x[4]*x[6]*x[0]*x[1] - 1.0*x[4]*x[6]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] - 2.0*x[4]*x[6]*x[1]*x[3] +
x[4]*x[6]*x[2]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] - 20.0*x[6]*x[0]*x[2]*x[3] - 4.0*x[6]*x[1]*x[2]*x[3] +

x[4]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 32.0*x[4]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 32.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] + 6.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[2]*x[3] - 8.0*x[6]*x[0]**3*x[2]*x[3] - 8.0*x[0]**2*x[1]*x[2]*x[3] - 16.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1]*x[2] - 3.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[2]*x[3] + 8.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3]+2.0*x[4]*x[6]*x[0]**2*x[2]*x[3]+4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3]) + 128.0*x[0]*(x[2] - 1.0)*(4.0*x[5] + 4.0*x[6] + x[4]*x[5] + x[4]*x[6] - 1.0*x[5]*x[6] + 20.0*x[5]*x[0] - 2.0*x[5]*x[1] - 2.0*x[6]*x[1] + 8.0*x[5]*x[3] + 8.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 - 16.0*x[5]*x[0]**2 - 16.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 + 4.0*x[4]*x[0]**5 + 20.0*x[5]*x[0]**4 + 4.0*x[5]*x[0]**5 - 32.0*x[0]**3*x[1] - 16.0*x[0]**3*x[3] - 16.0*x[0]**5*x[1] - 32.0*x[0]**5*x[3] + 64.0*x[0]**3 + 64.0*x[0]**5 - 1.0*x[4]*x[5]*x[0]**2 + x[4]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**2 + 16.0*x[4]*x[0]**3*x[1] + 4.0*x[5]*x[0]**2*x[1] - 4.0*x[6]*x[0]**2*x[1] - 4.0*x[4]*x[0]**3*x[3] + 8.0*x[4]*x[0]**5*x[1] + 10.0*x[5]*x[0]**4*x[1] + 4.0*x[5]*x[0]**3*x[3] + 8.0*x[5]*x[0]**5*x[1] + 16.0*x[6]*x[0]**2*x[3] - 2.0*x[4]*x[0]**5*x[3] - 8.0*x[5]*x[0]**4*x[3] - 2.0*x[5]*x[0]**5*x[3] + 8.0*x[0]**3*x[1]*x[3] + 8.0*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0] + 2.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[1] + x[4]*x[6]*x[1] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] - 8.0*x[5]*x[0]*x[1] + 10.0*x[5]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 4.0*x[6]*x[1]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[3] + x[5]*x[6]*x[0]*x[1] + 4.0*x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 4.0*x[5]*x[0]*x[1]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[6]*x[0]**2*x[1] + 2.0*x[5]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[3] + x[5]*x[6]*x[0]**2*x[3] - 4.0*x[4]*x[0]**3*x[1]*x[3] - 4.0*x[6]*x[0]**2*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3])*(8.0*x[5]*x[0] + 10.0*x[5]*x[3] - 8.0*x[4]*x[0]**2 + 8.0*x[5]*x[0]**2 + 8.0*x[4]*x[0]**4 + 8.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 + 8.0*x[5]*x[0]**4 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] - 16.0*x[0]**2*x[3] - 32.0*x[0]**4*x[1] - 128.0*x[0]**4*x[2] - 32.0*x[0]**4*x[3] - 64.0*x[0]**5*x[3] - 32.0*x[0]**2 + 128.0*x[0]**4 + 160.0*x[0]**5 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**4*x[1] - 8.0*x[5]*x[0]**2*x[2] + 4.0*x[5]*x[0]**3*x[1] - 8.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] + 4.0*x[5]*x[0]**2*x[3] - 8.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] - 2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] - 8.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 32.0*x[6]*x[0]**2*x[3] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] + 16.0*x[0]**2*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 40.0*x[0]**5*x[1]*x[2] + 32.0*x[0]**4*x[2]*x[3] + x[4]*x[5]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] + 16.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 10.0*x[5]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] + x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] - 16.0*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[2] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 16.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[2]*x[3] - 16.0*x[5]*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[4]*x[0]**5*x[2]*x[3] +

2.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 1.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] -
1.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3]+4.0*x[4]*x[0]**4*x[1]*x[2]*x[3]
- 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3] +
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +
16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -
16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] +
x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] -
1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2]
+ 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] -
16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] -
32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] +
16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] -
2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) +
(0.0000620039682539682539682539682539683968*x[0]**7*(64.0*(8.0*x[5]*x[0] + 10.0*x[5]*x[3] -
8.0*x[4]*x[0]**2 + 8.0*x[5]*x[0]**2 + 8.0*x[4]*x[0]**4 + 8.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 +
8.0*x[5]*x[0]**4 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] - 16.0*x[0]**2*x[3] - 32.0*x[0]**4*x[1] -
128.0*x[0]**4*x[2] - 32.0*x[0]**4*x[3] - 64.0*x[0]**5*x[3] - 32.0*x[0]**2 + 128.0*x[0]**4 +
160.0*x[0]**5 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] - 4.0*x[4]*x[0]**2*x[3] +
16.0*x[4]*x[0]**4*x[1] - 8.0*x[5]*x[0]**2*x[2] + 4.0*x[5]*x[0]**3*x[1] - 8.0*x[4]*x[0]**4*x[2] +
10.0*x[4]*x[0]**5*x[1] + 4.0*x[5]*x[0]**2*x[3] - 8.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] -
2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] - 8.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] +
32.0*x[6]*x[0]**2*x[3] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 16.0*x[0]**2*x[1]*x[2] +
8.0*x[0]**2*x[1]*x[3] + 16.0*x[0]**2*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] -
40.0*x[0]**5*x[1]*x[2] + 32.0*x[0]**4*x[2]*x[3] + x[4]*x[5]*x[3] - 2.0*x[5]*x[0]*x[1] +
2.0*x[5]*x[0]*x[2] + 16.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
10.0*x[5]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] +
x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 2.0*x[5]*x[0]*x[1]*x[2] -
4.0*x[5]*x[0]*x[1]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] -

16.0*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] +
x[4]*x[6]*x[0]**2*x[2] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] +
8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 4.0*x[4]*x[0]**2*x[2]*x[3] -
16.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] -
4.0*x[4]*x[0]**4*x[1]*x[3] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[2]*x[3] -
16.0*x[5]*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[4]*x[0]**5*x[2]*x[3] +
2.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 1.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] -
1.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] +
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3]
+ 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2 + 24.0*(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] +
32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 + 64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] -
512.0*x[0]**3*x[2] - 128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1]
+ 32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2]
+ 64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] -
32.0*x[5]*x[0]**3*x[2] + 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] +
128.0*x[0]**3*x[1]*x[2] - 128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] +
2.0*x[5]*x[6]*x[3] + 64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] -
32.0*x[5]*x[2]*x[3] - 8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] +
2.0*x[4]*x[5]*x[0]*x[2] + 2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] +
2.0*x[5]*x[6]*x[0]*x[1] + 2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] -
2.0*x[4]*x[5]*x[2]*x[3] + 4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] +
4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2]
+ 8.0*x[5]*x[1]*x[2]*x[3] + 8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] -
32.0*x[5]*x[0]**2*x[1]*x[2] + 32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] -
64.0*x[5]*x[0]**3*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] +
32.0*x[5]*x[0]**4*x[1]*x[2] - 8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] -
8.0*x[5]*x[0]**4*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] +
4.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] +
4.0*x[5]*x[6]*x[0]*x[1]*x[3] + 4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] -
4.0*x[4]*x[6]*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] -
16.0*x[6]*x[0]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(16.0*x[5]*x[0]**2 -
4.0*x[6]*x[3] - 16.0*x[4]*x[0]**3 - 4.0*x[5]*x[3] - 16.0*x[5]*x[0]**4 + 16.0*x[6]*x[0]**3 +
32.0*x[0]**3*x[1] + 64.0*x[0]**3*x[2] - 32.0*x[0]**6*x[3] - 64.0*x[0]**3 + 64.0*x[0]**6 -
16.0*x[4]*x[0]**3*x[1] - 4.0*x[5]*x[0]**2*x[1] + 16.0*x[4]*x[0]**3*x[2] + 4.0*x[5]*x[0]**2*x[2] +
8.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] - 8.0*x[5]*x[0]**4*x[1] + 4.0*x[4]*x[0]**6*x[1] +
16.0*x[5]*x[0]**4*x[2] + 4.0*x[4]*x[0]**6*x[2] + 4.0*x[5]*x[0]**4*x[3] + 4.0*x[5]*x[0]**6*x[1] +
16.0*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[0]**6*x[2] - 32.0*x[0]**3*x[1]*x[2] - 16.0*x[0]**6*x[1]*x[2] -
1.0*x[4]*x[5]*x[3] - 1.0*x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 2.0*x[5]*x[1]*x[3] + 4.0*x[5]*x[2]*x[3] +

