RISK BASED ANALYSIS AND DESIGN
OF STIFFENED PLATES

A thesis submitted in partial fulfillment
Of the requirements for the degree of
Master of Science in Engineering

By

HEATHER B. DWIRE
B.S., 2006, Wright State University, Dayton, Ohio.

2008
Wright State University
SCHOOL OF GRADUATE STUDIES

March 14, 2008

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Heather B. Dwire ENTITLED Risk Based Analysis and Design of Stiffened Plates BE ACCEPTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Engineering.

______________________________
Ravi C. Penmetsa, Ph.D.
Thesis Director

______________________________
George P. G. Huang, Ph.D.
Department Chair

Committee on Final Examination

______________________________
Ravi C. Penmetsa, Ph.D.

______________________________
Eric Tuegel, Ph.D.

______________________________
Nathan W. Klingbeil, Ph.D.

______________________________
Joseph F. Thomas, Jr., Ph.D.
Dean, School of Graduate Studies
ABSTRACT


The traditional Risk Based Design (RBD) process involves designing a structure based on risk estimates obtained during several iterations of an optimization routine. This approach is computationally expensive for large-scale aircraft structural systems. The main objective of this research is to establish a RBD algorithm and produce RBD plots for stiffened plates. Basic steps to check functionality will be done by first analyzing a flat plate for which closed formed equations are available and then moving to more complex geometries like stiffened plates.

Therefore, the concept of RBD plots that can be used for both structural sizing and risk assessment are introduced. RBD plots serve as a tool for failure probability assessment given geometry and applied load and can also be used to determine geometric constraints to be used in sizing given allowable failure probability. This approach transforms a reliability-based optimization problem into a deterministic optimization problem with geometric constraints.
# TABLE OF CONTENTS

1. Introduction ........................................................................................................... 1

2. Single Stiffness Inverse Based Internal Load Calculation ............................... 6
   2.1 Load Transfer Matrix ....................................................................................... 7
   2.2 PDF of Internal Load using Convolution ..................................................... 9
   2.3 Problem Description ..................................................................................... 10
   2.4 Summary ....................................................................................................... 17

3. Risk Based Design Plots for Flat Plates .......................................................... 18
   3.1 Probability Distribution Function Modeling .............................................. 18
   3.2 Failure Criteria for Flat Plates .................................................................. 24
   3.3 FFT Based Convolution ............................................................................ 25
   3.4 Risk Based Design Plots ............................................................................ 29
   3.5 Design and Analysis using Risk Based Design Plots ......................... 32
   3.6 Summary .................................................................................................... 39

4. Risk Based Design Plots for Stiffened Plates ............................................... 40
   4.1 Modeling & Analysis of a Stiffened Plate .................................................. 40
   4.2 Failure Criterion for Stiffened Plates ........................................................ 45
   4.3 Risk Based Design Plots ............................................................................ 47
   4.4 Summary .................................................................................................... 48

5. Conclusion .......................................................................................................... 49

Appendices ............................................................................................................... 51
LIST OF FIGURES

Figure 1.1 Typical Structural Test Failures 2
Figure 1.2 Risk-Based Design Plot Development 5
Figure 2.1 Load Transfer Matrix using NASTRAN 8
Figure 2.2 Typical Spar Cross-section 11
Figure 2.3 Finite Element Wing Model with Applied Load Locations 13
Figure 3.1 Sample Spectrum and Exceedence Plot 21
Figure 3.2 Tensile Strength Scatter Data Alclad 2524-T3 22
Figure 3.3 Lot Release Data for an Aluminum Alloy 23
Figure 3.4 Flat Plate in Transverse Loading 25
Figure 3.5 Stress-Strength Plot for a Cracked Plate (Net Section Yielding) 31
Figure 3.6 Stress-Strength Plot for a Cracked Plate (Fracture) 31
Figure 3.7 Risk Based Design Plot for an Aluminum (2024-T3) Flat Plate 32
Figure 3.8 Risk Based Design Plot for an Aluminum (2024-T3) Flat Plate with Various Beta Values 38
Figure 3.9 RBD Plot for an Aluminum (2024-T3) Flat Plate with Beta Values 39
Figure 4.1 Stiffened Plate and Material Lay-Out 42
Figure 4.2 Geometric factor ($\beta(a)$) plot from FRANC 2DL 44
Figure 4.3 Deformed Mesh of Stiffened Plate 45
Figure 4.4 Risk Based Design plot for Stiffened Plate 48
Figure 5.1 Failure Modes for a Stiffened Plate 50
**LIST OF TABLES**

**Table 2.1** Dimensions of the Selected Spar Section  

**Table 2.2** External Loads at Nodal Locations and the Transfer Parameters for $V_Z$  

**Table 2.3** External Loads at Nodal Locations and the Transfer Parameters for $M_X$  

**Table 2.4** Moments and Weights of Three of the External Load PDFs  

**Table 2.5** Moments and Weights of Yield and Shear Strength Distributions  

**Table 3.1** FALSTAFF Spectrum with Normalized Tensile Stress Exceedence Values  

**Table 3.2** Resulting Geometries Satisfying Distance Factor  

**Table 4.1** Material Properties for Each Assigned Area from Figure 4.1
Acknowledgement

This work has been partially supported by AFRL/RBSM and the Midwest Structural Sciences Center (MSSC). MSSC is supported by the U.S. Air Force Research Laboratory Air Vehicles Directorate under contract number FA8650-06-2-3620.
Dedications:

To all who have showed me support and had faith in me:

My parents, my little sister, Dr. Ravi C. Penmetsa, and many others.
1. Introduction

With the advances in computing speed during the recent years, multidisciplinary optimization of large structural systems has become increasingly common in various industries. Nuclear and offshore industries [1,2] have even introduced formal risk based design practices to minimize risk of failure of various components. Literature suggests that the structures designed for minimum risk would not only have lower failure rate but they also result in reduced operating costs over the life of the component. However, the aircraft industry has traditionally relied on factors of safety to design structures. This approach has proved that it is capable of producing safe structures even though its risk was never quantified. These factors work well for designing metallic structures subject to operating conditions that have been well modeled and measured on real aircraft for several decades.

When designing aircraft that will be produced in large numbers, destroying two airframes during the full-scale ground testing is a small portion of the program budget. However, in the future when the fleet size of a particular aircraft platform is small, full-scale testing will amount to a significant portion of the program budget. Also, the coupled thermal-mechanical-acoustic operating conditions that an aircraft will be subjected to, will not be possible to simulate on a full-scale airframe in a laboratory setting. Therefore, even if the existing factors of safety are capable of delivering highly reliable airframes, proof and ultimate testing of these full-scale models to validate the design will not be feasible. With
the elimination of full-scale ground tests, analysis tools become the only available alternative to validate the performance of these structures at the system level.

Past experience, as shown in Figure 1.1, indicates that modern aircraft designed using analytical tools and factors of safety have been experiencing higher number of failures during tests at limit and ultimate load conditions, compared to their counterparts from 1940-1976 time frame. These failures have been attributed to analysis errors, material processing issues, and production anomalies. This also suggests that merely replacing the full-scale tests with analytical predictions, due to lack of capabilities to simulate complex loading conditions, is not sufficient. Therefore, there is a need to incorporate these variations into the design process in order to obtain robust risk estimates of preliminary and detailed design configurations that will reduce/eliminate costly redesign phases and save millions of dollars for the U.S. Air Force.

Figure 1.1: Typical Structural Test Failures [3]
These advanced structural systems also need to be designed using risk integrated design processes which quantify risk of failure of the final structure. There are numerous risk assessment algorithms available in the literature to propagate uncertain input information through the structural system to determine its risk of failure. These techniques have already been used in several multidisciplinary optimization algorithms to size structures for minimum weight and risk. In the past, several researchers have developed risk based design algorithms that use surrogate models of the response to improve the efficiency of the risk assessment process.

While all these past advancements have made risk-based optimization [4-7] practical for large-scale structures they have still relied heavily on risk analyses. These algorithms also needed the user to be familiar with risk assessment methods in order to be able to integrate them into the optimization process. Over the past several years there have been numerous developments that addressed risk assessment of aircraft structures at both the component and system levels. These developments varied from Stochastic Finite Element Methods (SFEM) [8-11] that modeled spatial variation of input parameters to reliability analysis tools that typically handled parametric uncertainty [12-20]. These methods either require considerable computational resources or they use high fidelity models that require an expensive validation process. Moreover, all these methods have focused on modeling material strength and geometric variations either through random fields, parametric variations, or a combination of both. These modeling approaches, random fields, parametric variations and combinations, increased the computational expense for propagating material property variations through the structure.
In this research, an efficient method to analyze aircraft structural components is developed resulting in risk based design plots that can be easily used for structural sizing. The stiffened plate, a common structural configuration, is selected for this research. Since the current factor of safety based structural sizing does not explicitly quantify risk, risk based design plots will provide a means to incorporate risk into design of future aircraft systems.

The main objective of this research is to establish a Risk-Based Design algorithm and produce Risk-Based Design plots for stiffened plates. Since this research involves developing a new algorithm, steps must be taken to check its feasibility. Basic steps to check functionality will be done by first analyzing a flat plate for which closed formed equations are available [21] and then moving to more complex geometries like stiffened plates.

The following chapters will discuss the process of developing these Risk-Based Design plots, following the flow chart in Figure 1.2. An algorithm to efficiently model the Probability Distribution Functions (PDF) of internal loads, when PDF of external loads are available, is presented in Chapter 2. External loads are the aerodynamic loads, inertial loads, engine loads etc. and the internal loads are the reacting shear forces, bending moments, and axial forces at various locations on the structure. These internal load PDFs, along with material property PDFs, will then be used in Chapter 3 to determine failure probability due to yielding and fracture of flat plates. Chapter 3 will also present risk based design plots for flat plates. Chapter 4 will discuss modeling, analysis, and risk based design plots for stiffened plates along with details on the applicability of these plots.
Figure 1.2: Risk-Based Design Plot Development
Chapter 2: Single Stiffness Inverse Based Internal Load Calculation

Typically reliability analysis has been implemented by introducing strength variations through changes in the elastic modulus. This approach makes the stiffness matrix non-deterministic and increases the complexity of the assessment process. Therefore in this research, material property variations are modeled using structural strength to failure as a PDF. Since material testing usually yields strength to failure estimates, this is a meaningful choice which does not introduce randomness into the stiffness matrix. Since the stiffness matrix is deterministic only one finite element analysis of the aircraft structural model with multiple load cases is required to model the PDF of internal load. These multiple load cases can be efficiently analyzed using only the stiffness matrix inverse and without the need for any iterations or function approximations. Absence of iterations and function approximations leads to improved confidence in the predicted failure probability. The need for only one stiffness matrix inverse results in a highly efficient method for large-scale structures. The following sections will present details about the load transfer matrix which will be used to determine the PDFs of internal loads given the PDFs of external loads.
2.1 Load Transfer Matrix

The risk assessment process presented in this research is based on a load transfer approach that is commonly used in the aircraft industry to determine internal load at various cross-sections due to external loads. These internal loads are obtained using a load transfer matrix described below.

The load transfer matrix \((A)\) is an \(m \times n\) matrix where the \(m\) rows correspond to the required internal load locations and the \(n\) columns correspond to the external load locations. For any given load case the finite element model would have \(n\) nodal locations where the external load \((e_j)\) is applied. As shown in Figure 2.1, these \(n\) forces are applied one at a time and the ratio of internal load to the applied load, due to this single force is determined. These values, Equation (2.1), are the elements of the load transfer matrix:

\[
a_{ij} = \frac{\text{Internal load at } i^{\text{th}} \text{ nodal location due to } e_j}{e_j} \quad (2.1)
\]

This process of applying one load at a time to determine the ratio \(a_{ij}\) can be accomplished through the use of load cases in MSC-NASTRAN, which uses the same inverted stiffness matrix to determine all the internal loads with negligible computational effort for large \(n\). Therefore, irrespective of the size of the \(A\) matrix only one stiffness matrix inverse would be required to determine the internal loads.

