

# fe-safe 2017

USER GUIDE APPENDICES



**3DEXPERIENCE**<sup>®</sup>



## Trademarks

---

*fe-safe*, Abaqus, Isight, Tosca, the 3DS logo, and SIMULIA are commercial trademarks or registered trademarks of Dassault Systèmes or its subsidiaries in the United States and/or other countries. Use of any Dassault Systèmes or its subsidiaries trademarks is subject to their express written approval. Other company, product, and service names may be trademarks or service marks of their respective owners.

## Legal Notices

*fe-safe* and this documentation may be used or reproduced only in accordance with the terms of the software license agreement signed by the customer, or, absent such an agreement, the then current software license agreement to which the documentation relates.

This documentation and the software described in this documentation are subject to change without prior notice.

Dassault Systèmes and its subsidiaries shall not be responsible for the consequences of any errors or omissions that may appear in this documentation.

© Dassault Systèmes Simulia Corp, 2016.

### Third-Party Copyright Notices

Certain portions of *fe-safe* contain elements subject to copyright owned by the entities listed below.

© Battelle
© Endurica LLC
© Amec Foster Wheeler Nuclear UK Limited

*fe-safe* Licensed Programs may include open source software components. Source code for these components is available if required by the license.

The open source software components are grouped under the applicable licensing terms. Where required, links to common license terms are included below.

IP Asset Name	IP Asset Version	Copyright Notice
<a href="#">Under BSD 2-Clause</a>		
UnZip (from Info-ZIP)	2.4	Copyright (c) 1990-2009 Info-ZIP. All rights reserved.
<a href="#">Under BSD 3-Clause</a>		
Qt Solutions	2.6	Copyright (c) 2014 Digia Plc and/or its subsidiary(-ies) All rights reserved.

## 201 Appendix A - Notation

Notation generally follows accepted fatigue notation.

The first main definition of any term is shown as a section number in the Fatigue Theory Reference Manual.

Where the notation of one section conflicts with the general notation, or is restricted to one section, the section is shown in brackets (for example,  $S$  is generally used to denote nominal stress, but in section 7 the notes follow the accepted notation that  $S$  is a deviatoric true stress).

$a$	crack length, or half crack length	(10)
$A$	amplitude ratio	1
$A$	specimen area under load	2
$A_0$	specimen area at zero load	2
$b$	fatigue strength exponent (Basquin's exponent)	2
$c$	fatigue ductility exponent (Coffin-Manson exponent)	2
$e$	engineering strain or nominal strain	2
$E$	elastic (Young's) modulus	
$F(t)$	cumulative probability of failure	(9)
$K$	stress intensity factor	(10)
$K$	strain hardening coefficient	2
$K'$	cyclic strain hardening exponent	2
$K_f$	fatigue strength reduction factor	6
$k_0$	constant of an S-N curve	11
$K_t$	stress concentration factor	6
$K_{t\text{gross}}$	gross section stress concentration	6
$K_{t\text{nett}}$	nett section stress concentration	6
$m$	slope of S-N curve	11
$m_n$	nth moment of the PSD	(12)
$n$	number of applied cycles	1
$n$	strain hardening exponent	2
$n'$	cyclic strain hardening exponent	2
$N, N_f$	cycles to crack initiation or failure	2
$P$	load	
$p(x)$	probability density of $x$	(9)
$R$	stress ratio	5
$R(t)$	cumulative probability of survival	(9)
$S$	deviatoric (true) stress	(7)
$S_{a0}$	nominal stress amplitude at $S_m = 0$	
$S_{\text{ult}}$	ultimate tensile stress	
$S_{\text{max}}$	maximum nominal stress	5

## Appendix A

$S_{\min}$	minimum nominal stress	5
$S_a$	nominal stress amplitude	5
$S_m$	mean nominal stress	5
UTS	ultimate tensile stress	
$Z(t)$	risk of failure, or hazard function	(9)
$2N_f$	reversals to crack initiation	2
$\Delta\varepsilon$	true strain range	2
$\Delta\sigma$	true stress range	2
$\Delta S$	nominal stress range	5
$\varepsilon$	true strain	2
$\varepsilon_e$	elastic component of strain	2
$\varepsilon_f$	true fracture ductility	2
$\varepsilon_f'$	fatigue ductility coefficient	2
$\varepsilon_p$	plastic component of strain	2
$\varepsilon_t$	total strain	2
$\gamma$	shear strain	7
$\lambda_o$	number of positive slope zero crossings/unit time	(12)
$\mu$	mean value	(9)
$\mu$	number of peaks per unit time	(12)
$\nu$	Poisson's ratio	7
$\sigma$	standard deviation	(9,11,12)
$\sigma$	true stress	2
$\sigma_e$	equivalent stress	(7)
$\sigma_e$	elastic stress	2
$\sigma_f$	true fracture stress	2
$\sigma_f'$	fatigue strength coefficient	2
$\sigma_{ij}$	stress tensor	7
$\sigma_{\max}$	max true stress (usually in a cycle)	2
$\sigma_y$	yield stress	
$\sigma_1, \sigma_2, \sigma_3$	principal stresses	7
$\tau$	shear stress	7

## 202 Appendix B – Nomenclature

### 202.1 Typographic conventions used in this manual

The following typographic conventions are used in this manual:

Convention	Meaning
<b>Bold Text</b>	<ul style="list-style-type: none"> <li>Indicates the name of a dialogue box, control, indicator, graphics label or menu option.</li> </ul> <p>Note: the use of a double-bracket symbol, i.e. <b>&gt;&gt;</b>, indicates a sequence of menu options or dialogue box actions, for example:</p> <p><b>FEA Fatigue &gt;&gt; Analysis Options &gt;&gt; Read strains from FE models</b></p>
<i>Italics</i>	<ul style="list-style-type: none"> <li>Indicates the name of a Dassault Systemes product, for example: <i>fe-safe/Rotate.</i></li> <li>Indicates a reference to a figure, where Figure numbers use the following format: <i>Figure {section number}-{figure number}</i></li> </ul> <p>For example, the reference for the third figure in section 6.4.2 would be: <i>Figure 6.4.2-3</i></p>
Monospace text	<ul style="list-style-type: none"> <li>Indicates a file, directory or path.</li> <li>Indicates the content of a data file, log file or text displayed in the message log window.</li> </ul>
<bold monospace text in brackets>	<ul style="list-style-type: none"> <li>Indicates a reference to a file or directory (see B.2, below).</li> </ul>

### 202.2 References to files and directories

The following nomenclature is used throughout the *fe-safe* documentation when referring to files and directories. This nomenclature is as a shorthand method of referring to files and directories in the documentation only – the names have no meaning in the actual software. Where a path description used in the documentation is intended to be generic, the backwards-slash symbol used in Windows, i.e. \, is used as a directory separator. For Linux installations, this symbol should be interchanged with the forward-slash symbol, i.e. /.

Each of the shorthand file or folder references below (in angled brackets, < >), is followed by a short description of what it refers to.

**<RootDir>** The common **Product Root Installation Directory** (see section 3) may include installations of *fe-safe* Application, *fe-safe* Analysis Server, *fe-safe* Node Administration Utility, *fe-safe* Network Licence Server and *fe-safe* Network Licence Client Utility as well as common files and documentation.

**<InstallDir>** The ***fe-safe* Installation Directory** is established as part of the installation process (see section 3).

The directory `fe-safe` is the ***fe-safe* Installation Directory**.

This directory structure is discussed further in Appendix C.

Linux: On Linux installations, the ***fe-safe* Installation Directory** is determined during the install process. The default location is:

`<RootDir>/fe-safe/version.x.yy`

Windows: On Windows, the ***fe-safe* Installation Directory** is determined during the install process. The default location is:

`<RootDir>\fe-safe\version.x.yy`

In both cases, the `x` and `yy` refers to *fe-safe* major and minor version numbers.

**<UserName>** This is a user login name.

**<UserDir>** An ***fe-safe* User Directory** is created for each user the first time they run *fe-safe* when logged on with a particular login name.

The location of the user directory depends on the platform on which *fe-safe* is installed. This is discussed in detail in section 3 and Appendix C.

This directory will contain a copy of the local material database from **<LocalDir>** as well as the user's Project Directory.