2.0*x[6]*x[1]*x[3] + 4.0*x[6]*x[2]*x[3] - 2.0*x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[1]*x[3] +
x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] -
2.0*x[5]*x[1]*x[2]*x[3] - 2.0*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1] +
2.0*x[4]*x[5]*x[0]**2*x[2] - 1.0*x[4]*x[5]*x[0]**3*x[1] - 1.0*x[4]*x[5]*x[0]**3*x[2] +
x[4]*x[6]*x[0]**3*x[1] + x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[1] + x[5]*x[6]*x[0]**3*x[2]
+ 16.0*x[4]*x[0]**3*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[1]*x[2] - 2.0*x[5]*x[0]**2*x[1]*x[3] +
4.0*x[5]*x[0]**3*x[1]*x[2] + 2.0*x[5]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**4*x[1]*x[2] -
4.0*x[6]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**6*x[1]*x[2] + 2.0*x[5]*x[0]**4*x[1]*x[3] -
2.0*x[4]*x[0]**6*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 4.0*x[5]*x[0]**6*x[1]*x[2] -
2.0*x[4]*x[0]**6*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[3] - 2.0*x[5]*x[0]**6*x[2]*x[3] +
8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] +
x[4]*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]**2*x[1]*x[2] +
x[4]*x[5]*x[0]**2*x[1]*x[3] - 1.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + x[4]*x[5]*x[0]**2*x[2]*x[3] +
x[4]*x[6]*x[0]**3*x[1]*x[2] + x[4]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[2] +
x[4]*x[6]*x[0]**3*x[2]*x[3] + x[5]*x[6]*x[0]**3*x[1]*x[3] + x[5]*x[6]*x[0]**3*x[2]*x[3] -
2.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3]
- 2.0*x[4]*x[0]**6*x[1]*x[2]*x[3] - 2.0*x[5]*x[0]**6*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] +
x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) + (0.00043402777777777777777777777777778*x[4]**2*
(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 -
8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] +
32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] -
96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] -
28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] -
24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] +
10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] -
8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] -

1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] -
16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] +
6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] -
2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] +
2.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] +
4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] +
16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] + 28.0*x[5]*x[0]**3*x[1]*x[2] +
12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] + 4.0*x[6]*x[0]**3*x[1]*x[2] +
10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] + 6.0*x[5]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[2] -
3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] +
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2)/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +
16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -
16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] +
x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] -
1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2]
+ 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] -
16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] -
32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] +
16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] -
2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) +
(0.00115740740740740740740740740740741*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] -
2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 +
8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 +
32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3]
- 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 +
x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] +
20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] -
40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] +
10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] -
4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] +
x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] -

2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3]
+ 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] -
1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] +
2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] -
1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] +
4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] +
10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] +
2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] -
5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2]*x[3])*(128.0*x[4]*x[0]**3 -
32.0*x[6]*x[0] - 16.0*x[5]*x[3] - 16.0*x[6]*x[3] - 32.0*x[5]*x[0] - 160.0*x[5]*x[0]**2 +
128.0*x[4]*x[0]**4 - 192.0*x[5]*x[0]**3 + 64.0*x[5]*x[0]**4 - 64.0*x[6]*x[0]**3 +
32.0*x[4]*x[0]**6 + 160.0*x[5]*x[0]**5 + 32.0*x[5]*x[0]**6 + 512.0*x[0]**3*x[1] +
256.0*x[0]**3*x[2] + 1024.0*x[0]**4*x[1] + 256.0*x[0]**4*x[2] + 128.0*x[0]**4*x[3] +
512.0*x[0]**6*x[1] + 256.0*x[0]**6*x[2] + 128.0*x[0]**6*x[3] - 256.0*x[0]**3 - 512.0*x[0]**4 -
256.0*x[0]**6 - 256.0*x[0]**3*x[1]**2 - 512.0*x[0]**4*x[1]**2 - 256.0*x[0]**6*x[1]**2 +
16.0*x[4]*x[5]*x[0]**2 - 8.0*x[4]*x[5]*x[0]**3 + 8.0*x[4]*x[6]*x[0]**3 + 16.0*x[5]*x[6]*x[0]**2 +
8.0*x[5]*x[6]*x[0]**3 - 544.0*x[4]*x[0]**3*x[1] - 32.0*x[5]*x[0]*x[1]**2 + 320.0*x[5]*x[0]**2*x[1]
- 128.0*x[4]*x[0]**3*x[2] - 640.0*x[4]*x[0]**4*x[1] + 160.0*x[5]*x[0]**2*x[2] +
384.0*x[5]*x[0]**3*x[1] - 32.0*x[6]*x[0]*x[1]**2 - 80.0*x[5]*x[0]**2*x[3] +
192.0*x[5]*x[0]**3*x[2] - 128.0*x[5]*x[0]**4*x[1] + 128.0*x[6]*x[0]**3*x[1] -
32.0*x[4]*x[0]**4*x[3] - 160.0*x[4]*x[0]**6*x[1] - 320.0*x[5]*x[0]**4*x[2] -
320.0*x[5]*x[0]**5*x[1] + 64.0*x[6]*x[0]**3*x[2] - 32.0*x[4]*x[0]**6*x[2] - 16.0*x[5]*x[0]**4*x[3]
+ 80.0*x[5]*x[0]**5*x[2] - 64.0*x[5]*x[0]**6*x[1] - 64.0*x[6]*x[0]**3*x[3] - 16.0*x[4]*x[0]**6*x[3]
- 64.0*x[5]*x[0]**5*x[3] - 32.0*x[5]*x[0]**6*x[2] - 16.0*x[5]*x[0]**6*x[3] - 16.0*x[5]*x[1]**2*x[3]
- 16.0*x[6]*x[1]**2*x[3] - 512.0*x[0]**3*x[1]*x[2] - 512.0*x[0]**4*x[1]*x[2] -
256.0*x[0]**4*x[1]*x[3] - 64.0*x[0]**4*x[2]*x[3] - 512.0*x[0]**6*x[1]*x[2] -
256.0*x[0]**6*x[1]*x[3] - 128.0*x[0]**6*x[2]*x[3] + 128.0*x[4]*x[0]**3*x[1]**2 -
160.0*x[5]*x[0]**2*x[1]**2 + 32.0*x[4]*x[0]**4*x[1]**2 - 192.0*x[5]*x[0]**3*x[1]**2 +
64.0*x[5]*x[0]**4*x[1]**2 - 64.0*x[6]*x[0]**3*x[1]**2 - 16.0*x[4]*x[0]**6*x[1]**2 +
160.0*x[5]*x[0]**5*x[1]**2 + 32.0*x[5]*x[0]**6*x[1]**2 + 256.0*x[0]**3*x[1]**2*x[2] +
256.0*x[0]**4*x[1]**2*x[2] + 128.0*x[0]**4*x[1]**2*x[3] + 256.0*x[0]**6*x[1]**2*x[2] +
128.0*x[0]**6*x[1]**2*x[3] + 8.0*x[4]*x[5]*x[0] + 8.0*x[4]*x[6]*x[0] - 8.0*x[5]*x[6]*x[0] +
8.0*x[4]*x[5]*x[3] + 8.0*x[4]*x[6]*x[3] - 16.0*x[5]*x[6]*x[3] + 64.0*x[5]*x[0]*x[1] +
16.0*x[5]*x[0]*x[2] + 64.0*x[6]*x[0]*x[1] - 64.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[2] -
64.0*x[6]*x[0]*x[3] + 32.0*x[5]*x[1]*x[3] + 16.0*x[5]*x[2]*x[3] + 32.0*x[6]*x[1]*x[3] +
16.0*x[6]*x[2]*x[3] - 40.0*x[4]*x[5]*x[0]*x[1] - 40.0*x[4]*x[6]*x[0]*x[1] + 16.0*x[4]*x[5]*x[0]*x[3]
+ 16.0*x[5]*x[6]*x[0]*x[1] + 16.0*x[4]*x[6]*x[0]*x[3] - 8.0*x[5]*x[6]*x[0]*x[2] +
8.0*x[5]*x[6]*x[0]*x[3] - 34.0*x[4]*x[5]*x[1]*x[3] - 8.0*x[4]*x[5]*x[2]*x[3] -
34.0*x[4]*x[6]*x[1]*x[3] - 8.0*x[4]*x[6]*x[2]*x[3] + 32.0*x[5]*x[6]*x[1]*x[3] +
16.0*x[5]*x[6]*x[2]*x[3] - 32.0*x[5]*x[0]*x[1]*x[2] + 128.0*x[5]*x[0]*x[1]*x[3] -
32.0*x[6]*x[0]*x[1]*x[2] + 32.0*x[5]*x[0]*x[2]*x[3] + 128.0*x[6]*x[0]*x[1]*x[3] +
32.0*x[6]*x[0]*x[2]*x[3] - 32.0*x[5]*x[1]*x[2]*x[3] - 32.0*x[6]*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]**2 - 104.0*x[4]*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[5]*x[0]**2*x[2] -
8.0*x[4]*x[5]*x[0]**3*x[1] + 2.0*x[4]*x[6]*x[0]*x[1]**2 + 8.0*x[4]*x[5]*x[0]**2*x[3] +
8.0*x[4]*x[5]*x[0]**3*x[2] + 24.0*x[4]*x[5]*x[0]**4*x[1] - 40.0*x[4]*x[6]*x[0]**3*x[1] -
8.0*x[5]*x[6]*x[0]*x[1]**2 - 32.0*x[5]*x[6]*x[0]**2*x[1] - 8.0*x[4]*x[6]*x[0]**3*x[2] +
8.0*x[5]*x[6]*x[0]**2*x[2] - 16.0*x[5]*x[6]*x[0]**3*x[1] + 8.0*x[4]*x[6]*x[0]**3*x[3] +
32.0*x[5]*x[6]*x[0]**2*x[3] - 8.0*x[5]*x[6]*x[0]**3*x[2] + 8.0*x[5]*x[6]*x[0]**3*x[3] +
8.0*x[4]*x[5]*x[1]**2*x[3] + 8.0*x[4]*x[6]*x[1]**2*x[3] - 16.0*x[5]*x[6]*x[1]**2*x[3] +