This \(A\) matrix is then used to propagate external load PDFs through the structure to determine the internal load PDFs. Since tolerances in the structural members do not result in load path variations, for linear static analysis, multiplying various external load vectors with the \(A\) matrix will result in the internal loads. Therefore, a linear
relation exists between the internal load at any specified location and the applied loads which is shown in Equation (2.2):

\[
\text{Internal load at } i^{th} \text{ location} = \sum_{j=1}^{n} a_{ij} e_j
\]  

(2.2)

![Figure 2.1: Load Transfer Matrix using NASTRAN](image)

Since the internal load is a linear combination of external loads, the PDFs of the internal load can be determined using convolution [22] of the PDFs of the external loads. For improved efficiency Fast Fourier Transformation (FFT) can be used to perform the convolution [20].
2.2 PDF of Internal Load using Convolution

When a dependent variable $Z$ is available as the sum of two random variables $X$ and $Y$, PDF of $Z=X+Y$ can be obtained using the convolution integral [22] as shown in Equation (2.3):

$$f_Z(Z) = \int_{-\infty}^{\infty} (f_x(x) f_y(z-x)) dx = f_x(x) \otimes f_y(y)$$  \hspace{1cm} (2.3)

where $\otimes$ represents the convolution between the two PDFs. Using FFT, convolution in the physical domain can be converted into the product of the transformed PDFs of $X$ and $Y$ and the PDF of $Z$ can be obtained as shown below using the inverse Fourier transformation.

$$f_Z(z) = \text{ifft}(\text{fft}(f_x(x)) \cdot \text{fft}(f_y(y)))$$  \hspace{1cm} (2.4)

where ifft(.) is the Inverse Fast Fourier Transform and fft(.) is the Fast Fourier Transform. This approach can be extended to any arbitrarily large problem dealing with a linear combination of random variables.

Since internal loads are modeled as a linear combination of the product of $a_{ij}$ and external load $(e_j)$, the PDF of this product needs to be determined before applying convolution. This PDF can be determined using the chain rule as follows:

$$x_j = a_{ij} e_j$$

$$f_{x_j}(x_j) = \frac{1}{a_{ij}} f_{e_j}\left(\frac{x_j}{a_{ij}}\right)$$  \hspace{1cm} (2.5)

where $f_{x_j}(x_j)$ is the PDF of the variable $x_j$ and $f_{e_j}(\cdot)$ is the PDF of external load.

Therefore, the PDF of the internal load will be obtained using the following expression:
\[ Z_i = \sum_{j=1}^{n} a_{ij} e_j \]

\[ f_{z_i}(z_i) = \text{ifft} \left\{ \prod_{j=1}^{n} \text{fft} \left( \frac{1}{a_{ij}} f_{e_j(x_j)} \right) \right\} \quad (2.6) \]

where \( Z_i \) is the internal load at the \( i \text{th} \) location on the structure. This \( Z_i \) can be used to represent any one of the x, y, or z direction internal forces and moments.

### 2.3 Problem Description

A typical spar cross-section was selected to demonstrate the applicability of the method. The following failure modes were considered to determine the probability of failure of the spar cross-section:

- **Axial and bending**: \( P[f_y - \sigma_{11}(1 + MS_1) \geq 0] \)
- **Torsion and shear**: \( P[f_{su} - \sigma_{12}(1 + MS_2) \geq 0] \) \hspace{1cm} (2.7)

where \( f_y \) is the yield strength, \( f_{su} \) is the ultimate shear strength, \( \sigma_{11} = \frac{M_X Y}{I} \) is the normal stress due to bending and \( \sigma_{12} = \frac{V_Z Q}{I_t} \) is the shear stress. PDFs of the two strength parameters were modeled using the approach presented by Penmetsa, et. al Ref. [23].

\( MS_i \) represents a margin of safety that can be selected to give a desired probability of failure. However, in this paper \( MS_i = 0.0 \) was selected to demonstrate the approach.

When using the FFT-based convolution approach, a future change in \( MS_i \), if additional information becomes available, would result in a shift in the final PDF and requires no further analysis.

Load variations are introduced into \( \sigma_{11} \) and \( \sigma_{12} \) and the strength variations are introduced into \( f_y \) and \( f_{su} \). In this study geometric variations were not considered.
**Spar Cross-Section**

Figure 2.2 shows the representative spar section that is selected to demonstrate the risk assessment process. Dimensions of this section are presented in Table 2.1. The limit moment that is applied on this section is $M_X = -1,373,094$ lbs-in and the ultimate shear force is $V_Z = -56,663$ lbs. The top and bottom skins of this section are made of 7075-T6 Aluminum and the central “C” section is made of Hy-Tuf Forged Steel (220000-240000 psi H.T.).

![Typical Spar Cross-section](image)

**Figure 2.2:** Typical Spar Cross-section
Table 2.1: Dimensions of the Selected Spar Section

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Inches</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>0.184</td>
</tr>
<tr>
<td>D2</td>
<td>0.275</td>
</tr>
<tr>
<td>D3</td>
<td>0.184</td>
</tr>
<tr>
<td>D4</td>
<td>2.788</td>
</tr>
<tr>
<td>D5</td>
<td>0.267</td>
</tr>
<tr>
<td>D6</td>
<td>0.107</td>
</tr>
<tr>
<td>D7</td>
<td>0.279</td>
</tr>
<tr>
<td>D8</td>
<td>0.202</td>
</tr>
<tr>
<td>D9</td>
<td>0.315</td>
</tr>
<tr>
<td>D10</td>
<td>0.202</td>
</tr>
<tr>
<td>D11</td>
<td>2.801</td>
</tr>
<tr>
<td>D12</td>
<td>3.402</td>
</tr>
<tr>
<td>D13</td>
<td>1.770</td>
</tr>
<tr>
<td>D14</td>
<td>1.770</td>
</tr>
<tr>
<td>D15</td>
<td>2.600</td>
</tr>
<tr>
<td>D16</td>
<td>2.600</td>
</tr>
</tbody>
</table>

Using the load transfer matrix algorithm shown in the previous section $M_X$ and $V_Z$, at this location, can be written in terms of external loads ($e_j$) as:

\[
V_Z = \sum_{j=1}^{44} a_{1j} e_j
\]

\[
M_X = \sum_{j=1}^{44} a_{2j} e_j
\]

In Equation (2.8), 44 corresponds to the number of external loads selected to determine the $A$ matrix that would result in the required $V_Z$ and $M_X$ values. These loads and $A$ matrix correspond to a wing configuration, Figure 2.3, available at Wright State University and are scaled for this application. Tables 2.2 and 2.3 have all the values necessary to calculate the internal moment and shear force using external loads. A multimodal PDF modeling technique presented in Ref. [23] is used to assign a PDF to all the 44 external loads.
The FFT-based PDF integration scheme can now be implemented to obtain the PDFs of $M_X$ and $V_Z$. These PDFs will be used to determine the PDF of $\sigma_{11}$ and $\sigma_{12}$. Using the PDFs of the stress and of material strength, failure probability can be obtained by integrating the joint PDF in the intersection region where stress is greater than strength.
The PDF of external load used in this example is a weighted sum of log-normal and normal distributions as shown in Equation (2.9):

\[
\text{Load PDF} = \frac{a \ast (1 - \text{LogNPDF}[M_1,SD_1]) + b \ast (1 - \text{NormalPDF}[M_2,SD_2]) + c \ast (1 - \text{NormalPDF}[M_3,SD_3]) + d \ast (1 - \text{NormalPDF}[M_4,SD_4]) + e \ast (1 - \text{NormalPDF}[M_5,SD_5])}{a + b + c + d + e}
\]  

\[\text{(2.9)}\]

where the first portion of the LoadPDF is a log-normal distribution (LogNPDF) and the other portions of the LoadPDF equation are normal PDF distributions.

<table>
<thead>
<tr>
<th></th>
<th>(a_i)</th>
<th>(e_i)</th>
<th></th>
<th>(a_i)</th>
<th>(e_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0990</td>
<td>4504.40</td>
<td>23</td>
<td>0.9492</td>
<td>5128.64</td>
</tr>
<tr>
<td>2</td>
<td>0.1519</td>
<td>4504.40</td>
<td>24</td>
<td>0.6052</td>
<td>4891.36</td>
</tr>
<tr>
<td>3</td>
<td>0.2400</td>
<td>2467.50</td>
<td>25</td>
<td>0.9770</td>
<td>896.11</td>
</tr>
<tr>
<td>4</td>
<td>0.3088</td>
<td>2467.50</td>
<td>26</td>
<td>1.0491</td>
<td>507.41</td>
</tr>
<tr>
<td>5</td>
<td>0.4404</td>
<td>4934.90</td>
<td>27</td>
<td>1.1220</td>
<td>253.75</td>
</tr>
<tr>
<td>6</td>
<td>0.6419</td>
<td>5699.60</td>
<td>28</td>
<td>1.2101</td>
<td>253.75</td>
</tr>
<tr>
<td>7</td>
<td>0.7608</td>
<td>5699.60</td>
<td>29</td>
<td>0.8354</td>
<td>448.06</td>
</tr>
<tr>
<td>8</td>
<td>-0.0558</td>
<td>4989.86</td>
<td>30</td>
<td>0.9485</td>
<td>448.06</td>
</tr>
<tr>
<td>9</td>
<td>-0.0299</td>
<td>4989.86</td>
<td>31</td>
<td>1.3154</td>
<td>1007.24</td>
</tr>
<tr>
<td>10</td>
<td>-0.0952</td>
<td>2225.75</td>
<td>32</td>
<td>1.3458</td>
<td>1206.84</td>
</tr>
<tr>
<td>11</td>
<td>-0.1032</td>
<td>2225.75</td>
<td>33</td>
<td>1.3787</td>
<td>1206.84</td>
</tr>
<tr>
<td>12</td>
<td>-0.0450</td>
<td>2537.11</td>
<td>34</td>
<td>1.2710</td>
<td>1007.24</td>
</tr>
<tr>
<td>13</td>
<td>-0.0359</td>
<td>2537.11</td>
<td>35</td>
<td>1.6050</td>
<td>568.10</td>
</tr>
<tr>
<td>14</td>
<td>-0.1028</td>
<td>4450.83</td>
<td>36</td>
<td>1.6225</td>
<td>568.10</td>
</tr>
<tr>
<td>15</td>
<td>-0.0169</td>
<td>5074.23</td>
<td>37</td>
<td>1.8857</td>
<td>224.29</td>
</tr>
<tr>
<td>16</td>
<td>0.0296</td>
<td>801.51</td>
<td>38</td>
<td>2.0122</td>
<td>224.29</td>
</tr>
<tr>
<td>17</td>
<td>0.0764</td>
<td>336.81</td>
<td>39</td>
<td>1.2396</td>
<td>1850.43</td>
</tr>
<tr>
<td>18</td>
<td>0.1788</td>
<td>464.70</td>
<td>40</td>
<td>1.2861</td>
<td>1850.43</td>
</tr>
<tr>
<td>19</td>
<td>0.3770</td>
<td>3025.25</td>
<td>41</td>
<td>1.5593</td>
<td>872.07</td>
</tr>
<tr>
<td>20</td>
<td>0.5138</td>
<td>3025.25</td>
<td>42</td>
<td>1.5775</td>
<td>872.07</td>
</tr>
<tr>
<td>21</td>
<td>0.7496</td>
<td>4891.36</td>
<td>43</td>
<td>1.4107</td>
<td>3700.86</td>
</tr>
<tr>
<td>22</td>
<td>0.8488</td>
<td>5128.64</td>
<td>44</td>
<td>1.6766</td>
<td>1744.14</td>
</tr>
</tbody>
</table>

**Table 2.2:** External Loads at Nodal Locations and the Transfer Parameters for \(V_Z\)
(NormalPDF). Values of a few of these weights and the moments of the distributions are shown in Table 2.4. Similarly, the strength PDF is modeled using a weighted sum of two normal distributions whose weights and moments are shown in Table 2.5.