**<ProjectDir>** Project directory is used to store configuration files for a fatigue analysis together with the loaded FEA Models (`model` directory) and analysis results to maintain a record of the entire analysis and to reference the files later. The default project directory is:

`<UserDir>/projects`

The location of the **Project Directory** can be configured when starting *fe-safe*. The current project path can be displayed in the **Analysis** Options dialogue. This folder can be stored in any location, but it should be noted that it can become quite large. Since it



contains the current working files it is recommended to have it on a local drive for best performance.

**<JobsDir>**

Jobs directory is used to store the project's jobs configuration files and analysis results. The default job directory is:

```
<ProjectDir>/jobs/job_xx
```

**<ResultsDir>**

The **Results File Directory** is the default location for FEA analysis results files. The default results file directory is:

```
<JobsDir>/fe-results
```

The location of the **Results File Directory** can be configured in the **Fatigue from FEA** dialogue.

**<FEDDir>**

The **Loaded FEA Models FED Folder** contains the working *fe-safe* FED folder, called FESAFE.FED.

The default location of the working FED folder is:

```
<ProjectDir>\model\FESAFE.FED
```

**<DataDir>**

The **Sample Data Directory** contains sample data for the tutorials, and is a subdirectory of the installation directory:

```
<InstallDir>\version.x.yy\data
```

**<DatabaseDir>**

The **Material Database Directory** contains the main material database resource supplied with *fe-safe*, and is a subdirectory of the installation directory:

```
<InstallDir>\version.x.yy/database
```

**<DocsDir>**

The **Documentation Directory** contains *fe-safe* user documentation in PDF format, and is a subdirectory of the installation directory:

```
<InstallDir>\version.x.yy/documentation
```

**<ExeDir>**

The **Executable Directory** contains the executable files and libraries used by *fe-safe*, and is a subdirectory of the installation directory:

```
<InstallDir>\version.x.yy\exe
```

**<ExternalDir>**

The **External Libraries Directory** contains additional third-party libraries that *fe-safe* uses to interface to some of the supported third-party products. This directory is a subdirectory of the installation directory:

`<InstallDir>\version.x.yy\external`

**<KtDir>** The **Kt Directory** contains surface finish definition files (see section 5 and Appendix E). This directory is a subdirectory of the installation directory:

`<InstallDir>\version.x.yy\kt`

**<LocalDir>** The **Local Directory** contains default versions of the local material database file (`local.dbase`), the local database template file (`local.template`). These default files are copied to the user directory (**<UserDir>**) for each new user. This directory is a subdirectory of the installation directory:

`<InstallDir>\version.x.yy\Local`

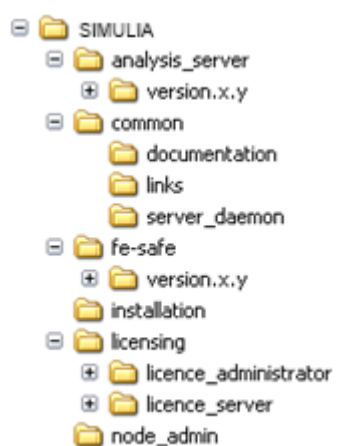
## 203 Appendix C – Software description

**Note:** Parts of this appendix use the nomenclature described in Appendix B as a shorthand method of referring to particular files and directories.

### 203.1 Files and directories used by *fe-safe*

#### 203.1.1 Installed software

When components of the *fe-safe* product family are installed, on any platform, the following directory structure is created in the chosen root directory (see section 3):



#### `\analysis_server\version.x.yy`

This directory contains the *fe-safe* Analysis Server, which is an optional component for Distributed Memory Processing (DMP) analysis. The Distributed Processing is described in detail in separate documentation.

#### `\common`

This directory contains common (shared) files and documentation.

#### `\fe-safe\version.x.yy`

This directory contains *fe-safe* or *safe4fatigue* main product installation.

#### `\installation`

This directory contains product installation information and uninstaller.

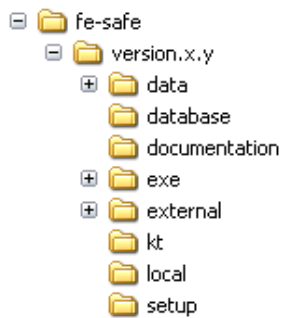
#### `\licensing`

This directory contains the Network Licence Server and Network Licence Client Utility, which are optional components for serving and managing product licensing, see section 4.

### `\node_admin`

This directory contains a utility tool for managing and monitoring of analysis server nodes for Distributed Memory Processing (DMP). Distributed Processing is described in detail in separate documentation.

When the main product, *fe-safe* or *safe4fatigue*, is installed, on any platform, the following directory structure is created in the chosen installation directory (see section 3):



### `fe-safe\version.x.yy\data`

This directory contains sample data for use with tutorials.

### `fe-safe\version.x.yy\database`

This directory contains the *fe-safe* materials database and sample materials databases.

### `fe-safe\version.x.yy\documentation`

This directory contains *fe-safe* documentation in PDF format.

### `fe-safe\version.x.yy\exe`

This directory contains the main program executable files, libraries and the licence key file.

### `fe-safe\version.x.yy\external`

This directory contains third-party API library files required by some interface routines.

### `fe-safe\version.x.yy\kt`

This directory contains surface finish data files.

### `fesafe\version.x.yy\local`

This directory contains default local database files.

### 203.1.2 The *fe-safe* User Directory

An individual *fe-safe* User Directory is created for each user, the first time they run *fe-safe* when logged on with a particular login name. Each user directory contains configuration files and material database files pertaining to an individual login name as well as their projects directory. The location of the user directory is specified during installation and depends on the platform on which *fe-safe* is installed.

Linux: On Linux installations the default location of the user directory is:

```
{user's home directory}/fe-safe.version.x.yy
```

This method of configuration ensures that a user can use the same user directory regardless of where they are logged on, provided that they log on with the same user name. Disk space quotas and file and directory permissions for this directory are a property of the operating system, and are configured by the system administrator.

Windows: On Windows installations the default location of the user directory is the 'My Documents' or simply 'Documents' directory, e.g. on Windows 7 that would be:

```
C:\Users\{username}\Documents\fe-safe.version.x.yy
```

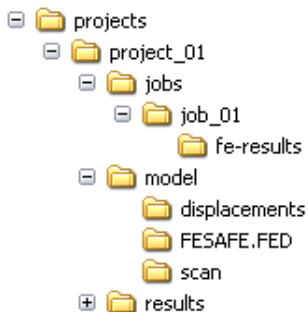
When a user directory is first created it contains:

- the user's local material database `local.dbase`
- the user's system settings files:

```
user.stli
gui.stli
```

### 203.1.3 The *fe-safe* Projects Directory

A *fe-safe* Project Directory is used each time *fe-safe* is run. If the directory does not exist the following structure is created in the chosen location (defaults to the User Directory: `<UserDir>/projects`):



The *fe-safe* Projects Directory is used to store the project configuration file, configuration files for the FEA fatigue analysis (`jobs` directory) and its results (`<JobsDir>/job_xx/fe-results` directory), together with the loaded FEA Models (`model` directory) and signal analysis results (`results` directory) to maintain a record of the entire analysis and to reference the files later.

The location of the **Project Directory** can be configured when starting *fe-safe*. The current project path can be displayed in the **Analysis Options** dialogue. This folder can be stored in any location, but it should be noted that it can become quite large. Since it contains the current working files it is recommended to have it on a local drive for best performance.

### 203.2 Settings

*fe-safe* and *safe4fatigue* use a system of settings to control program options.