544.0*x[4]*x[0]**3*x[1]*x[2] + 16.0*x[5]*x[0]*x[1]**2*x[2] - 320.0*x[5]*x[0]**2*x[1]*x[2] +
96.0*x[4]*x[0]**4*x[1]*x[2] - 64.0*x[5]*x[0]*x[1]**2*x[3] + 160.0*x[5]*x[0]**2*x[1]*x[3] -
384.0*x[5]*x[0]**3*x[1]*x[2] + 16.0*x[6]*x[0]*x[1]**2*x[2] + 160.0*x[4]*x[0]**4*x[1]*x[3] +
80.0*x[5]*x[0]**2*x[2]*x[3] + 640.0*x[5]*x[0]**4*x[1]*x[2] - 64.0*x[6]*x[0]*x[1]**2*x[3] -
128.0*x[6]*x[0]**3*x[1]*x[2] + 160.0*x[4]*x[0]**6*x[1]*x[2] + 32.0*x[5]*x[0]**4*x[1]*x[3] -
160.0*x[5]*x[0]**5*x[1]*x[2] + 128.0*x[6]*x[0]**3*x[1]*x[3] + 80.0*x[4]*x[0]**6*x[1]*x[3] +
80.0*x[5]*x[0]**4*x[2]*x[3] + 128.0*x[5]*x[0]**5*x[1]*x[3] + 64.0*x[5]*x[0]**6*x[1]*x[2] +
64.0*x[6]*x[0]**3*x[2]*x[3] + 16.0*x[4]*x[0]**6*x[2]*x[3] - 32.0*x[5]*x[0]**5*x[2]*x[3] +
32.0*x[5]*x[0]**6*x[1]*x[3] + 16.0*x[5]*x[0]**6*x[2]*x[3] + 16.0*x[5]*x[1]**2*x[2]*x[3] +
16.0*x[6]*x[1]**2*x[2]*x[3] + 128.0*x[0]**4*x[1]*x[2]*x[3] + 256.0*x[0]**6*x[1]*x[2]*x[3] -
32.0*x[4]*x[5]*x[0]**2*x[1]**2 - 92.0*x[4]*x[5]*x[0]**3*x[1]**2 - 4.0*x[4]*x[6]*x[0]**3*x[1]**2 +
16.0*x[5]*x[6]*x[0]**2*x[1]**2 + 30.0*x[4]*x[5]*x[0]**5*x[1]**2 + 8.0*x[5]*x[6]*x[0]**3*x[1]**2 -
128.0*x[4]*x[0]**3*x[1]**2*x[2]+160.0*x[5]*x[0]**2*x[1]**2*x[2]+96.0*x[4]*x[0]**4*x[1]**2*x[2]
- 80.0*x[5]*x[0]**2*x[1]**2*x[3] + 192.0*x[5]*x[0]**3*x[1]**2*x[2] -8.0*x[4]*x[0]**4*x[1]**2*x[3]
- 320.0*x[5]*x[0]**4*x[1]**2*x[2]+64.0*x[6]*x[0]**3*x[1]**2*x[2]+16.0*x[4]*x[0]**6*x[1]**2*x[2]
- 16.0*x[5]*x[0]**4*x[1]**2*x[3] + 80.0*x[5]*x[0]**5*x[1]**2*x[2] -64.0*x[6]*x[0]**3*x[1]**2*x[3]
+ 8.0*x[4]*x[0]**6*x[1]**2*x[3] - 64.0*x[5]*x[0]**5*x[1]**2*x[3] - 32.0*x[5]*x[0]**6*x[1]**2*x[2]
- 16.0*x[5]*x[0]**6*x[1]**2*x[3] - 64.0*x[0]**4*x[1]**2*x[2]*x[3]-128.0*x[0]**6*x[1]**2*x[2]*x[3]
+3.0*x[4]*x[5]*x[6]*x[0]**2*x[1]**2+32.0*x[4]*x[5]*x[0]**2*x[1]**2*x[2]-
16.0*x[4]*x[5]*x[0]**2*x[1]**2*x[3]+92.0*x[4]*x[5]*x[0]**3*x[1]**2*x[2]-
96.0*x[4]*x[5]*x[0]**4*x[1]**2*x[2] + 4.0*x[4]*x[6]*x[0]**3*x[1]**2*x[2] +
8.0*x[5]*x[6]*x[0]**2*x[1]**2*x[2] + 30.0*x[4]*x[5]*x[0]**5*x[1]**2*x[2] -
4.0*x[4]*x[6]*x[0]**3*x[1]**2*x[3] + 32.0*x[5]*x[6]*x[0]**2*x[1]**2*x[3] -
8.0*x[5]*x[6]*x[0]**3*x[1]**2*x[2] - 12.0*x[4]*x[5]*x[0]**5*x[1]**2*x[3] +
8.0*x[5]*x[6]*x[0]**3*x[1]**2*x[3] + 80.0*x[5]*x[0]**2*x[1]**2*x[2]*x[3] -
24.0*x[4]*x[0]**4*x[1]**2*x[2]*x[3] + 80.0*x[5]*x[0]**4*x[1]**2*x[2]*x[3] +
64.0*x[6]*x[0]**3*x[1]**2*x[2]*x[3] - 8.0*x[4]*x[0]**6*x[1]**2*x[2]*x[3] -
32.0*x[5]*x[0]**5*x[1]**2*x[2]*x[3] + 16.0*x[5]*x[0]**6*x[1]**2*x[2]*x[3] -
3.0*x[4]*x[5]*x[6]*x[1]*x[3] + 6.0*x[4]*x[5]*x[0]*x[1]*x[2] - 80.0*x[4]*x[5]*x[0]*x[1]*x[3] +
6.0*x[4]*x[6]*x[0]*x[1]*x[2] - 80.0*x[4]*x[6]*x[0]*x[1]*x[3] + 16.0*x[5]*x[6]*x[0]*x[1]*x[2] -
16.0*x[5]*x[6]*x[0]*x[1]*x[3] - 40.0*x[5]*x[6]*x[0]*x[2]*x[3] + 34.0*x[4]*x[5]*x[1]*x[2]*x[3] +
34.0*x[4]*x[6]*x[1]*x[2]*x[3] - 32.0*x[5]*x[6]*x[1]*x[2]*x[3] - 64.0*x[5]*x[0]*x[1]*x[2]*x[3] -
64.0*x[6]*x[0]*x[1]*x[2]*x[3] - 3.0*x[4]*x[5]*x[6]*x[0]*x[1]**2 - 6.0*x[4]*x[5]*x[6]*x[1]**2*x[3] +
6.0*x[4]*x[5]*x[0]*x[1]**2*x[2]+104.0*x[4]*x[5]*x[0]**2*x[1]*x[2]
+4.0*x[4]*x[5]*x[0]*x[1]**2*x[3] - 52.0*x[4]*x[5]*x[0]**2*x[1]*x[3] +
8.0*x[4]*x[5]*x[0]**3*x[1]*x[2] + 6.0*x[4]*x[6]*x[0]*x[1]**2*x[2] -
8.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 72.0*x[4]*x[5]*x[0]**4*x[1]*x[2] +
4.0*x[4]*x[6]*x[0]*x[1]**2*x[3] + 40.0*x[4]*x[6]*x[0]**3*x[1]*x[2] -
8.0*x[5]*x[6]*x[0]*x[1]**2*x[2] - 16.0*x[5]*x[6]*x[0]**2*x[1]*x[2] -
6.0*x[4]*x[5]*x[0]**4*x[1]*x[3] + 30.0*x[4]*x[5]*x[0]**5*x[1]*x[2] -
40.0*x[4]*x[6]*x[0]**3*x[1]*x[3] + 8.0*x[5]*x[6]*x[0]*x[1]**2*x[3] -
64.0*x[5]*x[6]*x[0]**2*x[1]*x[3] + 16.0*x[5]*x[6]*x[0]**3*x[1]*x[2] -
8.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 16.0*x[5]*x[6]*x[0]**2*x[2]*x[3] -
16.0*x[5]*x[6]*x[0]**3*x[1]*x[3] - 8.0*x[5]*x[6]*x[0]**3*x[2]*x[3] -
8.0*x[4]*x[5]*x[1]**2*x[2]*x[3] - 8.0*x[4]*x[6]*x[1]**2*x[2]*x[3] +
16.0*x[5]*x[6]*x[1]**2*x[2]*x[3] + 32.0*x[5]*x[0]*x[1]**2*x[2]*x[3] -
160.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 24.0*x[4]*x[0]**4*x[1]*x[2]*x[3] +
32.0*x[6]*x[0]*x[1]**2*x[2]*x[3] - 160.0*x[5]*x[0]**4*x[1]*x[2]*x[3] -
128.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 80.0*x[4]*x[0]**6*x[1]*x[2]*x[3] +
64.0*x[5]*x[0]**5*x[1]*x[2]*x[3] - 32.0*x[5]*x[0]**6*x[1]*x[2]*x[3] -
3.0*x[4]*x[5]*x[6]*x[0]*x[1]**2*x[2] + 3.0*x[4]*x[5]*x[6]*x[0]**2*x[1]*x[2] +

```
6.0*x[4]*x[5]*x[6]*x[1]**2*x[2]*x[3] + 12.0*x[4]*x[5]*x[0]*x[1]**2*x[2]*x[3] +
52.0*x[4]*x[5]*x[0]**2*x[1]*x[2]*x[3] + 12.0*x[4]*x[6]*x[0]*x[1]**2*x[2]*x[3] +
18.0*x[4]*x[5]*x[0]**4*x[1]*x[2]*x[3] + 40.0*x[4]*x[6]*x[0]**3*x[1]*x[2]*x[3] -
40.0*x[5]*x[6]*x[0]*x[1]**2*x[2]*x[3] - 32.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] -
12.0*x[4]*x[5]*x[0]**5*x[1]*x[2]*x[3] + 16.0*x[5]*x[6]*x[0]**3*x[1]*x[2]*x[3] +
3.0*x[4]*x[5]*x[6]*x[0]**2*x[1]**2*x[2] + 6.0*x[4]*x[5]*x[6]*x[0]**2*x[1]**2*x[3] +
16.0*x[4]*x[5]*x[0]**2*x[1]**2*x[2]*x[3] + 24.0*x[4]*x[5]*x[0]**4*x[1]**2*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]**3*x[1]**2*x[2]*x[3] + 16.0*x[5]*x[6]*x[0]**2*x[1]**2*x[2]*x[3] -
12.0*x[4]*x[5]*x[0]**5*x[1]**2*x[2]*x[3] - 8.0*x[5]*x[6]*x[0]**3*x[1]**2*x[2]*x[3] -
3.0*x[4]*x[5]*x[6]*x[0]*x[1]*x[2] + 3.0*x[4]*x[5]*x[6]*x[0]*x[1]*x[3] +
3.0*x[4]*x[5]*x[6]*x[1]*x[2]*x[3] + 12.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
12.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 80.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3] +
6.0*x[4]*x[5]*x[6]*x[0]**2*x[1]**2*x[2]*x[3] - 9.0*x[4]*x[5]*x[6]*x[0]*x[1]*x[2]*x[3] -
12.0*x[4]*x[5]*x[6]*x[0]*x[1]**2*x[2]*x[3] + 6.0*x[4]*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3]))
/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 +
32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3]
+ 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] -
16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1]
+ 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] +
16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2]
+ x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] -
4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] +
x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] +
2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3]
- 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] +
4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] +
16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] -
4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] -
4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] +
16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] +
x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] +
x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] -
2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3]
+ 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) + (0.0004340277777777777777777777777778*x[5]**2*x[0]*
(8.0*x[4]*x[0]**4 - 10.0*x[6]*x[3] - 8.0*x[4]*x[0]**2 - 16.0*x[4]*x[0]**3 - 8.0*x[6]*x[0] +
16.0*x[6]*x[0]**2 + 8.0*x[6]*x[0]**3 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] + 64.0*x[0]**3*x[1] -
16.0*x[0]**2*x[3] + 160.0*x[0]**3*x[2] - 160.0*x[0]**4*x[2] + 16.0*x[0]**6*x[1] -
64.0*x[0]**5*x[3] - 32.0*x[0]**6*x[2] - 16.0*x[0]**6*x[3] - 32.0*x[0]**2 - 160.0*x[0]**3 +
160.0*x[0]**5 + 32.0*x[0]**6 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] -
32.0*x[4]*x[0]**3*x[1] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**3*x[2] - 24.0*x[4]*x[0]**4*x[2] +
10.0*x[4]*x[0]**5*x[1] + 4.0*x[6]*x[0]**3*x[1] - 2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] +
32.0*x[6]*x[0]**2*x[3] - 8.0*x[6]*x[0]**3*x[2] + 8.0*x[6]*x[0]**3*x[3] - 16.0*x[0]**2*x[1]*x[2] +
8.0*x[0]**2*x[1]*x[3] - 64.0*x[0]**3*x[1]*x[2] + 16.0*x[0]**2*x[2]*x[3] + 64.0*x[0]**4*x[1]*x[2] -
40.0*x[0]**5*x[1]*x[2] + 40.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[1]*x[2] - 8.0*x[0]**6*x[1]*x[3] +
16.0*x[0]**6*x[2]*x[3] - 1.0*x[4]*x[6]*x[3] + 2.0*x[6]*x[0]*x[1] - 2.0*x[6]*x[0]*x[2] +
4.0*x[6]*x[1]*x[3] + 10.0*x[6]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1] - 1.0*x[4]*x[6]*x[0]*x[2] +
x[4]*x[6]*x[0]*x[3] - 2.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] -
```