<table>
<thead>
<tr>
<th></th>
<th>$a_{2i}$</th>
<th>$\theta_i$</th>
<th></th>
<th>$a_{3j}$</th>
<th>$\theta_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.3631</td>
<td>4504.40</td>
<td>23</td>
<td>23.642</td>
<td>5128.64</td>
</tr>
<tr>
<td>2</td>
<td>3.4253</td>
<td>4504.40</td>
<td>24</td>
<td>14.829</td>
<td>4891.36</td>
</tr>
<tr>
<td>3</td>
<td>5.21</td>
<td>2467.50</td>
<td>25</td>
<td>21.573</td>
<td>896.11</td>
</tr>
<tr>
<td>4</td>
<td>6.5932</td>
<td>2467.50</td>
<td>26</td>
<td>23.914</td>
<td>507.41</td>
</tr>
<tr>
<td>5</td>
<td>9.1501</td>
<td>4934.90</td>
<td>27</td>
<td>26.416</td>
<td>253.75</td>
</tr>
<tr>
<td>6</td>
<td>13.285</td>
<td>5699.60</td>
<td>28</td>
<td>27.854</td>
<td>253.75</td>
</tr>
<tr>
<td>7</td>
<td>16.044</td>
<td>5699.60</td>
<td>29</td>
<td>17.947</td>
<td>448.06</td>
</tr>
<tr>
<td>8</td>
<td>-0.3524</td>
<td>4989.86</td>
<td>30</td>
<td>19.534</td>
<td>448.06</td>
</tr>
<tr>
<td>9</td>
<td>-0.0877</td>
<td>4989.86</td>
<td>31</td>
<td>26.791</td>
<td>1007.24</td>
</tr>
<tr>
<td>10</td>
<td>-0.0199</td>
<td>2225.75</td>
<td>32</td>
<td>28.694</td>
<td>1206.84</td>
</tr>
<tr>
<td>11</td>
<td>0.2394</td>
<td>2225.75</td>
<td>33</td>
<td>30.735</td>
<td>1206.84</td>
</tr>
<tr>
<td>12</td>
<td>0.4853</td>
<td>2537.11</td>
<td>34</td>
<td>24.223</td>
<td>1007.24</td>
</tr>
<tr>
<td>13</td>
<td>0.9274</td>
<td>2537.11</td>
<td>35</td>
<td>33.543</td>
<td>568.10</td>
</tr>
<tr>
<td>14</td>
<td>0.6531</td>
<td>4450.83</td>
<td>36</td>
<td>35.169</td>
<td>568.10</td>
</tr>
<tr>
<td>15</td>
<td>1.5557</td>
<td>5074.23</td>
<td>37</td>
<td>37.086</td>
<td>224.29</td>
</tr>
<tr>
<td>16</td>
<td>2.8217</td>
<td>801.51</td>
<td>38</td>
<td>36.346</td>
<td>224.29</td>
</tr>
<tr>
<td>17</td>
<td>3.9123</td>
<td>336.81</td>
<td>39</td>
<td>30.682</td>
<td>1850.43</td>
</tr>
<tr>
<td>18</td>
<td>6.0612</td>
<td>464.70</td>
<td>40</td>
<td>32.454</td>
<td>1850.43</td>
</tr>
<tr>
<td>19</td>
<td>10.003</td>
<td>3025.25</td>
<td>41</td>
<td>36.351</td>
<td>872.07</td>
</tr>
<tr>
<td>20</td>
<td>12.831</td>
<td>3025.25</td>
<td>42</td>
<td>37.642</td>
<td>872.07</td>
</tr>
<tr>
<td>21</td>
<td>18.256</td>
<td>4891.36</td>
<td>43</td>
<td>36.203</td>
<td>3700.86</td>
</tr>
<tr>
<td>22</td>
<td>20.85</td>
<td>5128.64</td>
<td>44</td>
<td>40.956</td>
<td>1744.14</td>
</tr>
</tbody>
</table>

**Table 2.3:** External Loads at Nodal Locations and the Transfer Parameters for $M_X$
Table 2.4: Moments and Weights of Three of the External Load PDFs

<table>
<thead>
<tr>
<th></th>
<th>Moment 1 (lb-ft)</th>
<th>Moment 2 (lb-ft)</th>
<th>Moment 3 (lb-ft)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>2345.50</td>
<td>1284.90</td>
<td>2569.70</td>
</tr>
<tr>
<td>SD1</td>
<td>477.90</td>
<td>261.79</td>
<td>523.58</td>
</tr>
<tr>
<td>M2</td>
<td>1183.40</td>
<td>648.25</td>
<td>1296.50</td>
</tr>
<tr>
<td>SD2</td>
<td>258.26</td>
<td>141.47</td>
<td>282.94</td>
</tr>
<tr>
<td>M3</td>
<td>2385.50</td>
<td>1295.80</td>
<td>2591.50</td>
</tr>
<tr>
<td>SD3</td>
<td>346.610</td>
<td>189.870</td>
<td>379.740</td>
</tr>
<tr>
<td>M4</td>
<td>1164.90</td>
<td>638.14</td>
<td>1276.30</td>
</tr>
<tr>
<td>SD4</td>
<td>313.800</td>
<td>171.900</td>
<td>343.790</td>
</tr>
<tr>
<td>M5</td>
<td>1193.800</td>
<td>653.940</td>
<td>1307.900</td>
</tr>
<tr>
<td>SD5</td>
<td>353.390</td>
<td>193.580</td>
<td>387.160</td>
</tr>
</tbody>
</table>

Table 2.5: Moments and Weights of Yield and Shear Strength Distributions

<table>
<thead>
<tr>
<th></th>
<th>Yield Strength (lb/in²)</th>
<th>Shear Strength (lb/in²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>75202.00</td>
<td>145980.00</td>
</tr>
<tr>
<td>SD1</td>
<td>2145.50</td>
<td>4164.80</td>
</tr>
<tr>
<td>M2</td>
<td>71228.00</td>
<td>138270.00</td>
</tr>
<tr>
<td>SD2</td>
<td>2771.80</td>
<td>5380.50</td>
</tr>
<tr>
<td>a</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>b</td>
<td>0.16</td>
<td>0.16</td>
</tr>
</tbody>
</table>

Based on Equation (2.7), for $M_S = 0.0$, the probability of failure due to bending was determined to be $1.006 \times 10^{-8}$, and the probability of failure due to ultimate shear was determined to be $2.024 \times 10^{-6}$. These values correspond to a case where the external loads are uncorrelated. When these correlations are available a simple extension of the FFT algorithm would enable risk estimation in the presence of correlated loads.
2.4 Summary

This chapter presents details of a new internal load PDF modeling technique that requires only one stiffness matrix inversion when dealing with static analysis. This was made possible through a new modeling technique that eliminated the need for propagating strength variations through the finite element model. Also this method does not use any surrogate models (response surface, kriging, etc.), which typically tend to introduce errors into the assessment process. Therefore, the presented method can predict the failure probability of large-scale aircraft structures with a high level of confidence compared to other methods available in the literature.
Chapter 3: Risk Based Design Plots for Flat Plates

This chapter discusses the process of obtaining the Risk Based Design (RBD) plots for flat plates. RBD plots directly provide the risk of failure estimates for a component using simple scaling factors that are similar to margin of safety calculations that are commonly used by designers. These design plots are developed using normalized PDFs of load and material properties. A flat plate is selected as an example to demonstrate the development and use of risk based design plots. Details on PDF modeling and normalization, scaling factor, failure criteria and the resulting RBD plots, will be presented in the following sections.

3.1 Probability Distribution Function Modeling

Applied load or stress \((L \text{ or } \sigma)\), yield strength \((\sigma_y)\), and fracture toughness \((k_c)\) are modeled as random variables whose distribution functions are determined using load spectrum [24], MIL-HDBK-5J information [25], and test data respectively. These PDFs are all normalized such that they can be scaled to an arbitrary load, material strength, and toughness values through a simple PDF transformation. Currently, the Load PDF is modeled using the exceedence data available for the FALSTAFF spectrum. Table 3.1 shows the exceedence data from the FALSTAFF spectrum where 1.0 corresponds to the maximum load in the spectrum.
“The FALSTAFF Spectrum is a standard load sequence considered representative of the load-time history in the lower wing skin near the wing root of a fighter aircraft.” [24] Since this research deals with plates in transverse loading, only the tensile exceedence portion of the FALSTAFF spectrum is considered.

<table>
<thead>
<tr>
<th>Normalized Stress</th>
<th>Exceedences</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.02</td>
<td>18000</td>
</tr>
<tr>
<td>0.17</td>
<td>17500</td>
</tr>
<tr>
<td>0.21</td>
<td>17000</td>
</tr>
<tr>
<td>0.25</td>
<td>13000</td>
</tr>
<tr>
<td>0.29</td>
<td>8500</td>
</tr>
<tr>
<td>0.33</td>
<td>6600</td>
</tr>
<tr>
<td>0.37</td>
<td>5000</td>
</tr>
<tr>
<td>0.41</td>
<td>4000</td>
</tr>
<tr>
<td>0.46</td>
<td>3300</td>
</tr>
<tr>
<td>0.50</td>
<td>2200</td>
</tr>
<tr>
<td>0.55</td>
<td>1600</td>
</tr>
<tr>
<td>0.59</td>
<td>1100</td>
</tr>
<tr>
<td>0.63</td>
<td>700</td>
</tr>
<tr>
<td>0.67</td>
<td>450</td>
</tr>
<tr>
<td>0.70</td>
<td>260</td>
</tr>
<tr>
<td>0.75</td>
<td>160</td>
</tr>
<tr>
<td>0.79</td>
<td>90</td>
</tr>
<tr>
<td>0.82</td>
<td>40</td>
</tr>
<tr>
<td>0.86</td>
<td>20</td>
</tr>
<tr>
<td>0.90</td>
<td>9</td>
</tr>
<tr>
<td>1.00</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 3.1: FALSTAFF Spectrum with Normalized Tensile Stress Exceedence Values

Penmetsa et. al. have shown in Ref [23] that none of the standard statistical distributions are capable of completely representing the probability distribution information of the FALSTAFF spectrum. Moreover, the traditional distribution fitting algorithms and tests are suitable to fit a distribution in the nominal regions of the PDF and fail to capture the required accuracy in the tails. This is because low values of the cumulative distribution function (CDF) in the tail regions result in a numerically small difference between the actual and fitted CDF, suggesting the traditional test fitted distribution is accurate. These
tail regions are critical for the accurately predicting risk or failure probability. Therefore, a new test statistic was adopted in Ref. [23] to model these distribution functions. Instead of using traditional distributions that have either two or three parameters that define the PDF, this research uses a weighted sum of distributions to model the PDFs. The following equation represents how the distributions are combined.

\[
CDF = \frac{a \cdot CDF_1 + b \cdot CDF_2 + c \cdot CDF_3 + d \cdot CDF_4}{a + b + c + d}
\] (3.1)