The system settings are stored in an individual user's settings files, i.e. in the files:

```
<UserDir>\user.stli and
<UserDir>\gui.stli
```

The project settings are stored in the project settings file, i.e. in the files:

```
<ProjectDir>\project.stli and
<JobsDir>\job.stli
```

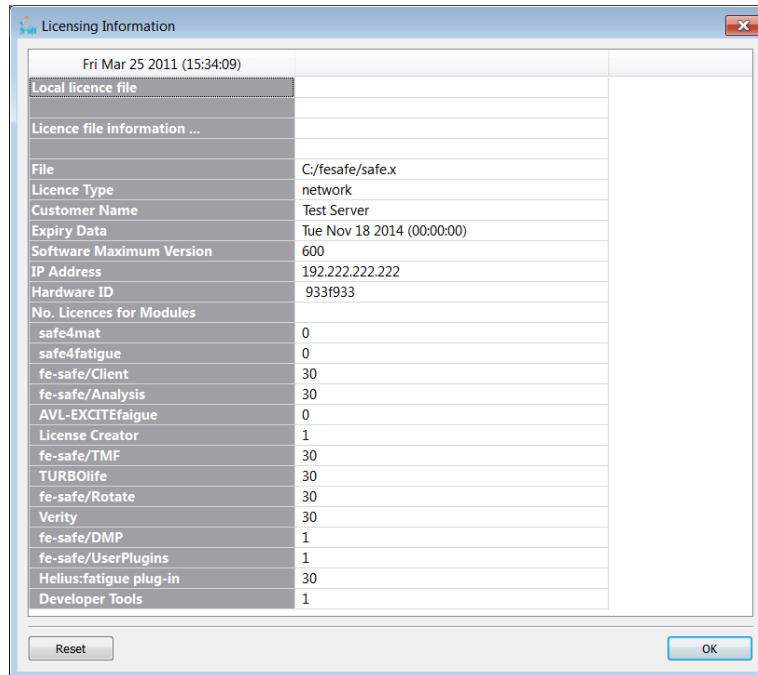
The same settings system is used when saving or retrieving fatigue analysis definitions, using the **Save FEA Fatigue Definition File** and **Open FEA Fatigue Definition File** options in the **Project** section of the **File** menu. Analysis definitions combine information from both the *project.stli* and *job.stli* files and are saved to project definition files with extension *.stlx*.

### 203.3 Licensing Information

A summary of licensing information can be obtained by selecting:

**Help >> Licensing Information**

which displays detailed version information about the current licence, as shown below:



## 204 Appendix D – Integrating fe-safe into a 3rd party product

### 204.1 Introduction

This appendix outlines the suggested technique for integrating a batch version of *fe-safe* into a 3<sup>rd</sup> party product. The technique assumes that the 3<sup>rd</sup> party product will read the FEA results and format them for *fe-safe* and it will extract the calculated fatigue contours from a .fer ASCII results file created by *fe-safe*. More detailed description of command line arguments and running *fe-safe* in batch mode can be found in section 23.

### 204.2 Reading FEA information

The 3<sup>rd</sup> party product should convert the FEA model to a .csv ASCII format for *fe-safe*. The syntax is outlined in Appendix E.

The .csv file should include stresses, strains (for elastic-plastic analysis), temperatures (for thermal analysis) and any group information. Group information allows different sections of a model to be analysed with different parameters – materials, surface finishes etc.

### 204.3 Analysis definition

The analysis is defined using a fe-safe Project Definition (\*.stlx) file. This is an ASCII file format and is outlined in Appendix E.

The loading is defined using a .ldf file. The format of this file is outlined in section 13. This is an ASCII file format.

Ensure that the .stlx file created references the .csv model storing the formatted FEA model, the loading definition file and also specifies the output file as a .fer file.

The rest of the analysis definitions can be specified at the time *fe-safe* runs in batch mode as command-line parameters and keywords. See section 23 and Appendix E.

### 204.4 Running fe-safe in batch mode

*fe-safe* can be operated from the command line, using batch and macro files as described in section 23. The command line options of interest are **j=c:\myfiles\mymodel.csv** and **b=c:\myfiles\def.stlx**.

By defining the **b=<stlxfile>** option *fe-safe* will perform a fatigue analysis defined by the project definition file <stlxfile>, and the **j=refresh** or **j=<csvfile>** option will either force fe-safe to refresh the FE model defined in the Project Definition (\*.stlx) file, or specify the path and filename for the particular analysis. The optional parameter **<kwd>=<value>** can be used to configure analysis options.

For example :

```
C:\SIMULIA\fe-safe\version.2016\exe\fesafe_cl.exe b=c:\myfiles\def.stlx j=c:\mymodel.csv
```

For a more detailed description of all command line options and parameters please refer to section 23.

### 204.5 Reading results

The format of the *fe-safe* FE fatigue results (FER) file (\*.fer) is outlined in Appendix E.





## 205 Appendix E - File Formats

### 205.1 Introduction

This appendix gives details of the various file formats used in *fe-safe*.

The file types and their extensions are discussed in the following sections. Using the recommended file extensions will ensure that *fe-safe* recognises the file correctly.

Binary and ASCII file types are supported, and the file types comprise:

- (i) Proprietary data file formats - see 205.2, below;
- (ii) Third-party data file formats - see 205.3, below;
- (iii) Proprietary FE file formats - see 205.4, below;
- (iv) Third-party FE file formats - see 205.5, below;
- (v) Third-party viewer formats - see 205.6, below;
- (vi) *fe-safe* user file formats - see 205.7, below;
- (vii) *fe-safe* working file formats - see 205.8, below.

#### 205.1.1 Customised and third-party interfaces to *fe-safe*

Dassault Systemes can supply information relating to the format and the use of the file types used as working files in *fe-safe*. Appendix E contains comprehensive information for many of the file formats used.

Where practicable, *fe-safe* endeavours to maintain consistency and backward-compatibility of these file types between versions. However, in future releases of the software, file formats may be subject to change or their use modified. This should be taken into account when creating interfaces between *fe-safe* (or files used by *fe-safe*) and third-party or in-house code. Please contact Simulia support if you develop an interface to an *fe-safe* working file, so that we can take this into consideration when future releases are planned.

Note that most ASCII formats used by *fe-safe* are token-based, so care should be taken not to rely on the layout of the file when interfacing to it.

#### 205.1.2 Binary file formats

##### **Portability between platforms:**

The binary DAC file is portable between platforms, i.e.

- a file created in Windows can be used on any of the supported Linux platforms;
- a file created on any of the supported Linux platforms can be used in Windows.

Other binary formats are not portable between platforms, i.e.

- a file created in Windows can only be used on a Windows or supported Linux platform.

Third party binary FEA files such as binary Abaqus \*.fil files are only portable between machines that have the same byte order.

### 205.1.3 ASCII file formats

#### Portability between platforms:

On Windows platforms a line in an ASCII file is terminated by a carriage return character followed by a line feed character. Windows applications will normally use these line termination characters.

On Linux platforms a line in an ASCII file is terminated by a line feed character only. Linux applications will normally use this line termination character.

An ASCII file can be transferred from a Windows machine to a Linux machine (or vice-versa) using an FTP utility. Most FTP utilities will automatically convert the line termination characters to the correct type. If a file is loaded from a tape or CD produced on a different platform, the line termination characters may be incorrect. This can usually be corrected by FTP'ing the file to the other platform and then FTP'ing it back again.

### 205.2 Proprietary data file formats

*fe-safe* and *safe4fatigue* support the following proprietary data file types:

- industry standard binary DAC file (\*.dac)
- analogue multi-channel AMC file (\*.amc)
- ASCII single and multi-channel data files (\*.txt, \*.asc, etc.)
- ASCII histogram files (\*.txt, \*.asc, etc.)

#### 205.2.1 Binary DAC file (\*.dac)

The DAC file is a single channel binary data file. This file type was original defined by Prosig Limited in the 1970's, and is supported by many signal processing suites. However, some third-party software packages use extensions or modifications to the original DAC format, so compatibility between suites cannot always be guaranteed.

The *fe-safe* implementation of the DAC file supports:

- single channel sequential data;
- xy data;
- xy data containing hysteresis loops;
- three-dimensional histogram data.

The file includes a header containing data type, signal and file information.

The data structure for this format is available on request.

The DAC file format is used as the default format for data files created in *fe-safe* when using signal generation and signal modification functions.

The default file extension for this file type is `.dac`. However, in some contexts, files in this format may have alternative file extensions. For example: the results of a "time at level" analysis are written to a file with extension `*.tal`, which uses the DAC single channel sequential data file format.

Files produced by *fe-safe* in DAC format can be converted to other data formats using the **File >> Data Files >> Save Data Files as...** option.