20.0*x[6]*x[0]*x[2]*x[3] - 4.0*x[6]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] +
x[4]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] +
32.0*x[4]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**2*x[2]*x[3] - 32.0*x[4]*x[0]**4*x[1]*x[2] -
4.0*x[6]*x[0]**2*x[1]*x[2] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[2] +
6.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] + 4.0*x[6]*x[0]**3*x[1]*x[3] -
4.0*x[4]*x[0]**5*x[2]*x[3] - 8.0*x[6]*x[0]**3*x[2]*x[3] - 8.0*x[0]**2*x[1]*x[2]*x[3] -
16.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] + 8.0*x[0]**6*x[1]*x[2]*x[3] -
1.0*x[4]*x[6]*x[0]*x[1]*x[2] - 3.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[2]*x[3] +
8.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3]+8.0*x[4]*x[0]**4*x[1]*x[2]*x[3]
- 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3]-4.0*x[6]*x[0]**3*x[1]*x[2]*x[3]
+ 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3])**2)/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) - (0.000868055555555555555555555555556*x[0]**8*
(32.0*x[5]*x[0] + 32.0*x[6]*x[0] + 32.0*x[5]*x[3] + 32.0*x[6]*x[3] + 32.0*x[4]*x[0]**3 +
64.0*x[5]*x[0]**2 + 32.0*x[5]*x[0]**3 - 128.0*x[0]**3*x[1] - 512.0*x[0]**3*x[2] -
128.0*x[0]**4*x[3] + 512.0*x[0]**3 + 512.0*x[0]**4 + 64.0*x[4]*x[0]**3*x[1] +
32.0*x[5]*x[0]**2*x[1] - 32.0*x[4]*x[0]**3*x[2] + 32.0*x[4]*x[0]**4*x[1] - 64.0*x[5]*x[0]**2*x[2] +
64.0*x[5]*x[0]**3*x[1] + 32.0*x[4]*x[0]**4*x[2] + 32.0*x[5]*x[0]**2*x[3] - 32.0*x[5]*x[0]**3*x[2]
+ 32.0*x[5]*x[0]**4*x[1] + 32.0*x[5]*x[0]**4*x[2] + 128.0*x[0]**3*x[1]*x[2] -
128.0*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[5]*x[3] + 2.0*x[4]*x[6]*x[3] + 2.0*x[5]*x[6]*x[3] +
64.0*x[5]*x[0]*x[3] + 64.0*x[6]*x[0]*x[3] - 8.0*x[5]*x[1]*x[3] - 32.0*x[5]*x[2]*x[3] -
8.0*x[6]*x[1]*x[3] - 32.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1] + 2.0*x[4]*x[5]*x[0]*x[2] +
2.0*x[4]*x[6]*x[0]*x[1] + 2.0*x[4]*x[6]*x[0]*x[2] + 2.0*x[5]*x[6]*x[0]*x[1] +
2.0*x[5]*x[6]*x[0]*x[2] + 4.0*x[4]*x[5]*x[1]*x[3] - 2.0*x[4]*x[5]*x[2]*x[3] +
4.0*x[4]*x[6]*x[1]*x[3] - 2.0*x[4]*x[6]*x[2]*x[3] + 4.0*x[5]*x[6]*x[1]*x[3] - 2.0*x[5]*x[6]*x[2]*x[3]
- 8.0*x[5]*x[0]*x[1]*x[2] - 8.0*x[6]*x[0]*x[1]*x[2] + 8.0*x[5]*x[1]*x[2]*x[3] +
8.0*x[6]*x[1]*x[2]*x[3] - 64.0*x[4]*x[0]**3*x[1]*x[2] - 32.0*x[5]*x[0]**2*x[1]*x[2] +
32.0*x[4]*x[0]**4*x[1]*x[2] + 16.0*x[5]*x[0]**2*x[1]*x[3] - 64.0*x[5]*x[0]**3*x[1]*x[2] -