An optimization algorithm was implemented that begins with one distribution and adds additional distributions as need be in order to minimize the error between the fitted and exact distributions, which is determined based on the test statistic presented in Ref [23]. This test statistic used Log$_{10}$(CDF) of both the actual and fitted distributions to accommodate the tail regions with sufficient accuracy. The parameter selected to minimize is the sum of the absolute difference at all the comparison points for both the distributions. These comparison points are the discrete locations selected from the input CDF (exceedence) data. For example, 21 comparison points were used for FALSTAFF spectrum (Table 3.1). Figure 3.1 shows a sample FALSTAFF load spectrum and the resulting exceedence plot of the FALSTAFF spectrum.
This weighted distribution fitting scheme was used to determine load, material strength, and fracture strength PDFs. Just as the load PDF was modeled using FALSTAFF spectrum, the yield strength PDF was modeled using tensile strength scatter information of Alclad 2524-T3, available in MIL-HDBK-5J [25], Figure 3.2. The PDF of fracture toughness was, however, determined using lot release data for an aluminum alloy (actual specification was unavailable), Figure 3.3. The lot release data provided the $\frac{da}{dN}$ vs. $\Delta K$ for 74 test specimens. Scatter of $\Delta K$ at $\frac{da}{dN} = 10^{-2}$ was used to model the PDF of fracture toughness. Information from these three sources was used to model load, strength, and fracture toughness distributions, which are then normalized to represent generic scatter information.
Figure 3.2: Tensile Strength Scatter Data Al clad 2524-T3

Different normalization factors were used for each of these distributions. The load distribution was normalized such that 1.0 on the abscissa represented the Limit Load ($LL$) condition where $Pr[\text{Load} > LL] = 10^{-7}$. For the normalized strength distribution, 1.0 corresponded to either A or B basis strength allowables. A-basis values for material strength are those values which will be exceeded 99% of the time with a 95% confidence interval and are commonly used in the design of transport aircraft. B-basis values are exceeded 90% of the time with a 95% confidence interval; these values are used in the design of tactical aircraft and therefore will be used in this research. Since mean values for fracture toughness are typically available in the Mil-Handbooks [25], the PDF was normalized such that 1.0 corresponds to mean $K_C$. 
These normalized PDFs can now be scaled using PDF transformation to any arbitrary limit load value, yield strength allowable, or mean $K_C$. The normalized PDFs of load, yield strength, and fracture toughness are shown below.

\[
PDF_\sigma = \frac{0.067569 \text{LogNormPDF}(0.46514, 0.076857) + 0.29604 \text{NormPDF}(0.2849, 0.05942) + 0.097061 \text{NormPDF}(0.18372, 0.044771) + 0.21045 \text{NormPDF}(0.16812, 0.054012) + 0.11394 \text{NormPDF}(0.17471, 0.028033)}{0.067569 + 0.29604 + 0.097061 + 0.21045 + 0.11394}
\]  

(3.2)

**Figure 3.3:** Lot Release Data for an Aluminum Alloy
3.2 Failure Criteria for the Design and Analysis of a Flat Plate

Two failure criteria were considered in this research for design and analysis of a flat plate under uniaxial tension. Failure was defined as stress exceeding the residual strength of the structure. Residual strength is the minimum of (1) the net section yield and (2) the stress to cause fracture. The following equations show the residual strength for net section yielding of a cracked plate, and for fracture of a cracked plate [21]. When $a$ is equal to zero, Equation 3.5 results in the yield criteria for an undamaged plate. Yield strength is represented by $\sigma_y$, $K_C$ is the fracture toughness, and $\alpha$ is the ratio of crack length to plate width. The variable definitions are shown in Figure 3.4.

$$R.S_{\text{damaged}} = \sigma_y I(w - 2a)$$  \hspace{1cm} (3.5)

$$R.S_{\kappa} = K_C I \left( \frac{2w}{\pi \alpha} \cos \left( \frac{\pi \alpha}{2} \right) \right)$$  \hspace{1cm} (3.6)

Based on these definitions of residual strength, probability of failure for this flat plate subject to random load can be assessed using the following equations.

Net-section yielding of a cracked plate: $P_f = \mathbb{P} \left[ \sigma - \left( \frac{2a}{w} \right) \sigma_y > 0 \right]$  \hspace{1cm} (3.7)
Fracture of a cracked plate: 

\[ P_f = P \left[ LL - K_{ct} \sqrt{\frac{2w}{\pi \alpha}} \cos \left( \frac{\pi \alpha}{2} \right) > 0 \right] \]  

(3.8)

where \( \alpha = \frac{2a}{w} \)

**Figure 3.4:** Flat Plate in Transverse Loading

The above-equations can be implemented using Fast Fourier Transform (FFT) based numerical convolution technique to determine the failure probability due to these individual failure modes. All three equations are in the form of \( X - m Y \) where \( X \) and \( Y \) are random variables and \( m \) is a deterministic parameter that depends on the geometry. The following section provides details on estimating the probability of failure when the failure criterion is of the form \( X - m Y \).

### 3.3 FFT-Based Convolution

Fast Fourier Transform (FFT) based convolution is a well-developed technique and has been used for many decades in signal processing [27]. FFT algorithm derives its
efficiency from its transformation into frequency domain, which converts an expensive convolution in the physical domain to an inexpensive product of two signals (PDF in this case) in the frequency domain. The following are the steps involved in an FFT based convolution algorithm for a problem of the form “$X - mY$”.

**Step 1:** FFT-based convolution is applicable for a linear combination of random variables. Therefore, the equation needs to be transformed into “$X+Z$”. This can be achieved by introducing a transformation variable $Z = -mY$. The PDF of $Z$ can be determined using chain rule as follows

$$f_Z = \frac{dy}{dz} f_Y = \left| \frac{1}{-m} \right| f_Y \left( \frac{-1}{m} Z \right)$$

(3.9)

**Step 2:** Once the failure condition is available as a linear combination of two random variables $X$ and $Z$, their PDFs need to be discretized using a common discretization factor. This discretization enables implementation of an efficient Discrete FFT algorithm instead of continuous transformation techniques that are inapplicable for this situation. The discretization factor for the PDFs is determined based on the number of data points and the bounds of the variables used for the convolution. Most FFT algorithms are optimized to handle points as powers of 2. Therefore, $2^{10}$ to $2^{12}$ points are used in this research. A convergence study can be performed and the number of data points can be determined based on the required accuracy level. Once the number of data points are selected the smallest range (max-min of $X$ or $Z$) is selected to determine the discretization factor. The smallest is considered such that both the distributions will be represented with at least the selected number of data points.
This discretization step yields two vectors of different sizes depending on the range of $X$ and $Z$. For example, consider a case where $X$ is a random variable with range $[0, 4]$ representing the entire area under its PDF, and $Z$ has a range $[3, 9]$. If the required number of data points was selected as $2^{10}$, then the discretization factor would be $4/2^{10}$, and the size of discretized $X$ vector will be $2^{10}$. When the PDF of $Z$ is discretized using the same factor, its size is determined by calculating the absolute difference in the range between $X$ and $Z$ ($6-4=2$), dividing by the discretization factor, and then adding the value to the current length of the $X$ vector. This results in a length of $2^{10}+(2/4)*2^{10}$ for the $Z$ vector. However, in order to apply discrete FFT algorithm the sizes of the vectors need to be equal. Therefore, $X$ vector is padded with zeros to make its size equal to $2^{10}+(2/4)*2^{10}$. This padding would mean that the new $X$ vector represents a range $[0, 6]$ instead of $[0, 4]$. With this padding both the vectors are of equal size but they are not a power of 2. So a final padding is required that would accomplish two tasks: add additional zeros until the size is equal to a power of 2, and then double the size of vectors. The first criterion of making the size a power of 2 would ensure better efficiency because the discrete FFT algorithms are optimized to deal with powers of 2. The second criterion of doubling the vector would eliminate the circular convolution issue of discrete FFT.

The first criterion of making the size equal to a power of 2 would alter the range of the vectors and this new range needs to be determined. For the above example, the vectors $X[0, 6]$ and $Z[3, 9]$ are of size $2^{10}+2^{9}$, since the next power of 2 is $2^{11}$ both the vectors are padded with $2^{9}$ zeros. Based on this, the new range of the two vectors will be $X [0, 8]$, and $Z [3, 11]$. This range can be determined using the discretization factor.
\[
\begin{align*}
DiscretizationFactor &= \frac{4}{2^{10}} = 0.0039 \\
NewRangeExtension &= \frac{4}{2^{10}} 2^9 = 2
\end{align*}
\] (3.10)

**Step 3:** Once the discrete vectors are available the next step is to obtain the FFT of each of these vectors. In this research, MATLAB FFT routines were used but any generic FFT algorithm would yield the same results.

\[
A = \text{fft}(X) \\
B = \text{fft}(Z)
\] (3.11)

**Step 4:** FFT of a PDF vector in step 3 converts the convolution integral of two PDFs into the product of two vectors \(A\) and \(B\). Therefore, both these vectors are multiplied one element to element using complex number multiplication.

For the entire length of \(A\)

\[
C(i) = A(i) \cdot B(i)
\] (3.12)

**Step 5:** Inverse FFT of the vector \(C\) will finally result in the PDF of the failure condition represented by \(g = X - mY\).

\[
f_g = \text{ifft}(C)
\] (3.13)

**Step 6:** The range of the final PDF of \(g\) is determined using the ranges of \(X\) and \(Z\). Since the failure condition is a linear sum of two variables range of \(g\) will be the interval addition of the two ranges. Based on the values selected earlier, \(X[0, 8]\) and \(Z[3, 11]\) which would results in range of \(g[0+3, 8+11] = [3, 19]\). Therefore, the final PDF would represent this range of values. It should be noted that the size of the final \(g\) vector would be equal to the size of the input vectors after doubling their length.

**Step 7:** During this step, the area under the failure function is integrated based on the failure limit using a numerical integration scheme. For example, if \(g<5\) represents failure,
the first 512 points (IntegrationRange/DiscretizationFactor) from the PDF of \( g \) will be used to determine the failure probability.

The FFT-based integration process is highly efficient once implemented and can be applied to problems with non-traditional PDF definitions without the need for any approximations of the PDF. Therefore, in this research this is an ideal choice due to the non-traditional load, strength, and fracture toughness PDF models. Even though the failure criteria are available as closed form equations, Monte Carlo simulations cannot be applied in this situation because the low probabilities of failure values, on the order of \( 10^{-8} \), make Monte Carlo [28] integration inefficient.

Using this PDF convolution scheme, risk based design plots have been generated for an aluminum flat plate. Details about these design plots are presented in the following section.

3.4 Risk Based Design Plots

In this section, a novel concept of RBD plots will be introduced. These plots are like probabilistic characteristic charts that can be used to determine failure probability directly from the plots without the need for reliability analysis. These plots are developed using the concept of distance factor (\( DF \)), which represents distance between the two PDFs used in the failure equation. For an intact structure, this distance factor would represent the distance between the limit load and the strength allowable, which is same as the margin of safety. However, for plate with a crack this distance factor for net section yielding criterion represents the distance between the limit load and the load to cause net
section yielding. For fracture, the distance factor represents the distance between limit load and load to cause fracture. The yield strength is represented by $F_{ty}$, and the limit load is represented by $LL$. The two distance factors used in this research are shown in Equations 3.14, 3.15a, and 3.15b. When $a$ is zero, the distance factor from Equation 3.14 results in the distance factor or margin of safety for the intact structure.