## 205.2.2 Analogue multi-channel AMC file (\*.amc)

The AMC file format is a very efficient proprietary binary data file format supporting single or multi-channel data. The file includes a header containing signal and file information.

The data file header record contains record pointers to each of the sub header records and to the raw data. This has the advantage that backward compatibility will be maintained if additional sub header records are added.

Each record is 512 bytes long.

The following standard records are documented below:

- header record
- general details record
- analogue channel details records
- event details records
- binary data record

### **Header record**

This record contains general file header information and pointers to each of the other records.

Item	Data type
Program which created data file	Char*12
Date of creation	3 Int*2
Number of records	Int*4
Pointer to General Details	Int*4
Pointer to Analogue Channel Details	Int*4
Unused pointer	Int*4
Pointer to Raw Data	Int*4
Pointer to Event information	Int*4
Pointer to Application specific recs	Int*4
Pointer to textual recs	Int*4
Free Pointers (13 of)	Int*4
Number of Analogue Channel records	Int*2
Number of Digital channel records	Int*2
Number of Event records	Int*2
Number of Application Specific records	Int*2
Number of text records	Int*2
Free record counters	15 Int*2
Version number of file	Int*2
The remainder of the record is empty for future use.	.....

Each pointer is a record pointer, i.e. the first 512 byte record is record 1, the second 512 bytes record is 2, etc..

The Event, Application and Textual records are optional and can be omitted if required, by setting the pointers to 0 and the number of records of this type to 0.

#### General details record

This record contains any information applied to all the data channels.

Item	Data Type
Sample Rate	Real*4
Number of Analogue Channels	Int*2
Number of Digital Channels	Int*2
Number of points per channel	Int*4
Bits per channel	Int*2
Start date	6 Int*2
Date type	Int*2
Number of Events	Int*2
Base time for data	Real*4
Textual Information (Last 128 bytes in record)	Char*128

The bits per channel can be 8,12,16 or 32, where

8 bit	:	1 byte storage integer	(0-255)
12 bit	:	2 byte storage integer	(0-4095)
16 bit	:	2 byte storage integer	(0-32767)
32 bit	:	4 byte storage float	

Date type:

1	:	UK	day/month/year
2	:	European and North American	month/day/year
3	:	Japanese	year/month/day

The remainder of the record is not used.

### Analogue channel details record

Each of these records defines each of the analogue channels. Only those defined appear in the raw data section.

Item	Data Type
Channel Number	Int*2
Channel Active	Int*2
Channel Name	Char*12
Channel Label	Char*10
Channel Units	Char*4
Maximum Scale	Real*4
Minimum Scale	Real*4
Max/Min set in next two values	Int*2
Maximum	Real*4
Minimum	Real*4
Angle type (0:Unknown,1:Rads,2:Degs)	Int*2
Channel Description (Last 128 bytes in record)	Char*128

### Event details record (optional)

The file can contain up to 12 event records. In each of these records 8 events can be stored, as follows:

Item	Data Type
Event position	8 Int*4
Event description	8 Char*40
Event descriptor	8 Char*10
Event value	8 Real*4

The event position is an offset of blocks of channels from the start of the binary data, the start being position 1.

The remainder of each record is not used.

**Textual records (optional)**

At present there are either no text records or three. The following information is stored in the three text records:

Item	Data Type
Number of ordinary text lines	Int*2
Extra details present	Int*2
Ordinary text lines	17 Char*60
Number of extra text lines	Int*2
Extra text lines	8 Char*60
Not used	28 Char*1

**Raw binary data**

This section contains all of the data relating to the channels defined in the channel details sections. The data is written in a multiplexed format for each of the active channels, as follows:

```
< scan #1 / channel #1 >
< scan #1 /      :      >
< scan #1 / channel #n >
< scan #2 / channel #1 >
< scan #2 /      :      >
< scan #2 / channel #n >
< scan #3 / channel #1 >
<      :      /      :      >
< scan #m / channel #n >
```

For example, for a three channel file:

```
< scan #1 / channel #1 >
< scan #1 / channel #2 >
< scan #1 / channel #3 >
< scan #2 / channel #1 >
< scan #2 / channel #2 >
< scan #2 / channel #3 >
< scan #3 / channel #1 >
< scan #3 / channel #2 >
< scan #3 / channel #3 >
<      :      /      :      >
<      :      /      :      >
<      :      /      :      >
etc.
```

Each sample is stored according to the bits per channel number in the general details record, where:

- 8 bit : 1 byte storage integer (0-255)
- 12 bit : 2 byte storage integer (0-4095)
- 16 bit : 2 byte storage integer (0-32767)
- 32 bit : 4 byte storage float

For 8, 12 and 16 bit data types, a data value, J, can be converted into an engineering value, J<sub>E</sub>, using the scale values extracted from the general details record, where:

$$J_E = (\text{Minimum Scale}) + J \times (\text{Maximum Scale} - \text{Minimum Scale})$$

### 205.2.3 ASCII single and multi-channel data files (\*.txt, \*.asc, etc.)

The ASCII data file format provides a straightforward method for inputting single or multi-channel x-y or time-y data. Data can be entered using any text editor. Alternatively, it can be exported from a spreadsheet application (for example Microsoft Excel) or a user's own program.

The format also provides a powerful generic solution for importing time history data from third-party data acquisition packages, almost all of which can export data in an ASCII format.

In its simplest form, an ASCII data file has no header, containing only columns of data. Alternatively, a header can be included containing signal information.

#### ASCII data file syntax

- (i) ASCII data files can either include a header or have no header. If the header does not conform to the Safe Technology ASCII header format, it will be skipped, and the number of columns in the file will be determined from the file format, if this is possible.
- (ii) Data can be comma, space or tab delimited. Multiple consecutive delimiting characters count as one character.
- (iii) Each line, including the last line, must be correctly terminated (see 205.1.3, above).
- (iv) The maximum line length that *fe-safe* can read is 8192 characters.
- (v) If no header is included then the file must have the file extension \*.txt or \*.asc.
- (vi) If a header is included, then the file can have any file extension. However, the default types \*.txt and \*.asc are most efficient.
- (vii) If a header is included, a blank line must be included between the header and the data, at line six of the file.
- (viii) If part of the header is required, then the whole header (the first six lines of the file) must be included. The sixth line of the header is a blank line between the header information and the data, and must always be included. The header must contain the following lines:

Line 1	-	the string: SafeTechnologyASCII
Line 2	-	the sample rate
Line 3	-	the x-axis label and units
Line 4	-	the channel names
Line 5	-	the y-axis label and units
Line 6	-	a blank line
Line 7 – end	-	comma, space or tab-delimited columns of data
	-	multiple consecutive delimiting characters count as one

Example 1 – Single channel ASCII data file with no header:

(E.2.3.1-1\_Sample\_ASCII\_data\_file.txt)

```
358.401
263.099
244.907
993.477
643.779
249.101
```

**Example 2 – Multi-channel ASCII data file with no header:**

(E.2.3.1-2\_Sample\_ASCII\_data\_file.txt)

```

358.401    219.987    22.5000    358.401
263.099   -1953.55   -39.2821   -1953.55
244.907   -1122.10   -42.3279   -1122.10
993.477    242.360    46.3464    993.477
643.779    192.140    64.7305    643.779
249.101   -871.678   -48.6921   -871.678

```

**Example 3 – Single channel ASCII data file with header:**

(E.2.3.1-3\_Sample\_ASCII\_data\_file.txt)

```

SafeTechnologyASCII
100
Time:Secs
RosetteStrain
Strain:uE

358.401
263.099
244.907
993.477
643.779
249.101

```

**Example 4 – Multi-channel ASCII data file with header:**

(E.2.3.1-4\_Sample\_ASCII\_data\_file.txt)

```

SafeTechnologyASCII
100
Time:Secs
RosetteStrain1 RosetteStrain2 RosetteStrain3 RosetteStrain4
Strain:uE      Strain:uE      Strain:uE      Strain:uE

358.401        219.987        22.5000        358.401
263.099       -1953.55       -39.2821       -1953.55
244.907       -1122.10       -42.3279       -1122.10
993.477        242.360        46.3464        993.477
643.779        192.140        64.7305        643.779
249.101       -871.678       -48.6921       -871.678

```

**205.2.4 ASCII histogram files (\*.txt, \*.asc, etc.)**

The ASCII histogram file format provides a straightforward method for inputting range-mean histogram data. Data can be entered using any text editor. Alternatively, it can be exported from a spreadsheet application (for example Microsoft Excel) or a user's own program.