8.0*x[4]*x[0]**4*x[1]*x[3] - 32.0*x[5]*x[0]**2*x[2]*x[3] + 32.0*x[5]*x[0]**4*x[1]*x[2] -
8.0*x[4]*x[0]**4*x[2]*x[3] - 8.0*x[5]*x[0]**4*x[1]*x[3] - 8.0*x[5]*x[0]**4*x[2]*x[3] +
32.0*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[1]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[5]*x[0]*x[2]*x[3] + 4.0*x[4]*x[6]*x[0]*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2] + 4.0*x[4]*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[3] +
4.0*x[5]*x[6]*x[0]*x[2]*x[3] - 4.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[1]*x[2]*x[3] -
4.0*x[5]*x[6]*x[1]*x[2]*x[3] - 16.0*x[5]*x[0]*x[1]*x[2]*x[3] - 16.0*x[6]*x[0]*x[1]*x[2]*x[3] -
16.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])*(8.0*x[5]*x[0] +
10.0*x[5]*x[3] - 8.0*x[4]*x[0]**2 + 8.0*x[5]*x[0]**2 + 8.0*x[4]*x[0]**4 + 8.0*x[5]*x[0]**3 +
16.0*x[6]*x[0]**2 + 8.0*x[5]*x[0]**4 + 16.0*x[0]**2*x[1] + 32.0*x[0]**2*x[2] - 16.0*x[0]**2*x[3] -
32.0*x[0]**4*x[1] - 128.0*x[0]**4*x[2] - 32.0*x[0]**4*x[3] - 64.0*x[0]**5*x[3] - 32.0*x[0]**2 +
128.0*x[0]**4 + 160.0*x[0]**5 - 8.0*x[4]*x[0]**2*x[1] + 8.0*x[4]*x[0]**2*x[2] -
4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**4*x[1] - 8.0*x[5]*x[0]**2*x[2] + 4.0*x[5]*x[0]**3*x[1] -
8.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] + 4.0*x[5]*x[0]**2*x[3] - 8.0*x[5]*x[0]**3*x[2] +
16.0*x[5]*x[0]**4*x[1] - 2.0*x[4]*x[0]**4*x[3] + 10.0*x[4]*x[0]**5*x[2] - 8.0*x[5]*x[0]**4*x[2] +
10.0*x[5]*x[0]**5*x[1] + 32.0*x[6]*x[0]**2*x[3] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] -
16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] + 16.0*x[0]**2*x[2]*x[3] + 32.0*x[0]**4*x[1]*x[2] +
8.0*x[0]**4*x[1]*x[3] - 40.0*x[0]**5*x[1]*x[2] + 32.0*x[0]**4*x[2]*x[3] + x[4]*x[5]*x[3] -
2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] + 16.0*x[5]*x[0]*x[3] + 16.0*x[6]*x[0]*x[3] -
4.0*x[5]*x[1]*x[3] - 10.0*x[5]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] +
x[4]*x[6]*x[0]*x[3] + x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] -
2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3]
- 16.0*x[6]*x[0]*x[2]*x[3] + 4.0*x[5]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1] +
x[4]*x[6]*x[0]**2*x[2] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] +
8.0*x[4]*x[0]**2*x[1]*x[2] - 4.0*x[4]*x[0]**2*x[1]*x[3] + 4.0*x[4]*x[0]**2*x[2]*x[3] -
16.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] -
4.0*x[4]*x[0]**4*x[1]*x[3] + 10.0*x[4]*x[0]**5*x[1]*x[2] - 4.0*x[5]*x[0]**2*x[2]*x[3] -
16.0*x[5]*x[0]**4*x[1]*x[2] + 2.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] + 10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[4]*x[0]**5*x[2]*x[3] +
2.0*x[5]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**2*x[1]*x[2]*x[3] - 8.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]*x[1]*x[3] - 1.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] -
1.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] +
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3]
+ 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] -
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]))/((16.0*x[5]*x[0] + 16.0*x[6]*x[0] + 16.0*x[5]*x[3] +
16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 + 16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] -
256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 + 256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] +
16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] + 16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] +
32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2] + 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2]
+ 16.0*x[5]*x[0]**4*x[1] + 16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] -
64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] +
32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] - 16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] -