Distance factor for net section yield: 

$$DF_{NS} = \frac{(1-\frac{2a}{w})(wt)F_{ty} - LL}{LL}$$

(3.14)

Distance factor for fracture: 

$$DF_{KC} = \frac{BK_c - LL}{LL}$$

(3.15a)

Where $B = t \sqrt{\frac{2w}{\pi\alpha} \cos\left(\frac{\pi\alpha}{2}\right)}$ and $\alpha = \frac{2a}{w}$

(3.15b)

The Figures 3.5, and 3.6, show the concept of distance factor used in this research. Using these definitions for the distance factors, a relation between distance factor and probability of failure for these failure criteria can be determined using FFT-based integration. This integration can be performed using the normalized PDFs of load, yield strength, and fracture toughness. Figure 3.7 shows the RBD plot generated for a flat plate made of aluminum 2024-T3 and subjected to the FALSTAFF spectrum.
Figure 3.5: Stress-Strength Plot for a Cracked Plate (Net Section Yielding)

Figure 3.6: Stress-Strength Plot for a Cracked Plate (Fracture)
3.5 Design and Analysis using RBD Plots

RBD plots developed in this research serve two purposes: one is to generate a geometric constraint for the flat plate given an allowable failure probability, and the other is to determine failure probability given geometry and loading conditions for a flat plate. Once these design plots are generated there would be no need for additional reliability analysis for sizing the structure. This approach converts a reliability based design optimization task that is used for sizing into a deterministic optimization task with geometric constraints that are predetermined based on risk allowables.
Numerical Example:

This example of a flat plate, Figure 3.4, demonstrates the concept of analysis and design using RBD plots. A flat plate made of Aluminum 2024-T3 with \( w = 30 \text{ in}, \ t = 0.2 \text{ in}, \ a = 5 \text{ in}, \) and margin of safety of zero is selected as a demonstration example. In this example, the allowable yield strength \( F_{\text{ty}} = 48 \text{ ksi} \) and mean fracture toughness \( K_C = 100 \text{ ksi} \cdot \sqrt{\text{in}} \) [31] were selected. Since we already have a structure and would like to determine the failure probability given that the plate is cracked, the first step would be to determine the distance factors for net section yield and fracture. Using this distance factor the failure probability can be directly obtained from RBD plot in Figure 3.7.

*Failure Probability Calculations:*

*Net Section Yielding*

\[
DF_{\text{NS}} = \frac{\left(1 - \frac{2a}{w}\right) F_{\text{ty}} - LL}{LL} = \frac{\left(1 - \frac{2(5)}{30}\right) 48 - 48}{48} = \frac{2}{3} - \frac{48}{48} = -0.3333
\]

*Fracture*

\[
\alpha = \frac{2a}{w} = \frac{2(5)}{30} = \frac{1}{3}
\]

\[
B = t \sqrt{\frac{2w}{\pi \alpha} \cos \left(\frac{\pi \alpha}{2}\right)} = 0.2 \sqrt{\frac{2(30)}{\pi \left(\frac{1}{3}\right)} \cos \left(\frac{\pi (1/3)}{2}\right)} = 0.2 \sqrt{28.6479 \cos(0.5236)} = 0.9962
\]

\[
DF_{\text{Kc}} = \frac{BK_C - LL}{LL} = \frac{(0.9962)100 - 48}{48} = 1.075
\]
Interpolations of Probability of failures resulted in the following:

Probability of failure due to net section yielding = 6.1039955 E-4

Probability of failure due to fracture = negligibly small number

The results from this example show that the net section yield criteria govern the probability of failure. The structure defined will fail due to net section yielding.

In another example the stress applied to the structure is given as 45ksi, and the plate is made of aluminum 2024-T3. The designer would now need to size the structure such that the failure probability due to net section yield and fracture is less than 1E-7. In this case, an appropriate distance factor will be selected based on the failure probability constraints from the RBD plot and this factor would represent a constraint on the geometry of the structure. Using this information, the structure can be sized for minimum weight while satisfying these geometric constraints.

*Distance Factor Calculation: use of interpolation and RBD plot values*

\[
DF_{NS} = \left( \frac{1(10^{-7}) - 9.124(10^{-8})}{1.204(10^{-7}) - 9.124(10^{-8})} \right) (-0.02 + 0.01) + -0.01 = -0.01300
\]

\[
DF_{Kc} = \left( \frac{1(10^{-7}) - 8.064(10^{-8})}{1.071(10^{-7}) - 8.064(10^{-8})} \right) (-0.01 - 0) + 0 = -0.007317
\]

Required distance factor for net section yielding = -0.013

Required distance factor for fracture = -0.0073

A combination of geometries can be found to satisfy the distance factor Equations 3.14 and 3.15. Values for \(w\) were chosen to calculate the resulting \(a\) values, maximum allowable crack length (2\(a\)) before probability of failure is exceeded, in net section yield. The values from net section yield for \(a\) and \(w\) were then used to calculate values for \(t\) for
fracture criteria so all probability of failures pertaining to net section yield and fracture were satisfied with the same plate geometries. The resulting geometric combinations for three different cases can be seen in Table 3.2.

**Geometry Calculations:**

For net section yielding, the geometric relations are:

\[
\left(1 - \frac{2a}{w}\right) = \frac{DF_{S_y}LL + LL}{F_y} = \frac{-0.013(45) + 45}{48} = 0.9253
\]

Choosing \( w \) as 24in, the outcome of \( a \) is:

\[
a = \frac{(1 - 0.9253)w}{2} = \frac{(1 - 0.9253)24}{2} = 0.8964
\]

Resulting in a maximum crack length of \( 2a = 1.7928 \)in.

The values for \( a \) and \( w \) are then applied to the equations for fracture:

\[
B = t \sqrt{\frac{2w}{\pi \alpha} \cos\left(\frac{\pi \alpha}{2}\right)}
\]

\( B \) is also related to the \( DF_{Kc} \) by

\[
B = \frac{DF_{Kc}LL + LL}{K_c} = \frac{-0.0073(45) + 45}{100} = 0.4467
\]

Solving for \( t \):

\[
\alpha = \frac{2a}{w} = \frac{2(0.8964)}{24} = 0.0747
\]

\[
t = \frac{B}{\sqrt{\frac{2w}{\pi \alpha} \cos\left(\frac{\pi \alpha}{2}\right)} = \frac{0.4467}{\sqrt{\frac{2(24)}{\pi(0.0747)} \cos\left(\frac{\pi(0.0747)}{2}\right)}} = 0.0312
\]

The final dimensions for the plate calculated above are \( w = 24 \)in, allowable \( a = 0.8964 \)in, and \( t = 0.0312 \)in.
Since the distance factor is not a common term that designers are familiar with, a few modifications are introduced into the design plots. The new plots are generated based on the ratio of crack length to plate width, \( \alpha = \frac{2a}{w} \). This parameter is widely used in the aerospace industry. Therefore, this will enable easy transition of Risk-Based Design technology into the industry.

While \( \alpha \) represents the crack length in the plate, the geometric and strength parameters are combined into a new term \( \beta \). The form of this new term depends on whether we are dealing with net-section yielding or fracture and they are given in Equations 3.19 and 3.20.

Net section yielding:

\[
\alpha = 1 - \frac{DF_{\text{NetSection}} + 1}{\beta} \quad (3.19a)
\]

where \( \alpha = \frac{2a}{w} \) and \( \beta = \frac{F_{\text{wt}}}{LL} \) \hspace{1cm} (3.19b)

Fracture:

\[
f(\alpha) = \left( \frac{2}{\pi \alpha} \cos \left[ \frac{\pi \alpha}{2} \right] \right) = \left( \frac{DF + 1}{\beta} \right)^2 \quad (3.20a)
\]

### Table 3.2: Resulting Geometries Satisfying Distance Factor

<table>
<thead>
<tr>
<th></th>
<th>( Pf )</th>
<th>DF</th>
<th>Geometric Constants</th>
<th>a</th>
<th>w</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net Section</td>
<td>1.00E-07</td>
<td>0.0013</td>
<td>0.9253</td>
<td>0.9983</td>
<td>24</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>1.00E-07</td>
<td>0.0013</td>
<td>0.9253</td>
<td>1.3444</td>
<td>36</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>1.00E-07</td>
<td>0.0013</td>
<td>0.9253</td>
<td>1.7925</td>
<td>48</td>
<td>na</td>
</tr>
<tr>
<td>Fracture</td>
<td>1.00E-07</td>
<td>-0.0073</td>
<td>0.4457</td>
<td>0.8983</td>
<td>24</td>
<td>0.0313</td>
</tr>
<tr>
<td></td>
<td>1.00E-07</td>
<td>-0.0073</td>
<td>0.4457</td>
<td>1.3444</td>
<td>36</td>
<td>0.0256</td>
</tr>
<tr>
<td></td>
<td>1.00E-07</td>
<td>-0.0073</td>
<td>0.4457</td>
<td>1.7925</td>
<td>48</td>
<td>0.0222</td>
</tr>
</tbody>
</table>

Geometric Constant for Net Section is \( (1-2a/w) \)

Geometric Constant for Fracture is \( B = t' \sqrt{\left[ \frac{2}{w} \pi \alpha \cos(\pi \alpha/2) \right]} \)
where $\beta = \frac{K_c t \sqrt{w}}{LL}$ \hfill (3.20b)

Figure 3.8 shows the variation of probability of failure with crack size for different designs represented by $\beta$. Therefore, for risk analysis of a given geometry, loading, and material properties, these $\beta$ values can directly give a designer information about the failure probability and its sensitivity to crack length. Based on this chart, a designer can also determine a required $\beta$ that can minimize probability of failure. These plots can be used for analysis and design giving a lot more information than any of the existing risk assessment methods that predict only one value for probability of failure. As can be seen from Figure 3.9, these plots can also identify conditions when one particular failure mode is more probable compared to the others.
Figure 3.8: Risk Based Design plot for an Aluminum (2024-T3) Flat Plate with Various Beta Values

Figure 3.9 demonstrates the importance of each failure criteria. The plot indicates that it is more likely to experience net section yielding until the ratio of crack length to plate width reaches around 0.11, after which it is more likely to fail due to fracture. These plots can be generated for various configurations and can be used for design and analysis of flat plates.
3.6 Summary

By creating PDFs from the input criteria for loading, material properties, and strength, and using these PDFs along with FFT, the probabilities of failure for a cracked flat plate with fracture and net section yielding criteria can be determined. RBD plots can be used to determine geometric design constraints based on distance factors or crack length given a target probability of failure. These resulting plots allow for analysis of existing flat plates or can be used to design flat plates, incorporating the risk of failure into the design before construction. The RBD plots for flat plates have created a basic path to follow in establishing such plots for stiffened plates and other structural members.
Chapter 4: Risk Based Design Plots for Stiffened Plates

While RBD plots for flat plates have been developed in Chapter 3, a similar process can be applied to stiffened plates. Stiffened plates have numerous applications in both aerospace and naval structures. Typical wing and fuselage sections of fighter or transport aircraft have stiffened plates that need to be designed based on risk of fracture criteria. Similarly, ship hull structures also have stiffened plates that are sized based on fracture criteria. Therefore, the developments presented in this chapter have a wide range of applications and they represent a new design methodology that seamlessly integrates risk assessment into existing technology.

Development of RBD plots for stiffened plates will require additional effort since there is no closed-form equation available for fracture criteria. The following Chapter will present details of modeling and analysis of a stiffened plate in FRANC-2DL to obtain the stress intensity factors for various crack lengths. These stress intensity factors will later be used to determine probability of fracture and develop RBD plots.