Unlike the ASCII data file, an ASCII histogram file must always include a header.

**ASCII histogram file syntax**

- (i) ASCII histogram files must always include a header.
- (ii) Data can be comma, space or tab delimited. Multiple consecutive delimiting characters count as one character.
- (iii) Each line, including the last line, must be correctly terminated (see 205.1.2.1, above).
- (iv) A value must be entered for every range-mean pair.
- (v) The file can have any file extension. However, the default types \*.txt and \*.asc are most efficient.



- (ix) The file must contain the following lines:
- Line 1 - the string: `SafeTechnologyASCII`
  - Line 2 - `-1` (must be `-1` to indicate that this is a histogram file)
  - Line 3 - channel name
  - Line 4 - X name, Y name, Z name
  - Line 5 - a blank line
  - Line 6 - either a blank line, or the string: `LOWER`
  - a blank line implies the bin values are mid values; `LOWER` indicates they are the lower edge of the bin
  - Line 7 - list of bin edges for Y dimension (MEAN)
  - Line 8 – end - first column is X bin edge value
  - remaining columns are counts for each Y bin
  - columns are comma, space or tab-delimited
  - multiple consecutive delimiting characters count as one
- (ix) Bins MUST be evenly spaced in both directions.

Example – ASCII histogram file:

(E.2.4.1\_Sample\_ASCII\_histogram.txt)

```
SafeTechnologyASCII
-1
Rainflow
Range:y Mean:y Cycles

LOWER
-1800.0 -1400.0 -1000.0 -600.0 -200.0 200.0 600.0 1000.0 1400.0
10.0 1 2 5 11 23 11 5 2 1
20.0 0 0 1 4 15 4 1 0 0
30.0 0 0 0 1 6 1 0 0 0
40.0 0 0 0 0 2 0 0 0 0
50.0 0 0 0 0 1 0 0 0 0
```

### 205.3 Third-party data file formats

*fe-safe* and *safe4fatigue* support the following third-party data file types:

- Servotest SBF and SBR files (\*.sbf, \*.sbr)
- Snap-Master file (\*.sm?)
- MTS RPCIII binary data file (\*.rsp)
- Adams multi-column ASCII tabular data (\*.tab)
- ANSYS Modal Coordinates File (\*.mcf)
- ASAM MDF4 binary data file (\*.mf4)

The interfaces to these file formats operate without conversion. In other words, no translation is required, *fe-safe* works directly from the data file.

*fe-safe* endeavours to maintain interface support to the latest versions of supported third-party data files.

These file formats are discussed in Appendix F – “Interfacing to third-party products – data files”.

## 205.4 Proprietary FE file formats

*fe-safe* can import Finite Element analysis data (i.e. stresses, strains and temperatures) from the following proprietary FE file types:

- ASCII FE tensor file (\*.csv, \*.txt, \*.asc)
- *fe-safe* FE data (FED) folder (\*.fed)

### 205.4.1 ASCII FE tensor file

The ASCII FE data file facilitates analysis of FE results from any FE source. Data can be entered using any text editor. Alternatively it can be exported from a spreadsheet application (for example Microsoft Excel), an FE package or a user's own program.

The file contains stress and/or strain tensor data for elements or nodes. Multiple datasets, multiple nodes (per element), shells and mesh topology are supported.

#### ASCII FE data file syntax

- (i) Data can be comma, space or tab delimited. Multiple consecutive delimiting characters count as one character.
- (ii) Data should start at the beginning of each line, i.e. avoid beginning a line with a delimiting character.
- (iii) Each line, including the last line, must be correctly terminated (see 205.1.2.1, above).
- (iv) Blank lines are ignored.
- (v) Lines beginning with a # (hash) character are treated as comment lines.
- (vi) The file extension can be \*.txt, \*.csv or \*.asc.
- (vii) The capabilities of the file can be extended by specifying:

#### FORMAT 1

- This allows the support of parts and instances

See below for where alternate formats are used

- (viii) Instance definition (FORMAT 1 only):

To reduce file size a reference can be made between instance names and a number using the format:

INSTANCE <Instance\_no> "<Instance\_name>"

e.g. Specifying the instance

INSTANCE 1 "REAR\_COIL\_OFFSIDE"

Allows ["REAR\_COIL\_OFFSIDE"]1234 to be replaced with [1]1234

Note: The instance numbers used within the file are no guarantee that *fe-safe* will use the same instance numbering within *fe-safe* analysis logs etc.

- (ix) Tensor definition:

*Elemental* tensor data is expressed in the format:

<El\_no> [<Shell\_no>] <Node\_no> <Sxx> <Syy> <Szz> <Sxy> <Syz> <Sxz> <Temp>

where <shell\_no> is an optional parameter and can be omitted or substituted with the value -1.

*Nodal* tensor data is expressed in the format:

<Node\_no> [<Shell\_no>] <Instance\_of\_node> <Sxx> <Syy> <Szz> <Sxy> <Syz> <Sxz> <Temp>

where `<Instance_of_node>` is an incremental value, since there may be more than value for a node, for example where the node is shared by two elements of different types or different materials.

FORMAT 1:

The item ID is in fe-safe standard form e.g.

<code>&lt;El_no&gt;.&lt;El_node&gt;</code>	An element item with a mandatory sub-node specified
<code>&lt;Node_no&gt;</code>	A node item
<code>&lt;El_no&gt;.&lt;El_node&gt;:&lt;Shell_no&gt;</code>	Optional shell by appending <code>:&lt;Shell_no&gt;</code>
<code>[“&lt;Instance_name&gt;”]&lt;Node_no&gt;</code>	Optional part instance using instance name
<code>[&lt;Instance_no&gt;]&lt;Node_no&gt;</code>	Optional part instance using instance ID, see INSTANCE above

Elements and nodes can have both shell and an instance specified.

Tensor data is expressed in the format:

`<Item> <Sxx> <Syy> <Szz> <Sxy> <Syz> <Sxz>`

Where temperature is allowed the format is:

`<Item> <Sxx> <Syy> <Szz> <Sxy> <Syz> <Sxz> <Temp>`

For non-tensor data the format includes the minimal data, e.g. temperature only is:

`<Item> <Temp>`

- (x) The beginning and end of the datasets may be specified using the keywords `DATASET` and `END`, as follows:

```
DATASET <name_of_dataset>, src=<sourcefile_path+filename>, time=<time>, step=<step>, inc=<increment>,
      position=<data_position>, type=<data_type>
:
:
<Tensor definition>
:
:
END
```

Where a dataset header is used, the name of the dataset must be specified. The other parameters are optional.

- (xi) The `DATASET` parameters have the following meanings:

Parameter	Value
<code>src</code>	The path and filename of the FE model file.
<code>time</code>	FE solution time.
<code>freq</code>	The frequency of a dynamic FE solution.
<code>step</code>	FE solution step number.

inc	FE solution increment number.
pos	<p>Data position. Possible values are:</p> <p>ELEMENT {default} - Data at nodes on elements.</p> <p>NODE - Data at nodes.</p> <p>INTEGRATION - Data at integration points.</p> <p>CENTROID - Data at the element centroid.</p> <p>ELANDCENT - Data at the nodes and centroid of an element.</p>
type	<p>Data type. Possible values are:</p> <p>STRESS {default} - Six-component stress tensor plus temperature value.</p> <p>STRESS-ONLY - Six-component stress tensor (*see Note 1).</p> <p>STRAIN - Six-component strain tensor plus temperature value.</p> <p>STRAIN-ONLY - Six-component strain tensor (*see Note 1).</p> <p>IMAG-STRESS - Imaginary part of six-component stress tensor plus temperature value.</p> <p>IMAG-STRESS-ONLY - Imaginary part of six-component stress tensor (*see Note 1).</p> <p>REAL-STRESS - Real part of six-component stress tensor plus temperature value.</p> <p>REAL-STRESS-ONLY - Real part of six-component stress tensor (*see Note 1).</p> <p>TEMPERATURES-ONLY - Temperature value (*see Note 2).</p> <p>VERITY-FORCES - 3 membrane and 3 bending force components (*see Note 1).</p> <p>THICKNESS - Shell thickness value (FORMAT 1 only)</p> <p>Custom Scalar:&lt;Var_name&gt; - 1 value (FORMAT 1 only)</p> <p>Custom Vector:&lt;Var_name&gt; - 3 value (FORMAT 1 only)</p> <p>Custom Tensor:&lt;Var_name&gt; - 6 values (FORMAT 1 only)</p>

\* Note 1: Regardless of the data type defined in the DATASET header, the tensor definition always has the format described in (vii), above. All tensor definition fields must be present. For data types that include a temperature, the last field is used to store the temperature value. Prior to FORMAT 1, if

the temperature is defined in a separate dataset, then the temperature value for the last field in each tensor definition will be ignored.