16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] + x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] - 32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] + 8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] - 16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] + 2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) - (0.00347222222222222222222222222222222*x[4]*(x[1] - 1.0)*(16.0*x[4]*x[0]**3 - 4.0*x[6]*x[0] - 2.0*x[5]*x[3] - 2.0*x[6]*x[3] - 4.0*x[5]*x[0] - 20.0*x[5]*x[0]**2 + 16.0*x[4]*x[0]**4 - 24.0*x[5]*x[0]**3 + 8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0]**3 + 4.0*x[4]*x[0]**6 + 20.0*x[5]*x[0]**5 + 4.0*x[5]*x[0]**6 + 32.0*x[0]**3*x[2] + 32.0*x[0]**4*x[2] + 16.0*x[0]**4*x[3] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 32.0*x[0]**3 - 64.0*x[0]**4 - 32.0*x[0]**6 + 2.0*x[4]*x[5]*x[0]**2 - 1.0*x[4]*x[5]*x[0]**3 + x[4]*x[6]*x[0]**3 + 2.0*x[5]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**3 - 16.0*x[4]*x[0]**3*x[2] + 20.0*x[5]*x[0]**2*x[2] - 10.0*x[5]*x[0]**2*x[3] + 24.0*x[5]*x[0]**3*x[2] - 4.0*x[4]*x[0]**4*x[3] - 40.0*x[5]*x[0]**4*x[2] + 8.0*x[6]*x[0]**3*x[2] - 4.0*x[4]*x[0]**6*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] - 2.0*x[4]*x[0]**6*x[3] - 8.0*x[5]*x[0]**5*x[3] - 4.0*x[5]*x[0]**6*x[2] - 2.0*x[5]*x[0]**6*x[3] - 8.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[2]*x[3] + x[4]*x[5]*x[0] + x[4]*x[6]*x[0] - 1.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[3] + x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] + 2.0*x[5]*x[0]*x[2] - 8.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 8.0*x[6]*x[0]*x[3] + 2.0*x[5]*x[2]*x[3] + 2.0*x[6]*x[2]*x[3] + 2.0*x[4]*x[5]*x[0]*x[3] + 2.0*x[4]*x[6]*x[0]*x[3] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] - 1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[2]*x[3] + 4.0*x[5]*x[0]*x[2]*x[3] + 4.0*x[6]*x[0]*x[2]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[2] + x[4]*x[5]*x[0]**2*x[3] + x[4]*x[5]*x[0]**3*x[2] - 1.0*x[4]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**2*x[2] + x[4]*x[6]*x[0]**3*x[3] + 4.0*x[5]*x[6]*x[0]**2*x[3] - 1.0*x[5]*x[6]*x[0]**3*x[2] + x[5]*x[6]*x[0]**3*x[3] + 10.0*x[5]*x[0]**2*x[2]*x[3] + 10.0*x[5]*x[0]**4*x[2]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[4]*x[0]**6*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] + 2.0*x[5]*x[0]**6*x[2]*x[3] - 5.0*x[5]*x[6]*x[0]*x[2]*x[3] - 1.0*x[4]*x[5]*x[0]**2*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]**3*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]**2*x[2]*x[3]-1.0*x[5]*x[6]*x[0]**3*x[2]*x[3]) *(8.0*x[5]*x[0]**4 - 8.0*x[6]*x[0] - 6.0*x[5]*x[3] - 6.0*x[6]*x[3] - 24.0*x[5]*x[0]**2 - 8.0*x[5]*x[0]**3 - 8.0*x[5]*x[0] - 8.0*x[6]*x[0]**3 + 96.0*x[0]**3*x[2] - 32.0*x[0]**4*x[1] + 32.0*x[0]**4*x[2] + 32.0*x[0]**4*x[3] - 16.0*x[0]**6*x[1] + 32.0*x[0]**6*x[2] + 16.0*x[0]**6*x[3] - 96.0*x[0]**3 - 128.0*x[0]**4 - 32.0*x[0]**6 - 16.0*x[5]*x[0]**2*x[1] + 24.0*x[5]*x[0]**2*x[2] - 28.0*x[5]*x[0]**3*x[1] - 12.0*x[5]*x[0]**2*x[3] + 8.0*x[5]*x[0]**3*x[2] - 4.0*x[6]*x[0]**3*x[1] - 24.0*x[5]*x[0]**4*x[2] + 10.0*x[5]*x[0]**5*x[1] + 8.0*x[6]*x[0]**3*x[2] - 2.0*x[5]*x[0]**4*x[3] + 10.0*x[5]*x[0]**5*x[2] - 8.0*x[6]*x[0]**3*x[3] + 32.0*x[0]**4*x[1]*x[2] + 8.0*x[0]**4*x[1]*x[3] - 8.0*x[0]**4*x[2]*x[3] + 16.0*x[0]**6*x[1]*x[2] + 8.0*x[0]**6*x[1]*x[3] - 16.0*x[0]**6*x[2]*x[3] - 1.0*x[5]*x[6]*x[3] - 2.0*x[5]*x[0]*x[1] + 2.0*x[5]*x[0]*x[2] - 2.0*x[6]*x[0]*x[1] - 16.0*x[5]*x[0]*x[3] + 2.0*x[6]*x[0]*x[2] - 16.0*x[6]*x[0]*x[3] + 6.0*x[5]*x[2]*x[3] + 6.0*x[6]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1] - 1.0*x[5]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[3] - 2.0*x[5]*x[6]*x[1]*x[3] + x[5]*x[6]*x[2]*x[3] + 2.0*x[5]*x[0]*x[1]*x[2] - 4.0*x[5]*x[0]*x[1]*x[3] + 2.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[0]*x[2]*x[3] - 4.0*x[6]*x[0]*x[1]*x[3] +

4.0*x[6]*x[0]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1] + x[5]*x[6]*x[0]**2*x[2] +
16.0*x[5]*x[0]**2*x[1]*x[2] - 8.0*x[5]*x[0]**2*x[1]*x[3] + 28.0*x[5]*x[0]**3*x[1]*x[2] +
12.0*x[5]*x[0]**2*x[2]*x[3] - 32.0*x[5]*x[0]**4*x[1]*x[2] + 4.0*x[6]*x[0]**3*x[1]*x[2] +
10.0*x[5]*x[0]**5*x[1]*x[2] - 4.0*x[6]*x[0]**3*x[1]*x[3] + 6.0*x[5]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**5*x[1]*x[3] + 8.0*x[6]*x[0]**3*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[2]*x[3] -
8.0*x[0]**4*x[1]*x[2]*x[3] - 8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[5]*x[6]*x[0]*x[1]*x[2] -
3.0*x[5]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[2]*x[3] + 4.0*x[5]*x[0]*x[1]*x[2]*x[3] +
4.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[5]*x[6]*x[0]**2*x[1]*x[2] + 2.0*x[5]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[2]*x[3] + 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3]+8.0*x[5]*x[0]**4*x[1]*x[2]*x[3]
+ 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3]))/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3]-4.0*x[5]*x[0]**4*x[1]*x[2]*x[3]+2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3]
+ 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2) -
(0.0034722222222222222222222222222222*x[5]*x[0]**3*(x[2] - 1.0)*(4.0*x[5] + 4.0*x[6] + x[4]*x[5]
+ x[4]*x[6] - 1.0*x[5]*x[6] + 20.0*x[5]*x[0] - 2.0*x[5]*x[1] - 2.0*x[6]*x[1] + 8.0*x[5]*x[3] +
8.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 - 16.0*x[5]*x[0]**2 - 16.0*x[5]*x[0]**3 + 16.0*x[6]*x[0]**2 +
4.0*x[4]*x[0]**5 + 20.0*x[5]*x[0]**4 + 4.0*x[5]*x[0]**5 - 32.0*x[0]**3*x[1] - 16.0*x[0]**3*x[3] -
16.0*x[0]**5*x[1] - 32.0*x[0]**5*x[3] + 64.0*x[0]**3 + 64.0*x[0]**5 - 1.0*x[4]*x[5]*x[0]**2 +
x[4]*x[6]*x[0]**2 + x[5]*x[6]*x[0]**2 + 16.0*x[4]*x[0]**3*x[1] + 4.0*x[5]*x[0]**2*x[1] -
4.0*x[6]*x[0]**2*x[1] - 4.0*x[4]*x[0]**3*x[3] + 8.0*x[4]*x[0]**5*x[1] + 10.0*x[5]*x[0]**4*x[1] +
4.0*x[5]*x[0]**3*x[3] + 8.0*x[5]*x[0]**5*x[1] + 16.0*x[6]*x[0]**2*x[3] - 2.0*x[4]*x[0]**5*x[3] -
8.0*x[5]*x[0]**4*x[3] - 2.0*x[5]*x[0]**5*x[3] + 8.0*x[0]**3*x[1]*x[3] + 8.0*x[0]**5*x[1]*x[3] +
2.0*x[4]*x[5]*x[0] + 2.0*x[5]*x[6]*x[0] + x[4]*x[5]*x[1] + x[4]*x[6]*x[1] + 2.0*x[4]*x[5]*x[3] +
2.0*x[4]*x[6]*x[3] - 2.0*x[5]*x[6]*x[3] - 8.0*x[5]*x[0]*x[1] + 10.0*x[5]*x[0]*x[3] -
4.0*x[5]*x[1]*x[3] - 4.0*x[6]*x[1]*x[3] + 4.0*x[4]*x[5]*x[0]*x[1] + x[4]*x[5]*x[0]*x[3] +
x[5]*x[6]*x[0]*x[1] + 4.0*x[5]*x[6]*x[0]*x[3] + 2.0*x[4]*x[5]*x[1]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
4.0*x[5]*x[0]*x[1]*x[3] - 2.0*x[4]*x[5]*x[0]**2*x[1] + 2.0*x[4]*x[6]*x[0]**2*x[1] +
2.0*x[5]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[3] + x[5]*x[6]*x[0]**2*x[3] -
4.0*x[4]*x[0]**3*x[1]*x[3] - 4.0*x[6]*x[0]**2*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**5*x[1]*x[3] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] +