4.1 Modeling & Analysis of a Stiffened Plate

For this research, software from Cornell University [29] is used to model and analyze a stiffened plate. FRANC-2DL analyzes a three-dimensional model as various layers of a two dimensional model. Therefore, one layer of flat plate with a specified thickness and another layer of stiffeners with their thickness will represent a stiffened plate. These
layers are then adhered based on the conditions that are being simulated. For a riveted plate only regions that need to represent rivets are selected to adhere the two layers. In this research, an integral stiffener is modeled by selecting the entire surface of the stiffeners as the region to adhere the two layers. The pre-processor for FRANC-2DL that is used in this research is CASCA, which generates the finite element model for both the layers. These layers are then combined using a translation package called CAStoFRANC. Stiffeners were modeled as frames attached at the ends as shown in Figure 4.1 due to certain modeling restrictions of CASCA and FRANC-2DL. CASCA does not permit disconnected regions for constructing the finite element mesh. This requires creating individual stiffeners as separate layers. When these layers are combined and integrated into one model, FRANC-2DL stacks these layers on top of each other as opposed to placing all the layers on top of the flat plate. Therefore, to circumvent this modeling difficulty and still be able to use the advanced automatic meshing and analysis features of FRANC-2DL the stiffeners are connected at the ends in CASCA. The effects of these connected portions of stiffeners, labeled A in Figure 4.1, are minimized by using material properties in the connector regions that simulate near zero stiffness. These material properties are shown in Table 4.1. Moreover, these additional regions do not have load applied in the analysis because the strong shear modulus values of the adhesive introduced some errors into the calculations.
Dimensions for the stiffened plate modeled in this research are as follows (see Figure 4.1 for plate layout). Length and width of the base plate are 32 x 32 x 0.5 inches. The stiffener spacing ($b$) is 8 inches, stiffener width ($w$) is 1 inch, and the stiffeners extend the entire length of the plate (32 inches). The resulting stiffener thickness ($t_s$) is calculated using $\mu$ from Equation (4.1) and shown in Equation 4.2 [30]. This $\mu$ represents the ratio of stiffener stiffness to plate stiffness. By varying the $\mu$ values, effect of changes in dimensions of the rectangular stiffener and variations in its material stiffness can be studied using the design plots.

$$\mu = \frac{StiffenerStiffness}{PlateStiffness} = \frac{wt_s E_s}{wt E_s + bt E}$$ (4.1)
In this research, the stiffened plate is modeled using symmetry conditions that reduce the size of the problem. This also allows for a crack to be modeled as an edge crack and extend under four unbroken stiffeners. Due to certain remeshing difficulties while propagating the crack through the stiffeners, stiffener surface close to the crack had to be un-adhered, as shown in Figure 4.1 (area B).

\[
t_s = \frac{b\mu}{w(1-\mu)}
\]  

(4.2)

Following the steps outlined in Appendix D, the stress intensity factor for the stiffened plate for various crack lengths can be obtained. The resulting stress intensity factors are then normalized using the stress intensity factor of a flat plate to obtain the geometric factor, \(\beta(a)\) as shown in Equation (4.4). This geometric factor is plotted with respect to crack length divided by stiffener spacing, Figure 4.2. The deformed mesh for the cracked plate is shown in Figure 4.3. Since the stiffeners were not attached to the plate near the crack only the plate is remeshed as the crack grows through the plate. The goal of the current research is to develop a process of obtaining risk based design plots; therefore, this modeling approach was selected even though this introduces certain errors into the analysis.

<table>
<thead>
<tr>
<th>Material Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>E (psi)</td>
</tr>
<tr>
<td>(\nu)</td>
</tr>
<tr>
<td>thickness (in)</td>
</tr>
<tr>
<td>G (psi)</td>
</tr>
</tbody>
</table>

**Table 4.1:** Material Properties for Each Assigned Area from Figure 4.1
Figure 4.2: Geometric factor ($\beta(a)$) Plot from FRANC-2DL
4.2 Failure Criteria for Stiffened Plates

One of the design criteria to ensure damage tolerance of aircraft structures is the capability to sustain a two bay skin crack with a broken stiffener. In this research, only intact stiffeners were considered but future work can incorporate a broken stiffener model into this process. The stress intensity factor required for residual strength calculations are
obtained from FRANC-2DL as explained in the above section. The failure criterion used in this research to determine the probability of fracture is as follows:

\[
P_{F_{\text{Stiffened}}} = P[R.S. - \text{Stress} < 0] \tag{4.3a}
\]

\[
P_{F_{\text{Stiffened}}} = P\left[\frac{K_{CR}}{\sqrt{\pi a} \beta(a)} - \sigma < 0\right] \tag{4.3b}
\]

Where \(K_{CR}\) is the critical stress intensity factor, \(a\) is the half crack length, and \(\beta(a)\) is the geometric effect on stress intensity. In this research, the geometric effect due to stiffeners is estimated as follows:

\[
\beta(a) = \frac{K_{\text{Stiffened}}}{\sigma \sqrt{\pi a}} \tag{4.4}
\]

Where \(K_{\text{Stiffened}}\) is determined using FRANC-2DL and the denominator is the stress intensity factor for an unstiffened flat plate.

Equation (4.3b) can be rewritten as:

\[
P_{F_{\text{Stiffened}}} = P[cR - S < 0] \tag{4.5a}
\]

\[
c = \frac{1}{\sqrt{\pi a} \beta(a)} \tag{4.5b}
\]

\[
R = K_{CR} \tag{4.5c}
\]

\[
S = \sigma \tag{4.5d}
\]

This is clearly in the same form as Equations (3.9) and (3.10) from Chapter 3, and therefore can be easily integrated using the FFT based convolution technique discussed in Chapter 3.
4.3 Risk Based Design Plots for Stiffened Plates

Similar to the RBD plots that were obtained for flat plates in Chapter 3, RBD plots for stiffened plates provide information about the variation of probability of failure as a function of crack length over stiffener spacing. As shown in Figure 4.4, variation of probability of fracture for different $\mu$ (Equation (4.1)) can be determined.

In this example, an applied uniaxial stress of 48 ksi and mean fracture toughness of $100 \text{ksi in}^{1/2}$ [31] were selected. Using these plots, the required stiffness for the stringers that would result in an allowable probability of fracture for a two bay crack for a given stringer spacing ($b = 8''$ in this example) can be easily located. These plots can also be used for analysis given a certain configuration of stiffened plates. By varying the stiffener spacing or other geometric dimensions, these plots can be parameterized so that an optimal configuration of the stiffened plate can be selected based on allowable failure probability. Therefore, these design plots allow for risked based structural sizing without the need for expensive risk integrated optimization routines.
4.4 Summary

Using the stress intensity factors from FRANC-2DL and FFT based risk assessment technique RBD plots were created for a specific stiffened plate with cracks extending across multiple bays. The resulting plots can be used to analyze preexisting stiffened plates or set up a design basis for the stiffened plates by incorporating the risk of failure into the design before construction. The steps established in this Chapter give a means to progress with various parametric studies for stiffened plates and incorporate risk of failure into these stiffened plates.
5. Conclusion

In this research, details of a new internal load PDF modeling technique that requires only one stiffness matrix inversion when dealing with static analysis was introduced, reducing and possibly eliminating errors by doing away with function approximations. Using this technique, variations in the external air loads can be translated into variations in internal loads with minimal effort. These internal load variations are used to determine the failure probability of various aircraft structural components. RBD plots were created for both flat plates and stiffened plates with use of FFT based integration technique. The resulting methodology increased efficiency of calculations by incorporating risk of failure into the analysis and design of flat and stiffened plates. Using these plots, the designer can obtain information about the failure probability and its relation to all the structural changes. This improves the decision making process because the designer is well informed about all the possible consequences of design changes.

Since the design criteria for aircraft structures involve more than just fracture additional failure modes need to be incorporated into these design plots. This research is an attempt to develop a process that can be applied to other failure modes. Using the methodology developed, various configurations of stiffened panels can be examined for different failure criteria as shown in Figure 5.1. Other complexities that can be added to the stiffened plate are different types of stiffeners, i.e. hat, C, Z (Figure 5.2); stiffener connections, i.e. welding, rivets, single form; and different materials systems like
composites or hybrid. These variations can then be applied to curved plates that are typical of fuselage sections. Since the RBD plots developed in this research are normalized, trade-offs between these designs can easily be investigated.

**Figure 5.1:** Failure Modes for a Stiffened Plate

**Figure 5.2:** Types of Stiffeners
APPENDIX A

Nastran file for extracting grid point force (GPFORCE) at nodes 39005, 39037, 950371, & 950768

ASSIGN OUTPUT2='T38.op2' UNIT=12 FORM=UNFORMATTED
SOL 101
TIME 60
CEND
TITLE = MSC
SPC = 1
LOAD = 66610
SET 21 = 39005, 39037, 950371, 950768
DISPLACEMENT = ALL
STRESS = ALL
SPCFORCES = ALL
GPFORCE(PUNCH)=21
$*
$*
$*$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$*
$* BULK DATA
$*
$*$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$*
BEGIN BULK
$*
APPENDIX B

NASTRAN code that is essential for extracting the Stiffness Matrix (KLL) from a previously run file to be used with MatLab code: createdatabases.m (APPENDIX C)

RESTART VERSION=LAST,KEEP $ last file run
assign master='**.MASTER'
ASSIGN OUTPUT4='matfile', UNIT=12, FORM=FORMATTED, DELETE
ASSIGN OUTPUT4='rhsfile', UNIT=13, FORM=FORMATTED, DELETE
ID MSC,UM531 $ EXAMPLE
TIME 600
DIAG 8 $ PRINT MATRIX TRAILERS AND RECOVERED DATA BLOCKS
DIAG 31 $ PRINT MODULE PROPERTIES LIST (MPL)
SOL 100

MALTER 'MALTER:USERDMAP'
TYPE PARM,NDDL,I,PEID,MPC,SPC,LOAD,LUSETS,SEID $
TYPE DB,CSTM,PG,KLL,PL,ECTS,GPECT,SILS,GPLS $
TYPE DB,EST,KGG,BGPDTS,EQEXINS,GPDTS,USET $
TYPE DB,ETT,KFS $
PEID=0 $
SEID=0 $
MPC=0 $ 
SPC=2 $ 
LOAD=2 $ 
$MATPRT KLL///1 $ MATPRT OF KLL
OUTPUT4 KLL,,,//-1/-12/0/TRUE/9 $ Unit 12, may need FMS statement
OUTPUT4 PL,,,//-1/13/0/TRUE/9 $ Unit 13, may need FMS statement
$MATPRT PL//0/V,Y,NOPRT=-1 $ OPTIONALLY PRINT PL BY COLUMNS
ENDALTER
LINK USERDMAP,INCL=MSCOBJ $
CEND

TITLE = THIS ILLUSTRATES THE OUTPUT TYPES UM531
LABEL = DMAP DOES NOT USE MUCH FROM CASE CONTROL DECK

BEGIN BULK
PARAM,NOPRT,1 $ PRINT PG THIS TIME
PARAM,UNUSED,1 $ UNUSED PARAMETER
ENDDATA
% Heather B. Dwire  
% Created: December 4, 2006  
% Editted: December 12, 2006  
%  
% To make external load and internal load relations  
clear all  
close all  
clear  
iteration = 0;  
%  
% currently the Nastran10556Mod.dat has a set 21 which limits the program  
% to only four nodes being analyzed.  
%  
% open file(s) needed for reading and writing  
fid_in = fopen('Nastran10556Mod.dat', 'r');  
fid_out = fopen(sprintf('AllForceFile.dat'),'w');  
% aids in finding the FORCE values  
found_force = 0;  
% this reads entire file into the line 'file' (x amount of 'force' lines)  
while 1  
    file = fgets(fid_in);  
    if ~ischar(file), break, end  
    % need to check to make sure the matrix is large enough to  
    % achieve the funtion in the if statement when looking for FORCE  
    if size(file) < 5  
        ...  
    elseif file(1:5) == 'FORCE'  
        % equation gives number of FORCE values located in the file  
        found_force = found_force + 1;  
    end  
    %prints to file (currently blank.dat, with fid_out)  
    fprintf(fid_out, '%s', file);  
    % displays line 'file' to MatLab screen  
    disp(file)  
end  
% closes the file in and out  
fclose(fid_in);  
fclose(fid_out);  
%  
% this 'for' loop will create a file to analyse a structure under  
% individual loads  
for k=1:found_force  
    fid_in = fopen('Nastran10556Mod.dat', 'r');  

fid_single = fopen(sprintf('single_force_file.dat'),'w');
file = fgets(fid_in);