\* Note 2: A “TEMPERATURES-ONLY” dataset has the format described in (vii), above. Prior to FORMAT 1 however, the six tensor values (Sxx, Syy, etc.) are ignored.

\* Note 3: Imaginary and real datasets will be paired in the order in which they appear in the file.

(xii) The keyword `NOHEADERS` can be used (instead of the keywords `DATASET` and `END`) where all tensors relate to the same node and each tensor is to be treated as a separate dataset; This is not support with FORMAT 1

(xiii) The maximum number of datasets that can be read into *fe-safe* is currently limited to 256000. (Users wishing to exceed this limit should contact their local support office).

(xiv) Node co-ordinates are specified in a `CO-ORDS` block which has the format:

```
CO-ORDS
<Node_num> <x> <y> <z>
:
<Node_num> <x> <y> <z>
END
```

When using instances, prefix the `<Node_num>` with [`<Instance>`]

(xv) The format of element-nodes list is:

```
ELEMENTS
<Element_num> <Node_1_num> <Node_2_num> <Node_3_num> ...
:
<Element_num> <Node_1_num> <Node_2_num> <Node_3_num> ...
END
```

When using instances, prefix the `<Element_num>` and `<Node_?_num>` with [`<Instance>`]

(xvi) Each element can have a different number of nodes specified.

(xvii) The element nodes list may optionally specify the element and the element topology:

```
<Element_num> <Node_1_num> <Node_2_num> (<Element_type>)
<Element_num> <Node_1_num> <Node_2_num> (<Element_type>:<Element_topology>)
<Element_num> <Node_1_num> <Node_2_num> (:<Element_topology>)
```

(xviii) Accepted element types are:

Element Type Value	Element Description
LSHELLD / shell-tri	Linear triangular element
QSHELLD / shell-tri	Quadratic triangular element
LSHELL / shell	Linear quadralateral element
QSHELL / shell	Quadratic quadralateral element
LTET / tet	Linear tetrahedral element
QTET / tet	Quadratic tetrahedral element
LPYR / pyr	Linear pyramid element
QPYR / pyr	Quadratic pyramid element
LWDG / wdg	Linear wedge element
QWDG / wdg	Quadratic wedge element
LHEX / hex	Linear brick element
QHEX / hex	Quadratic brick element
UNCLASSIFIED	Default value, element has no type
oct	Octahedral element
beam	Beam element
conn	Ridgid beam elements

(xix) Accepted element topologies are:

Element Topology Value	Topology Description
NORMAL	Default value, element topology identical to Abaqus odb.
TYPE2	Element topology identical to I-Deas unv.
TYPE3	Element topology identical to Nastran.
UNKNOWN	Element topology is unknow and will not be used.

## Examples

To see an example of the file format for a particular data type, load a model of that type in fe-safe from another file format (for example FIL, ODB, RST, etc.), then save the file as an ASCII file, using:

**FILE >> FEA Solutions >> Save Loaded FE Models... >> Save as type: Text file (.txt)**

The file can be saved with extension: \*.txt, \*.csv or \*.asc.

Example 1 – ASCII FE tensor file – multiple datasets; multiple nodes; including shells; geometry:

(E.4.1.2-1\_Sample\_ASCII\_tensor\_file.txt)

```
# sample_ASCII_tensor_file_01
# ~~~~~
# - multiple datasets
# - multiple nodes
# - multiple groups

GROUP ALL
1
17
END

GROUP JUST1
1
END

#El_no Shell_no Node_no Sxx   Syx   Syy   Szz   Sxy   Syz   Sxz   Temp

DATASET sample_dataset_01
1      1      1      0      0      0      0      0      0      0
17     1      3      0      0      0      0      0      0      0
1      5      1      0      0      0      0      0      0      0
17     5      3      0      0      0      0      0      0      0
END

DATASET sample_dataset_02
1      1      1      1.42e2 3.33e4 0      2.1e-12 0      0      0
17     1      3      5.60e1 3.33e4 0      1.1e-12 0      0      0
1      5      1      1.42e2 3.33e4 0      2.1e-12 0      0      0
17     5      3      5.60e1 3.33e4 0      1.1e-12 0      0      0
END

DATASET sample_dataset_03
1      1      1      1.71e2 -1.12e1 0      -5.6e-11 0      0      0
17     1      3      -3.13e1 -3.13e1 0      -1.4e-11 0      0      0
1      5      1      -1.71e2 1.12e1 0      5.6e-11 0      0      0
17     5      3      3.13e1 3.13e1 0      1.4e-11 0      0      0
END

CO-ORDS
1      0      0      0
2      0      0.5  0
3      0.5  0      0
4      0.5  0.5  0
5      1      0      0
6      1      0.5  0
END

ELEMENTS
1      1      2      4      3      (LSHELL)
17     3      4      6      5      (LSHELL)
END
```

Example 2 – ASCII FE tensor file – multiple datasets; single node; no headers:

(E.4.1.2-2\_Sample\_ASCII\_tensor\_file.txt)

```
# sample_ASCII_tensor_file_02
# ~~~~~
# - multiple datasets
# - single node

#El_no Node_no Sxx      Syy      Szz      Sxy      Syz      Sxz      Temp

NOHEADERS
29      1          0          0          0          0          0          0          0
29      1      1.42e2  3.33e4  0          2.1e-12  0          0          0
29      1      5.60e1  3.33e4  0          1.1e-12  0          0          0
29      1      1.42e2  3.33e4  0          2.1e-12  0          0          0
29      1      5.60e1  3.33e4  0          1.1e-12  0          0          0
29      1          0          0          0          0          0          0          0
29      1      1.71e2  -1.12e1 0          -5.6e-11 0          0          0
29      1     -3.12e1 -3.12e1 0          -1.4e-11 0          0          0
29      1     -1.71e2 1.12e1  0          5.6e-11  0          0          0
29      1      3.11e1 -3.12e1 0          1.4e-11  0          0          0
29      1          0          0          0          0          0          0          0
```

Example 3 – ASCII FE tensor file – multiple datasets; multiple nodes; including shells:

(E.4.1.2-3\_Sample\_ASCII\_tensor\_file.txt)

```
# sample_ASCII_tensor_file_03
# ~~~~~
# - multiple datasets
# - multiple nodes
# - including shells

#El_no Sh_no  Node_no Sxx      Syy      Szz      Sxy      Syz      Sxz      Temp

DATASET sample_dataset_01
1       1       1         0          0          0          0          0          0          0
17      1       3         0          0          0          0          0          0          0
1       5       1         0          0          0          0          0          0          0
17      5       3         0          0          0          0          0          0          0
END

DATASET sample_dataset_02
1       1       1      1.42e2  3.33e4  0          2.1e-12  0          0          0
17      1       3      5.60e1  3.33e4  0          1.1e-12  0          0          0
1       5       1      1.42e2  3.33e4  0          2.1e-12  0          0          0
17      5       3      5.60e1  3.33e4  0          1.1e-12  0          0          0
END

DATASET sample_dataset_03
1       1       1      1.71e2 -1.12e1 0          -5.6e-11 0          0          0
17      1       3     -3.12e1 -3.12e1 0          -1.4e-11 0          0          0
1       5       1     -1.71e2 1.12e1  0          5.6e-11  0          0          0
17      5       3      3.11e1 -3.12e1 0          1.4e-11  0          0          0
END
```



### 205.4.2 User Defined ASCII Group Files

These files allow groups to be added to the **Current FE Models** window for use in the analysis. There are two versions of this file.