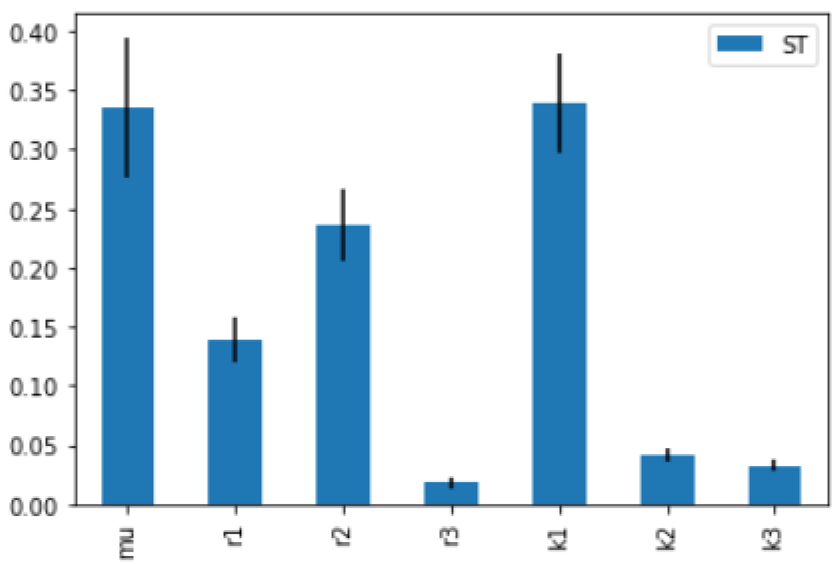
```
2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] +
2.0*x[5]*x[6]*x[0]**2*x[1]*x[3])*(8.0*x[4]*x[0]**4 - 10.0*x[6]*x[3] - 8.0*x[4]*x[0]**2 -
16.0*x[4]*x[0]**3 - 8.0*x[6]*x[0] + 16.0*x[6]*x[0]**2 + 8.0*x[6]*x[0]**3 + 16.0*x[0]**2*x[1] +
32.0*x[0]**2*x[2] + 64.0*x[0]**3*x[1] - 16.0*x[0]**2*x[3] + 160.0*x[0]**3*x[2] -
160.0*x[0]**4*x[2] + 16.0*x[0]**6*x[1] - 64.0*x[0]**5*x[3] - 32.0*x[0]**6*x[2] - 16.0*x[0]**6*x[3]
- 32.0*x[0]**2 - 160.0*x[0]**3 + 160.0*x[0]**5 + 32.0*x[0]**6 - 8.0*x[4]*x[0]**2*x[1] +
8.0*x[4]*x[0]**2*x[2] - 32.0*x[4]*x[0]**3*x[1] - 4.0*x[4]*x[0]**2*x[3] + 16.0*x[4]*x[0]**3*x[2] -
24.0*x[4]*x[0]**4*x[2] + 10.0*x[4]*x[0]**5*x[1] + 4.0*x[6]*x[0]**3*x[1] - 2.0*x[4]*x[0]**4*x[3] +
10.0*x[4]*x[0]**5*x[2] + 32.0*x[6]*x[0]**2*x[3] - 8.0*x[6]*x[0]**3*x[2] + 8.0*x[6]*x[0]**3*x[3] -
16.0*x[0]**2*x[1]*x[2] + 8.0*x[0]**2*x[1]*x[3] - 64.0*x[0]**3*x[1]*x[2] + 16.0*x[0]**2*x[2]*x[3] +
64.0*x[0]**4*x[1]*x[2] - 40.0*x[0]**5*x[1]*x[2] + 40.0*x[0]**4*x[2]*x[3] - 16.0*x[0]**6*x[1]*x[2] -
8.0*x[0]**6*x[1]*x[3] + 16.0*x[0]**6*x[2]*x[3] - 1.0*x[4]*x[6]*x[3] + 2.0*x[6]*x[0]*x[1] -
2.0*x[6]*x[0]*x[2] + 4.0*x[6]*x[1]*x[3] + 10.0*x[6]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1] -
1.0*x[4]*x[6]*x[0]*x[2] + x[4]*x[6]*x[0]*x[3] - 2.0*x[4]*x[6]*x[1]*x[3] + x[4]*x[6]*x[2]*x[3] +
2.0*x[6]*x[0]*x[1]*x[2] - 20.0*x[6]*x[0]*x[2]*x[3] - 4.0*x[6]*x[1]*x[2]*x[3] +
x[4]*x[6]*x[0]**2*x[1] + x[4]*x[6]*x[0]**2*x[2] + 8.0*x[4]*x[0]**2*x[1]*x[2] -
4.0*x[4]*x[0]**2*x[1]*x[3] + 32.0*x[4]*x[0]**3*x[1]*x[2] + 4.0*x[4]*x[0]**2*x[2]*x[3] -
32.0*x[4]*x[0]**4*x[1]*x[2] - 4.0*x[6]*x[0]**2*x[1]*x[2] + 10.0*x[4]*x[0]**5*x[1]*x[2] -
4.0*x[6]*x[0]**3*x[1]*x[2] + 6.0*x[4]*x[0]**4*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[3] +
4.0*x[6]*x[0]**3*x[1]*x[3] - 4.0*x[4]*x[0]**5*x[2]*x[3] - 8.0*x[6]*x[0]**3*x[2]*x[3] -
8.0*x[0]**2*x[1]*x[2]*x[3] - 16.0*x[0]**4*x[1]*x[2]*x[3] + 16.0*x[0]**5*x[1]*x[2]*x[3] +
8.0*x[0]**6*x[1]*x[2]*x[3] - 1.0*x[4]*x[6]*x[0]*x[1]*x[2] - 3.0*x[4]*x[6]*x[0]*x[2]*x[3] +
2.0*x[4]*x[6]*x[1]*x[2]*x[3] + 8.0*x[6]*x[0]*x[1]*x[2]*x[3] + x[4]*x[6]*x[0]**2*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]**2*x[1]*x[3] + 2.0*x[4]*x[6]*x[0]**2*x[2]*x[3] +
4.0*x[4]*x[0]**2*x[1]*x[2]*x[3] + 8.0*x[4]*x[0]**4*x[1]*x[2]*x[3] -
8.0*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[0]**5*x[1]*x[2]*x[3] - 4.0*x[6]*x[0]**3*x[1]*x[2]*x[3]
+ 2.0*x[4]*x[6]*x[0]**2*x[1]*x[2]*x[3] - 4.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3]))/((16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3]-4.0*x[5]*x[0]**4*x[1]*x[2]*x[3]+2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3]
+ 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])**2))*(16.0*x[5]*x[0] +
16.0*x[6]*x[0] + 16.0*x[5]*x[3] + 16.0*x[6]*x[3] + 16.0*x[4]*x[0]**3 + 32.0*x[5]*x[0]**2 +
```

```
16.0*x[5]*x[0]**3 - 64.0*x[0]**3*x[1] - 256.0*x[0]**3*x[2] - 64.0*x[0]**4*x[3] + 256.0*x[0]**3 +
256.0*x[0]**4 + 32.0*x[4]*x[0]**3*x[1] + 16.0*x[5]*x[0]**2*x[1] - 16.0*x[4]*x[0]**3*x[2] +
16.0*x[4]*x[0]**4*x[1] - 32.0*x[5]*x[0]**2*x[2] + 32.0*x[5]*x[0]**3*x[1] + 16.0*x[4]*x[0]**4*x[2]
+ 16.0*x[5]*x[0]**2*x[3] - 16.0*x[5]*x[0]**3*x[2] + 16.0*x[5]*x[0]**4*x[1] +
16.0*x[5]*x[0]**4*x[2] + 64.0*x[0]**3*x[1]*x[2] - 64.0*x[0]**4*x[1]*x[2] + x[4]*x[5]*x[3] +
x[4]*x[6]*x[3] + x[5]*x[6]*x[3] + 32.0*x[5]*x[0]*x[3] + 32.0*x[6]*x[0]*x[3] - 4.0*x[5]*x[1]*x[3] -
16.0*x[5]*x[2]*x[3] - 4.0*x[6]*x[1]*x[3] - 16.0*x[6]*x[2]*x[3] + x[4]*x[5]*x[0]*x[1] +
x[4]*x[5]*x[0]*x[2] + x[4]*x[6]*x[0]*x[1] + x[4]*x[6]*x[0]*x[2] + x[5]*x[6]*x[0]*x[1] +
x[5]*x[6]*x[0]*x[2] + 2.0*x[4]*x[5]*x[1]*x[3] - 1.0*x[4]*x[5]*x[2]*x[3] + 2.0*x[4]*x[6]*x[1]*x[3] -
1.0*x[4]*x[6]*x[2]*x[3] + 2.0*x[5]*x[6]*x[1]*x[3] - 1.0*x[5]*x[6]*x[2]*x[3] - 4.0*x[5]*x[0]*x[1]*x[2]
- 4.0*x[6]*x[0]*x[1]*x[2] + 4.0*x[5]*x[1]*x[2]*x[3] + 4.0*x[6]*x[1]*x[2]*x[3] -
32.0*x[4]*x[0]**3*x[1]*x[2] - 16.0*x[5]*x[0]**2*x[1]*x[2] + 16.0*x[4]*x[0]**4*x[1]*x[2] +
8.0*x[5]*x[0]**2*x[1]*x[3] - 32.0*x[5]*x[0]**3*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[1]*x[3] -
16.0*x[5]*x[0]**2*x[2]*x[3] + 16.0*x[5]*x[0]**4*x[1]*x[2] - 4.0*x[4]*x[0]**4*x[2]*x[3] -
4.0*x[5]*x[0]**4*x[1]*x[3] - 4.0*x[5]*x[0]**4*x[2]*x[3] + 16.0*x[0]**4*x[1]*x[2]*x[3] +
x[4]*x[5]*x[0]*x[1]*x[2] + 2.0*x[4]*x[5]*x[0]*x[1]*x[3] + x[4]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[5]*x[0]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[3] + x[5]*x[6]*x[0]*x[1]*x[2] +
2.0*x[4]*x[6]*x[0]*x[2]*x[3] + 2.0*x[5]*x[6]*x[0]*x[1]*x[3] + 2.0*x[5]*x[6]*x[0]*x[2]*x[3] -
2.0*x[4]*x[5]*x[1]*x[2]*x[3] - 2.0*x[4]*x[6]*x[1]*x[2]*x[3] - 2.0*x[5]*x[6]*x[1]*x[2]*x[3] -
8.0*x[5]*x[0]*x[1]*x[2]*x[3] - 8.0*x[6]*x[0]*x[1]*x[2]*x[3] - 8.0*x[5]*x[0]**2*x[1]*x[2]*x[3] -
4.0*x[4]*x[0]**4*x[1]*x[2]*x[3] - 4.0*x[5]*x[0]**4*x[1]*x[2]*x[3] +
2.0*x[4]*x[5]*x[0]*x[1]*x[2]*x[3] + 2.0*x[4]*x[6]*x[0]*x[1]*x[2]*x[3] +
2.0*x[5]*x[6]*x[0]*x[1]*x[2]*x[3])))**(1/2)) for x in X])
# Generate samples

param_values_CfB2 = saltelli.sample(problem_CfB2, 1024)

# Run model

Y_CfB2 = evaluate_CfB2(param_values_CfB2)

# Generate the dimension of sample data

print(param_values_CfB2.shape, Y_CfB2.shape)

# Perform analysis

Si_CfB2 = sobol.analyze(problem_CfB2, Y_CfB2, print_to_console=True)

# Print the first-order, second-order, total-order sensitivity indices

print()

# Generate sensitivity plot

Si_df_CfB2 = Si_CfB2.to_df()

barplot(Si_df_CfB2[0])

plot.show()
```