% write main bulk of file to newly created file
while (feof(fid_in)~=1)
    if(file(1) == 'F' & file(2) == 'O' & file(3) == 'R' & file(4) == 'C')
        break;
    else
        fprintf(fid_single, '%s', file);
        file = fgets(fid_in);
    end
end

% now a 'for' loop to consider the 'FORCE' cards
for temp = 1:found_force
    if temp == k
        % added to achieve the force value (and node at which it is
        % applied. Jason helped here (EE major).
        len = length(file);
        comma_count = 0;
        for i=1:len
            if(file(i) == ',')
                comma_count = comma_count + 1;
                if(comma_count == 6)
                    force_card = str2num(file((i+1):end));
                end
            end
        end
        [node] = strread(file, '%*s %*f %*f %*c %*c %*c %*f', 'delimiter', ',');
        fprintf(fid_single, '%s', file);
    end
end

% to finished writing needed data from original to new file
while 1
    fprintf(fid_single, '%s', file);
    file = fgets(fid_in);
    if ~ischar(file), break, end
end

% need to close the fid_single file so it can be run in Nastran
% and results can be acquired
fclose(fid_single);
fclose(fid_in);

system('rm -f *.f06 *.f04 *.pch *.DBA* *.MAST* *.op2 *.log')
% NOW run fid_single = single_force_file.dat file in NASTRAN
system('nastran single_force_file.dat batch=no')

% take resulting .pch file and get needed values
fid_pch = fopen('single_force_file.pch','r')
count = interation;

while (feof(fid_pch)~=1)
    file = fgets(fid_pch);
    if ((file(1)=='') && (file(37)=='F' | file(37) == '*'))
        continue;
    elseif (file(1)=='')
        fline = sscanf(file, '%d %d %s %d');
        % first nodes
        if (fline(1)== 39005 & fline(2)== 99101752)
            count = count + 1;
            F_x(count, 1) = 39005;
            F_x(count, 2) = 99101752;
            F_x(count, 4) = force_card;
            F_y(count, 1) = 39005;
            F_y(count, 2) = 99101752;
            F_y(count, 4) = force_card;
            F_z(count, 1) = 39005;
            F_z(count, 2) = 99101752;
            F_z(count, 4) = force_card;
            file = fgets(fid_pch);
            xc=sscanf(file,'%s %e %e %e %d');
            F_x(count, 3) = xc(7);
            F_y(count, 3) = xc(8);
            F_z(count, 3) = xc(9);
            file = fgets(fid_pch);
            xc=sscanf(file,'%s %e %e %e %d');
            M_x(count, 3) = xc(7);
            M_y(count, 3) = xc(8);
            M_z(count, 3) = xc(9);
        elseif (fline(1)== 39005 & fline(2)== 99101684)
            count = count + 1;
            F_x(count, 1) = 39005;
            F_x(count, 2) = 99101684;
            F_x(count, 4) = force_card;
            F_y(count, 1) = 39005;
            F_y(count, 2) = 99101684;
            F_y(count, 4) = force_card;
            F_z(count, 1) = 39005;
            F_z(count, 2) = 99101684;
            F_z(count, 4) = force_card;
            M_x(count, 1) = 39005;
            M_x(count, 2) = 99101684;
            M_x(count, 4) = force_card;
            M_y(count, 1) = 39005;
            M_y(count, 2) = 99101684;
            M_y(count, 4) = force_card;
            M_z(count, 1) = 39005;
            M_z(count, 2) = 99101684;
M_y(count, 4) = force_card;
M_z(count, 1) = 39005;
M_z(count, 2) = 99101684;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = x(7);
M_y(count, 3) = x(8);
M_z(count, 3) = x(9);
elseif (fline(1) == 39005 & fline(2) == 99102224)
    count = count + 1;
    F_x(count, 1) = 39005;
    F_x(count, 2) = 99102224;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 39005;
    F_y(count, 2) = 99102224;
    F_y(count, 4) = force_card;
    F_z(count, 1) = 39005;
    F_z(count, 2) = 99102224;
    F_z(count, 4) = force_card;
    M_x(count, 1) = 39005;
    M_x(count, 2) = 99102224;
    M_x(count, 4) = force_card;
    M_y(count, 1) = 39005;
    M_y(count, 2) = 99102224;
    M_y(count, 4) = force_card;
    M_z(count, 1) = 39005;
    M_z(count, 2) = 99102224;
    M_z(count, 4) = force_card;
    file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = x(7);
F_y(count, 3) = x(8);
F_z(count, 3) = x(9);
file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = x(7);
M_y(count, 3) = x(8);
M_z(count, 3) = x(9);
else:
    if (fline(1) == 39005 & fline(2) == 99102227)
        count = count + 1;
        F_x(count, 1) = 39005;
        F_x(count, 2) = 99102227;
        F_x(count, 4) = force_card;
        F_y(count, 1) = 39005;
        F_y(count, 2) = 99102227;
        F_y(count, 4) = force_card;
        F_z(count, 1) = 39005;
        F_z(count, 2) = 99102227;
        F_z(count, 4) = force_card;
        M_x(count, 1) = 39005;
M_x(count, 2) = 99102227;
M_x(count, 4) = force_card;
M_y(count, 1) = 39005;
M_y(count, 2) = 99102227;
M_y(count, 4) = force_card;
M_z(count, 1) = 39005;
M_z(count, 2) = 99102227;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc = sscanf(file, "%s %e %e %e %d");
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, "%s %e %e %e %d");
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);

% New nodes
elseif (fline(1)== 39037 & fline(2)== 99101107)
    count = count + 1;
    F_x(count, 1) = 39037;
    F_x(count, 2) = 99101107;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 39037;
    F_y(count, 2) = 99101107;
    F_y(count, 4) = force_card;
    F_z(count, 1) = 39037;
    F_z(count, 2) = 99101107;
    F_z(count, 4) = force_card;
    M_x(count, 1) = 39037;
    M_x(count, 2) = 99101107;
    M_x(count, 4) = force_card;
    M_y(count, 1) = 39037;
    M_y(count, 2) = 99101107;
    M_y(count, 4) = force_card;
    M_z(count, 1) = 39037;
    M_z(count, 2) = 99101107;
    M_z(count, 4) = force_card;
    file = fgets(fid_pch);
xc = sscanf(file, "%s %e %e %e %d");
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, "%s %e %e %e %d");
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1)== 39037 & fline(2)== 601011)
    count = count + 1;
    F_x(count, 1) = 39037;
    F_x(count, 2) = 601011;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 39037;
    F_y(count, 2) = 601011;
F_y(count, 4) = force_card;
F_z(count, 1) = 39037;
F_z(count, 2) = 601011;
F_z(count, 4) = force_card;
M_x(count, 1) = 39037;
M_x(count, 2) = 601011;
M_x(count, 4) = force_card;
M_y(count, 1) = 39037;
M_y(count, 2) = 601011;
M_y(count, 4) = force_card;
M_z(count, 1) = 39037;
M_z(count, 2) = 601011;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1) == 39037 & fline(2) == 99101853)
    count = count + 1;
    F_x(count, 1) = 39037;
    F_x(count, 2) = 99101853;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 39037;
    F_y(count, 2) = 99101853;
    F_y(count, 4) = force_card;
    F_z(count, 1) = 39037;
    F_z(count, 2) = 99101853;
    F_z(count, 4) = force_card;
    file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
    F_x(count, 3) = xc(7);
    F_y(count, 3) = xc(8);
    F_z(count, 3) = xc(9);
    file = fgets(fid_pch);
x = sscanf(file, '%s %e %e %e %d');
    M_x(count, 3) = xc(7);
    M_y(count, 3) = xc(8);
    M_z(count, 3) = xc(9);
elseif (fline(1) == 39037 & fline(2) == 99101862)
    count = count + 1;
    F_x(count, 1) = 39037;
\[
F_x(\text{count}, 2) = 99101862;
F_x(\text{count}, 4) = \text{force\_card};
F_y(\text{count}, 1) = 39037;
F_y(\text{count}, 2) = 99101862;
F_y(\text{count}, 4) = \text{force\_card};
F_z(\text{count}, 1) = 39037;
F_z(\text{count}, 2) = 99101862;
F_z(\text{count}, 4) = \text{force\_card};
M_x(\text{count}, 1) = 39037;
M_x(\text{count}, 2) = 99101862;
M_x(\text{count}, 4) = \text{force\_card};
M_y(\text{count}, 1) = 39037;
M_y(\text{count}, 2) = 99101862;
M_y(\text{count}, 4) = \text{force\_card};
M_z(\text{count}, 1) = 39037;
M_z(\text{count}, 2) = 99101862;
M_z(\text{count}, 4) = \text{force\_card};
\]

file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);

% New Nodes
elseif (fline(1) == 950768 & fline(2) == 99101751)
count = count + 1;
F_x(count, 1) = 950768;
F_x(count, 2) = 99101751;
F_x(count, 4) = \text{force\_card};
F_y(count, 1) = 950768;
F_y(count, 2) = 99101751;
F_y(count, 4) = \text{force\_card};
F_z(count, 1) = 950768;
F_z(count, 2) = 99101751;
F_z(count, 4) = \text{force\_card};
M_x(count, 1) = 950768;
M_x(count, 2) = 99101751;
M_x(count, 4) = \text{force\_card};
M_y(count, 1) = 950768;
M_y(count, 2) = 99101751;
M_y(count, 4) = \text{force\_card};
M_z(count, 1) = 950768;
M_z(count, 2) = 99101751;
M_z(count, 4) = \text{force\_card};
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1)== 950768 & fline(2)== 99102223)
count = count + 1;
F_x(count, 1) = 950768;
F_x(count, 2) = 99102223;
F_x(count, 4) = force_card;
F_y(count, 1) = 950768;
F_y(count, 2) = 99102223;
F_y(count, 4) = force_card;
F_z(count, 1) = 950768;
F_z(count, 2) = 99102223;
F_z(count, 4) = force_card;
M_x(count, 1) = 950768;
M_x(count, 2) = 99102223;
M_x(count, 4) = force_card;
M_y(count, 1) = 950768;
M_y(count, 2) = 99102223;
M_y(count, 4) = force_card;
M_z(count, 1) = 950768;
M_z(count, 2) = 99102223;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1)== 950768 & fline(2)== 99102225)
count = count + 1;
F_x(count, 1) = 950768;
F_x(count, 2) = 99102225;
F_x(count, 4) = force_card;
F_y(count, 1) = 950768;
F_y(count, 2) = 99102225;
F_y(count, 4) = force_card;
F_z(count, 1) = 950768;
F_z(count, 2) = 99102225;
F_z(count, 4) = force_card;
M_x(count, 1) = 950768;
M_x(count, 2) = 99102225;
M_x(count, 4) = force_card;
M_y(count, 1) = 950768;
M_y(count, 2) = 99102225;
M_y(count, 4) = force_card;
M_z(count, 1) = 950768;
M_z(count, 2) = 99102225;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
\begin{verbatim}
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);

elseif (fline(1)== 950768 & fline(2)== 99101683)
    count = count + 1;
    F_x(count, 1) = 950768;
    F_x(count, 2) = 99101683;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 950768;
    F_y(count, 2) = 99101683;
    F_y(count, 4) = force_card;
    F_z(count, 1) = 950768;
    F_z(count, 2) = 99101683;
    F_z(count, 4) = force_card;
    M_x(count, 1) = 950768;
    M_x(count, 2) = 99101683;
    M_x(count, 4) = force_card;
    M_y(count, 1) = 950768;
    M_y(count, 2) = 99101683;
    M_y(count, 4) = force_card;
    M_z(count, 1) = 950768;
    M_z(count, 2) = 99101683;
    M_z(count, 4) = force_card;
    file = fgets(fid_pch);
    xc=sscanf(file,'%s %e %e %e %d');
    F_x(count, 3) = xc(7);
    F_y(count, 3) = xc(8);
    F_z(count, 3) = xc(9);
    file = fgets(fid_pch);
    xc=sscanf(file,'%s %e %e %e %d');
    M_x(count, 3) = xc(7);
    M_y(count, 3) = xc(8);
    M_z(count, 3) = xc(9);
    % New nodes