#### Single group file.

The single group file is a comma, space, tab or new line-separated list of element or node IDs, defined either as single entries or ranges, for example :

```
969, 981, 990, 1051-1055, 1121-1131(2)
```

Name of the group will be automatically generated from the name of the file.

#### Multiple groups file.

The multiple group file contains a series of GROUP and END tokens with a list of element or node IDs between them. This is the same as the GROUP section of the ASCII FE tensor file outlined in section 205.4.1. For example :

```
GROUP Thin_Shaft
1060-1065 1057 1059 1068 1073 1075
END

GROUP Thin_Shaft_Near
1,5,6,25,39,190-195
END

GROUP Notch
892
893
END
```

### 205.4.3 *fe-safe* Finite Element Data (FED) folder (.fed)

The FED (.fed) folder contains a number of proprietary indexed binary format files, used internally by *fe-safe* for performing fatigue analyses from FE models.

The FED folder is used to store stress, strain, temperature and other information extracted from one of the supported FE file formats discussed in section 205.5, below. It can also be used as an efficient way to save and retrieve FE model data for re-use in *fe-safe*. This is particularly useful if a model is to be analysed more than once, or if the original model files are large.

The format of the FED folder files, and any associated documentation is owned by Dassault Systemes UK Ltd and is protected by United Kingdom copyright laws and international treaty provisions. You may not reverse engineer, decompile, or disassemble the FED folder files format.

When the model is imported, using the **Open Finite Element Model...** option, pertinent data is extracted from the model and is written to the FED folder. The folder is located in the model subdirectory of the current project directory, with a name FESAFE.FED.

Additional datasets, (for example additional stress datasets, strain datasets or a temperature dataset), can be appended to the FESAFE.FED folder, using the **Append Finite Element Model...** option. Appended datasets can also be imported from a file having a different file format, providing that the data relates to the same model, and the element and node numbers correspond. For example, note that Abaqus and I-DEAS use incompatible node numbering for some element types.

Element/node group information is loaded from the first file only (i.e. the file opened using the **Open Finite Element Model...** option).

The **Save Loaded FE Models...** option can be used to save the current FESAFE.FED folder to a different location. A saved FED folder can be retrieved later, in the same way as other FE model files, using the **Open Finite Element Model...** option - see section 5.

Note that when a FED folder is opened using the **Open Finite Element Model...** option, the contents of the file are used directly, without extracting data to a new FESAFE.FED folder. This can save read-in time.

## 205.5 Third-party FE file formats

*fe-safe* can read Finite Element analysis data (i.e. stresses, strains and temperatures) from the following third-party file types:

- Abaqus FIL results file (\*.fil)
- Abaqus output database ODB file (\*.odb)
- ANSYS RST results file (\*.rst)
- Nastran F06 print file (\*.f06)
- Nastran OP2 output file (\*.op2)
- Pro/Engineer stress and strain results files (\*.s01, \*.s02, etc.)
- Pro/Engineer temperature results files (\*.d01)
- SDRC I-DEAS UNV universal file (\*.unv)

*fe-safe* endeavours to maintain interface support to the latest versions of supported third-party FE packages. Details of file versions supported are given in the relevant sections of Appendix G.

Additionally, *fe-safe/Rotate* uses geometry information. Currently, the ANSYS results (RST), the Abaqus results (FIL) and ASCII model file formats are the only file types supported by the *fe-safe/Rotate* module – see section 21.

## 205.6 Third-party viewer formats

*fe-safe* is a dedicated fatigue analysis package used as part of a Finite Element design process, and as such, it does not include its own integrated FE viewer. The viewer or viewers used for the fatigue results will depend on the third-party FE packages being used. Some FE packages can be used to view the FE and fatigue results, whilst some require a separate viewer. *fe-safe* endeavours to maintain interface support for the latest versions of supported third-party viewers.

Commonly used viewers include:

- Abaqus Viewer (see Appendix H.)
  - can be used to view fatigue results from ODB files (\*.odb).
- ANSYS (see Appendix H.)

- can be used to view ANSYS RST results files (\*.rst).
- FEMAP / MSC/Nastran for Windows (see Appendix H.)
  - the FE model can be imported from a Nastran model file (\*.dat, \*.mod, \*.neu);
  - fatigue results can be exported from *fe-safe* as an ASCII CSV file (\*.txt, \*.csv, \*.asc) and superimposed onto the imported FE model.
- MSC/Patran (see Appendix H.)
  - the FE model can be imported from a Nastran model file (\*.dat, \*.mod, \*.neu);
  - fatigue results can be exported from *fe-safe* as an ASCII CSV file (\*.txt, \*.csv, \*.asc) and superimposed onto the imported FE model.
- Pro/Engineer (see Appendix H.)
  - can be used to view Pro/Engineer results files (\*.s0?).
- SDRC I-DEAS (see Appendix H.)
  - can be used to view I-DEAS universal results files (\*.unv).
- CADFIX / FAM4 (see Appendix H.)
  - can be used to view results from an Abaqus FIL file.
- FEMGV (see Appendix H.)
  - can be used to view results from an Abaqus FIL file.

#### 205.6.1 FEMAP ASCII results file (\*.csv, \*.txt, \*.asc)

Support for FEMAP ASCII results files is provided through the *fe-safe* ASCII FE tensor file format – see section 205.4.1, above, and Appendix H.

### 205.7 *fe-safe* user file formats

#### 205.7.1 Load definition (LDF) file (\*.ldf)

##### **Supporting legacy file definition formats.**

*From version 5.00, onwards*

The LDF file has replaced the block loading (SPC) format and the data set sequence (LCD) file format. From version 5.00, onwards, support for the LCD and SPC file formats is disabled. New users should always use the LDF file.

*From version 5.2, onwards*

*fe-safe* v5.2-00 saw the introduction of an enhanced GUI-based method for defining loading. Underlying the GUI method is the existing LDF file format, and an LDF file called “`current.ldf`” is maintained in the user directory.

Existing users can continue to edit LDF files using a text editor. However, it is anticipated that all new users and most existing users will use the GUI-based method for defining the loading.

##### **LDF file format**

The format of the LDF file is fully covered in section 13 of the user manual.

**HLDF file format**

The optional high-level loading definition HLDF file is described section 13 of the user manual.

**205.7.2 User-defined mean-stress correction definition (MSC) file (\*.msc)**

The user defined mean stress correction (MSC) function is used to define a set of correction factors as a function of the mean stress of a cycle, (similar to using a Goodman diagram). It is also used to define an infinite life envelope for FRF factor calculations, Haigh diagrams and Smith diagrams.

For a mean stress correction the envelope is used to evaluate the factor to apply to the cycles amplitude due to the mean stress before a life calculation is performed.

For FRF calculations the ratio of the distance to the envelope from the origin divided by the distance from the cycle to the envelope provides the safety factors.

Two methods for defining the envelope are supported. As a Haigh type diagram (Sa vs. Sm) or as a Smith type diagram (S vs. Sm).

For both methods the mean stress axis is made non-dimensional by dividing by the material ultimate tensile strength, UTS, (or the ultimate compressive strength, UCS, in compression).

Both methods have the same general formatting rules outlined below:

The MSC curve is defined in an MSC file. The format of the MSC definition file is the same as the format of the FRF definition file (see 205.7.3) and is documented below:

- The MSC and FRF files are ASCII text files.
- The token SMITH on a line by itself indicates that this is the Smith method of definition, otherwise the Haigh method is assumed.
- Lines with a # character in the first column are treated as comments.
- The first line is the temperature or a list of temperatures at which the MSC or FRF envelopes are defined. If the MSC or FRF is a function of temperature then a series of envelopes can be defined for different temperatures. If only one set is defined a temperature must still be defined. A maximum of 100 temperatures can be defined.
- The subsequent lines have the UTS or UCS fraction in column 1 then the MSC for each of the temperatures in the following columns. A maximum of 100 UTS fractions can be used to define the MSC. All negative fractions are assumed to be compressive and all positive fractions are assumed to be tensile.
- Since negative values of the UTS fraction are assumed to be compressive, they use the UCS rather than the UTS to evaluate the actual stress values.
- Each item is separated from the previous item by a space, comma or a tab. Multiple spaces and tabs are allowed.
- Empty lines are ignored.
- The last line in the file must end with a termination character (see general note regarding ASCII files in 205.1.3).