	ST	ST_conf
mu	0.335687	0.059232
r1	0.138494	0.019131
r2	0.235848	0.030296
r3	0.018217	0.004392
k1	0.338859	0.042218
k2	0.041973	0.005953
k3	0.032556	0.005268

	S1	S1_conf
mu	0.216424	0.053693
r1	0.090901	0.035246
r2	0.202391	0.038391
r3	0.005759	0.009937
k1	0.274246	0.049272
k2	0.022256	0.016667
k3	0.017044	0.015706

	S2	S2_conf
(mu, r1)	0.028587	0.069180
(mu, r2)	0.029456	0.069054
(mu, r3)	0.018530	0.063634
(mu, k1)	0.043892	0.074142
(mu, k2)	0.027243	0.064352
(mu, k3)	0.023821	0.067436
(r1, r2)	-0.000160	0.053196
(r1, r3)	-0.003566	0.046952
(r1, k1)	0.012042	0.059063
(r1, k2)	-0.004508	0.048931
(r1, k3)	-0.006012	0.047452
(r2, r3)	0.000009	0.056045
(r2, k1)	0.002556	0.074142
(r2, k2)	-0.002483	0.058720
(r2, k3)	-0.000068	0.058055
(r3, k1)	0.002506	0.015450
(r3, k2)	-0.002428	0.014148
(r3, k3)	-0.000007	0.013832
(k1, k2)	0.007548	0.064320
(k1, k3)	0.001788	0.062609
(k2, k3)	-0.002996	0.024322

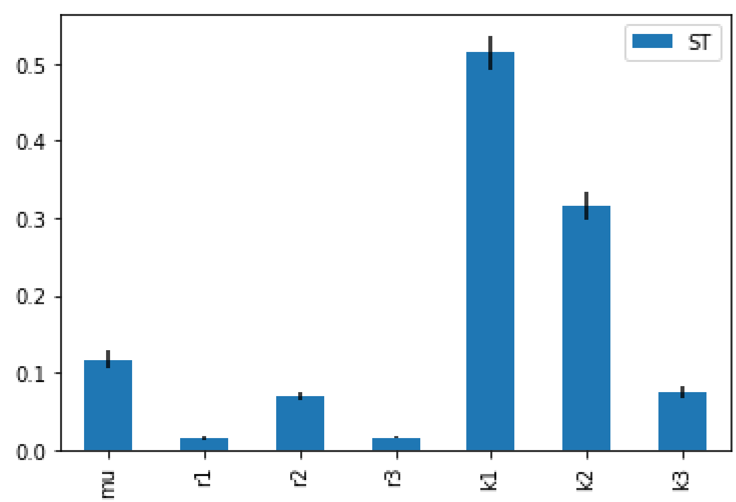


Sensitivity test results of CdB2 (value range of k1, k2, k3 = [0,12])

	ST	ST_conf
mu	0.117179	0.011131
r1	0.014900	0.001703
r2	0.068327	0.005247
r3	0.015959	0.001854
k1	0.513908	0.022587
k2	0.314813	0.017981
k3	0.075063	0.007160

	S1	S1_conf
mu	0.073332	0.014244
r1	0.003199	0.005209
r2	0.043970	0.010467
r3	0.006948	0.006059
k1	0.456158	0.026633
k2	0.259382	0.021287
k3	0.038852	0.012045

	S2	S2_conf
(mu, r1)	-0.000719	0.019121
(mu, r2)	0.005422	0.019783
(mu, r3)	0.000160	0.020195
(mu, k1)	0.008073	0.021655
(mu, k2)	0.008950	0.020210
(mu, k3)	0.011424	0.019839
(r1, r2)	0.000953	0.007269
(r1, r3)	0.001150	0.007382
(r1, k1)	0.008650	0.008340
(r1, k2)	0.002974	0.008069
(r1, k3)	0.001514	0.007574
(r2, r3)	-0.000106	0.016881
(r2, k1)	0.012550	0.017678
(r2, k2)	-0.000162	0.018095
(r2, k3)	-0.001037	0.016755
(r3, k1)	0.000608	0.010064
(r3, k2)	0.001824	0.008687
(r3, k3)	0.002956	0.008608
(k1, k2)	0.025330	0.039751
(k1, k3)	0.001244	0.032525
(k2, k3)	0.016597	0.029541



Sensitivity test results of CfB2 (value range of k1, k2, k3 = [0,12])

	ST	ST_conf
mu	0.302448	0.080207
r1	0.018799	0.005577
r2	0.100462	0.015593
r3	0.005606	0.001580
k1	0.681363	0.077061
k2	0.069598	0.018050
k3	0.135093	0.037278

	S1	S1_conf
mu	0.003855	0.056258
r1	-0.002404	0.011374
r2	0.065846	0.026718
r3	-0.001287	0.006074
k1	0.448930	0.092859
k2	0.037656	0.022258
k3	0.051162	0.034315

	S2	S2_conf
(mu, r1)	0.033767	0.058992
(mu, r2)	0.054141	0.058301
(mu, r3)	0.033462	0.058739
(mu, k1)	0.139230	0.094090
(mu, k2)	0.041082	0.058734
(mu, k3)	0.066040	0.060967
(r1, r2)	0.006679	0.015135
(r1, r3)	-0.000689	0.015951
(r1, k1)	-0.004091	0.033335
(r1, k2)	-0.000992	0.016006
(r1, k3)	0.003549	0.016058
(r2, r3)	0.009973	0.039558
(r2, k1)	0.018147	0.046295
(r2, k2)	0.005918	0.041409
(r2, k3)	-0.005327	0.042710
(r3, k1)	0.003270	0.010042
(r3, k2)	0.001999	0.008616
(r3, k3)	0.005293	0.008774
(k1, k2)	0.029360	0.141807
(k1, k3)	0.042398	0.145324
(k2, k3)	-0.006629	0.037754