elseif (fline(1)== 950371 & fline(2)== 99101111)
    count = count + 1;
    F_x(count, 1) = 950371;
    F_x(count, 2) = 99101111;
    F_x(count, 4) = force_card;
    F_y(count, 1) = 950371;
    F_y(count, 2) = 99101111;
    F_y(count, 4) = force_card;
    F_z(count, 1) = 950371;
    F_z(count, 2) = 99101111;
    F_z(count, 4) = force_card;
    M_x(count, 1) = 950371;
    M_x(count, 2) = 99101111;
    M_x(count, 4) = force_card;
    M_y(count, 1) = 950371;
    M_y(count, 2) = 99101111;
    M_y(count, 4) = force_card;
    M_z(count, 1) = 950371;
    M_z(count, 2) = 99101111;
\end{verbatim}
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1)== 950371 & fline(2)== 601012)
count = count + 1;
F_x(count, 1) = 950371;
F_x(count, 2) = 601012;
F_x(count, 4) = force_card;
F_y(count, 1) = 950371;
F_y(count, 2) = 601012;
F_y(count, 4) = force_card;
F_z(count, 1) = 950371;
F_z(count, 2) = 601012;
F_z(count, 4) = force_card;
M_x(count, 1) = 950371;
M_x(count, 2) = 601012;
M_x(count, 4) = force_card;
M_y(count, 1) = 950371;
M_y(count, 2) = 601012;
M_y(count, 4) = force_card;
M_z(count, 1) = 950371;
M_z(count, 2) = 601012;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc=sscanf(file,'%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1)== 950371 & fline(2)== 99101851)
count = count + 1;
F_x(count, 1) = 950371;
F_x(count, 2) = 99101851;
F_x(count, 4) = force_card;
F_y(count, 1) = 950371;
F_y(count, 2) = 99101851;
F_y(count, 4) = force_card;
F_z(count, 1) = 950371;
F_z(count, 2) = 99101851;
F_z(count, 4) = force_card;
M_x(count, 1) = 950371;
M_x(count, 2) = 99101851;
M_x(count, 4) = force_card;
M_y(count, 1) = 950371;
M_y(count, 2) = 99101851;
M_y(count, 4) = force_card;
M_z(count, 1) = 950371;
M_z(count, 2) = 99101851;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
elseif (fline(1) == 950371 & fline(2) == 99101861)
count = count + 1;
F_x(count, 1) = 950371;
F_x(count, 2) = 99101861;
F_x(count, 4) = force_card;
F_y(count, 1) = 950371;
F_y(count, 2) = 99101861;
F_y(count, 4) = force_card;
F_z(count, 1) = 950371;
F_z(count, 2) = 99101861;
F_z(count, 4) = force_card;
M_x(count, 1) = 950371;
M_x(count, 2) = 99101861;
M_x(count, 4) = force_card;
M_y(count, 1) = 950371;
M_y(count, 2) = 99101861;
M_y(count, 4) = force_card;
M_z(count, 1) = 950371;
M_z(count, 2) = 99101861;
M_z(count, 4) = force_card;
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
F_x(count, 3) = xc(7);
F_y(count, 3) = xc(8);
F_z(count, 3) = xc(9);
file = fgets(fid_pch);
xc = sscanf(file, '%s %e %e %e %d');
M_x(count, 3) = xc(7);
M_y(count, 3) = xc(8);
M_z(count, 3) = xc(9);
end

iteration = count;
end

fclose(fid_pch);
end

save(sprintf('FORCE_file.mat'), 'F_x', 'F_y', 'F_z', 'M_x', 'M_y', 'M_z')
APPENDIX D

Steps to produce crack propagation across entire stiffened plate in FRANC-2DL

Plate dimensions: 32in x 32in x 0.5in centered about (0,0)

Stiffener frame dimensions: 25in x 32in x 4in (Mu = 0.5), stiffener spacing 8in, stiffener width 1in, connections between stiffeners are 0.5in, all centered about (0,0)

1. Import combined *.inp file from CASTOFRANC translation into FRANC-2DL
   a. To ensure memory is available use a command prompt window to open FRANC-2DL: Open up a Command Prompt window (under Accessories)
   b. Go to the location of the franc2dL program
   c. Use the following command once in the correct directory:

      franc2dl –mem 300000000

   d. Enter in the file name when prompted

2. PRE-PROCESS – MATERIAL
   a. 2 materials are pre-existing (one for each layer), assign material properties from Table 4.1 for each respective layer, i.e. layer 1 (plate) is D, layer 2 (stiffener frame) is C.
   b. create 2 new materials: 1 adhesive and one material with negligible stiffness
      i. Material (3), NEW MAT – ADHESIVE – assign material properties from Table 4.1
ii. Material (4), NEW MAT – ELAST ISO – assign material properties from Table 4.1 (material A)

c. Assign material properties from the newly created material (4) to areas from Figure 4.1 labeled A in layer 2, be sure layer 2 is shown

i. Check that Material (4) is chosen by using the – MAT + option

ii. Choose SWITCH ELEM in the options bar and then pick all boxes in mesh which are NOT part of the stiffener(s). These boxes should now show the number (4) instead of the number (2).

d. RETURN to main menu

3. PRE-PROCESS – FIXITY (these are the boundary conditions)

a. In layer 1, FIX EDGE – Y – choose bottom edge of layer 1

b. In layer 1, FIX EDGE – X – choose left edge of layer 1 (symmetry B.C.)

c. **Switch to layer 2. In layer 2, FIX EDGE – Y – choose bottom edge of stiffener only (not area A from Figure 4.1). This process will be repeated for each stiffener edge.

d. RETURN to main menu

4. PRE-PROCESS – LOADS (these are the applied stresses)


b. **Switch to Layer 2: DIST LOAD – Y GLOBAL – CONSTANT – top edge of each stiffener section - enter stress applied (48 ksi = 48 EEX 3) – repeat this for each stiffener.

c. RETURN to main menu
5. Check layer 1 is displayed: MODIFY – ADD ADHESIVE – TOGGLE ALL
   a. Error “Unable to add adhesive for elements ####” This should not effect analysis
   b. While still in the ADD ADHESIVE menu option
      i. Un-toggle by hand the four boxes which the crack will propagate through, two above the crack line and two below the crack line
      ii. This step may be made easier by first doing step 6 and then returning to step 5b to un-toggle the boxes of the mesh above and below the crack
   c. RETURN to main menu

6. Apply Automatic crack propagation for Stress Intensity factor history output values
   a. Check layer 1 is displayed: MODIFY – NEW CRACK – NON-COHESIVE – EDGE CRACK
      i. click the TOLERANCE button at the bottom of the screen on the (-) side, this will place a box at (0,0), then select the far left edge on the x-axis. This step will verify the x-axis is chosen and the crack will propagate correctly.
      ii. Prompted to chose DONE
      iii. Enter the (x,y) values when prompted for crack tip placement (used a 0.2 inch initial crack, so the (x,y) = (-15.8, 0))
      iv. Enter 2 for the prompt “Specify minimum number of elements along the crack extension.”
      v. Click CONTINUE 3 times and then ACCEPT the resulting mesh
vi. RETURN to main menu (here is where step 5b can be executed more easily)

b. MODIFY – MOVE CRACK – AUTOMATIC
   
i. CRACK INCR – enter 0.5 (inches)
   
ii. STEPS: ## - enter 58
   
iii. PROPAGATE (and wait)
   
iv. RETURN to main menu

7. POST-PROCESS – FRACT MECH – SIF HISTORY – KI – this gives a plot of the Stress Intensity Factor with respect to crack length.
   
a. KI values can be saved to a file which can be read in Notepad by choosing J-FILE and saving with a file name
   
b. Also in POST-PROCESS, the DEFORMED MESH of the cracked plate can be seen.
   
c. RETURN to main menu, Save results: WRITE FILE – enter a file name

8. To read saved J-file and normalize the KI values
   
a. Open saved J-file in Notepad and re-save as a *.txt document so it can be accessed by Matlab code in Appendix E
   
b. Open MatLab, be sure code and new saved *.txt file are saved in the same directory
   
c. Run MatLab code for resulting plot of $\frac{\text{KI}}{(\sigma\sqrt{\pi a})}$ vs. crack length/stiffener spacing.

** To switch layers, RETURN to previous menu until – LAYER + option is available
% Heather B. Dwire
% Created: February 2, 2008
% Edited: February 4, 2008
% j-file reader/writer: ReadWriteJ.m
% read points, make matrices, normalize and plot
clear all
close all
clc
i = 0;
j = 0;
% read file for amount of steps
% fid_in = fopen('layers3232C16Results.txt', 'r')
% different MU used for layers3232C16Results.txt, MU = 0.8
fid_in = fopen('3232CSymResults.txt', 'r')
%
while 1
  file = fgets(fid_in);
  % breaks while loop
  if ~ischar(file), break, end
  % gets the # of step increments for cracks
  if size(file) < 5
    ... %
  elseif file(2:8) == 'Crack #'
    % get next line after line which starts with 'Crack'
    file = fgets(fid_in);
    % read values of line
    line = sscanf(file, '%d %d %e %e');
    steps1 = line(2);
  else
    ...
  end
%
% Finds all the lines starting with 'Step'
% Crack length values at the end of this line
if size(file) < 5
%  ...
elseif file(2:5) == 'Step'
  i = i+1;
  % turn line into values so crack length can be saved
  % cl = crack length
  cl = sscanf(file(30:39), '%e');
  % Make Matrix for Crack Lengths
  length(i,1) = cl; % l(row, column)
else
  file(10:14) == 'Total'
  % Now get KI values for each crack length
  j = j+1;
}
% get KI values for corresponding crack length
% ki = stress intensity factor for crack length
line = sscanf(file, '%*s %e %e %e %e ');
ki = line(1);
% Make Matrix for both Crack Lengths and KI values
KI(j,1) = ki;
else
    ...
% subsindex problem
end

% Collect matrix data
% save(sprintf('Matrix3232C16.mat'), 'steps1','length','KI')
save(sprintf('Matrix3232C2.mat'), 'steps1','length','KI')
clear all
close all
clc
load Matrix3232C2.mat
% load Matrix3232C2.mat
% Create the plot for S
% Stiffened panel stress intensity from Displacement Compatibility method
for k = 1:1:steps1
    % note that steps1 needs to be steps1*2 when dealing with internal
% crack and not an edge crack
    GF(k,1) = (1/(48000*sqrt(pi*length(k,1))))*KI(k,1);
end
aoverb = (1/8)*length;
% bay width is 8 inches
plot(aoverb(1:steps1), GF(1:steps1), '-*b', 'LineWidth',2, 'MarkerSize',6)
grid on
xlabel('crack length / stiffener spacing')
ylabel('KI/(\sigma_0\sqrt{\pi a})')
legend('\mu = 0.5')
% title('Stress Intensity plot from FRANC')
%
% used to modify and save data to be used in FFT convolution
% for z = 3:steps1+1
% GeoFact(1) = 1.00;
% ab(1) = 0.0;
% GeoFact(2) = 1.00;
% ab(2) = aoverb(1);
% GeoFact(z) = GF(z-1);
% ab(z) = aoverb(z-1);
% end
% GeoFact = GeoFact';
% ab = ab';
% save(sprintf('Need3232C16.mat'), 'GeoFact','ab')
% %save(sprintf('Need3232C2.mat'), 'GeoFact','ab')
REFERENCES