- The UTS fractions must decrease from top to bottom, the temperatures must increase from left to right, i.e.

>> Temperatures rise >>

↓ ↓ <b style="color: red;">UTS decreases</b> ↓ ↓	UTS_Fract1 UTS_Fract2 UTS_Fract3 UTS_Fract4	temp1 msc11 msc21 msc31 msc41	temp2 msc12 msc22 msc32 msc42	temp3 msc13 msc23 msc33 msc43
--	--	---	---	---

**Haigh type diagram (Sa versus Sm)**

The Haigh type diagram allows the mean stress correction or infinite life envelope to be defined as a function of the cycle amplitude and mean. The full envelope should be defined.

The vertical axis is made non-dimensional, by expressing the stress amplitude,  $S_a$ , as a ratio:

$$\frac{S_a}{S_{a0}} \quad (= \text{mean-stress correction factor})$$

Where  $S_{a0}$  is the stress amplitude at zero mean stress.

This ratio has a maximum value of 1.0 at a mean stress of zero.

At a mean stress equal to the material UTS, the allowable stress amplitude is zero, as the material is on the point of fracture. The mean stress axis therefore has a value of 1.0 at  $S_a = 0$ .

So, for a cycle (Sa, Sm) the value of the MSC factor is extracted for Sm and the equivalent zero mean stress is:

$$S_{a0} = \frac{S_a}{MSC}$$

or, if the fatigue algorithm uses strain amplitudes then:

$$e_{a0} = \frac{e_a}{MSC}$$

For example, an MSC allowing much higher cycles for compressive stresses is defined in the table below, resulting in the MSC in figure 205.7.2-1:

Sm/UTS	Sa/Sa0
1	0
0.8	0.05
0.6	0.15
0.4	0.35
0.2	0.6
0	1
Sm/UCS	
-0.2	1.6
-0.4	2.4
-0.5	2.5
-0.6	2
-0.8	0.8
-1.0	0

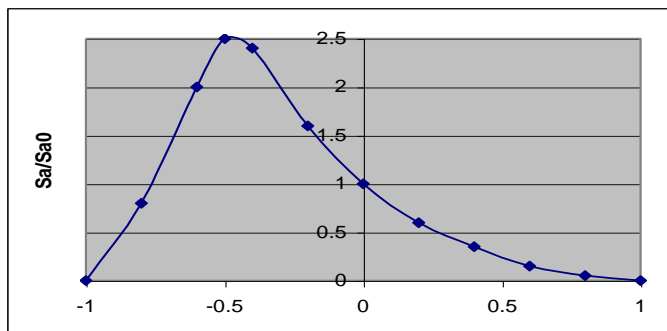


Figure 205.7.2-1

A sample file for multiple temperatures would be:

```
# Sample file
#      T1      T2      T3
      0      100     200

1      0      0      0
0.5    0.5    0.5    0.5
0      1      1      1
-0.25  1.3    1.4    1.5
-0.5    1.8    1.9    2.0
-0.75  0.7    0.75   0.8
-1     0      0      0
```

### Smith type diagrams.

The Smith type diagram allows the mean stress correction or infinite life envelope to be defined as a function of the cycle turning point stresses and the mean. Only the top section of the envelope should be defined. The lower section will be a mirror of the top section in the diagonal line joining the origin and the maximum mean stress point. The required section is shown for a Gerber envelope as a solid line in figure 205.7.3-1.

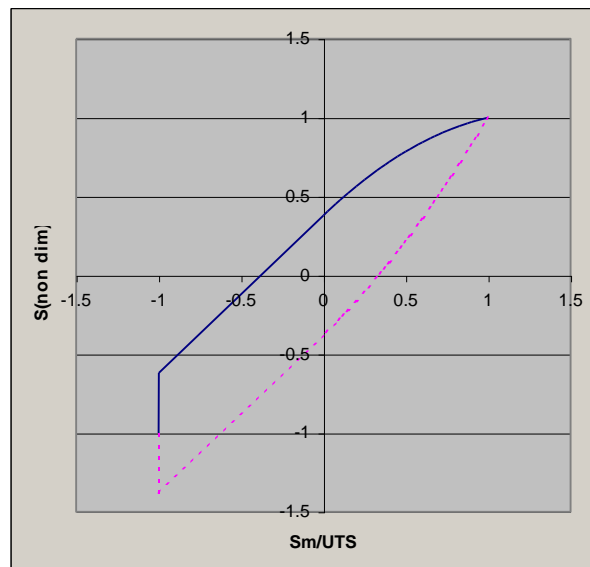


Figure 205.7-3.1

The stress values can be made non-dimensional by the maximum mean stress as shown above or so that the stress at zero mean is one. Within the code Smith diagrams are converted to Haigh diagrams prior to the analysis being performed, the factors will be corrected to ensure that the stress amplitude at zero means is one. The format of the Smith version of this file is identical to the Haigh version, the token SMITH must however be defined to inform the code that the file contains a Smith diagram.

The example below shows the Smith (LHS) and Haigh (RHS) versions for the Gerber MSC (the number of entries has been reduced to fit onto the page, the full versions are included in the database directory of your installation):

SMITH	
	0
1	1
0.9	0.972187
0.775	0.926734
0.7	0.893764
0.6	0.843155
0.5	0.784947
0.4	0.719141
0.325	0.6648
0.2	0.564732
0.1	0.47613
0	0.37993
-0.999	-0.61907
-1	-1

	0
1	0
0.9	0.19
0.775	0.399375
0.7	0.51
0.6	0.64
0.5	0.75
0.4	0.84
0.325	0.894375
0.2	0.96
0.1	0.99
0	1
-0.999	1
-1	0

### 205.7.3 Fatigue reliability factor envelope definition (FRF) file (\*.frf)

The FRF file uses exactly the same format as the MSC file – see 205.7.2.

### 205.7.4 SN data knock-down curve (KD) file (\*.kd)

A series of additional SN data scale factors in form of a knock-down curve can be applied to scale all stress data points in the defined material SN curve. The knock-down curve is defined in an KD file, with the following format:

- The KD file is an ASCII text files.
- Lines with a # character in the first column are treated as comments.
- Empty lines are ignored.
- The data lines have the Life (Nf) data followed by the scale factor. A maximum of 64 data pairs can be defined.
- The Life (Nf) values must increase from top to bottom.
- Negative values of either parameter are not allowed.
- Each item is separated from the previous item by a space, comma or a tab. Multiple spaces and tabs are allowed.
- At least 1 pair of values must be specified.
- The last line in the file must end with a termination character (see general note regarding ASCII files in 205.1.3).

A sample KD file would be:

```
# Sample file
#

100      1
1000     0.8
1e5      0.5
1e7      0.3
```

## 205.8 *fe-safe* working file formats

### 205.8.1 *fe-safe* settings files

*fe-safe* and *safe4fatigue* use a system of settings files to control program options. These files are individual to each user and are stored in the user's home directory.

The system settings are stored in an individual user's settings files, i.e. in the files:

```
<UserDir>\user.stli  and
<UserDir>\gui.stli
```

These files contain the system settings of *fe-safe* such as material database paths.

The project settings are stored in the project settings file, i.e. in the files:

```
<ProjectDir>\project.stli  and
<JobsDir>\job.stli
```

These files contain the project and job settings required to run the analysis.

The same settings system is used when saving or retrieving fatigue analysis definitions, using the **Save FEA Fatigue Definition File** and **Open FEA Fatigue Definition File** options in the **Project** section of the **File** menu. Analysis definitions combine information from both the `project.stli` and `job.stli` files and are saved to project definition files with extension `.stlx`.

### 205.8.2 *fe-safe* legacy keyword files, \*.kwd

#### Use of keywords and keyword files

Keyword files were used in *fe-safe* prior to version 6.00 to control program options and settings, from version 6.00 onwards they were replaced by settings files, see section 205.8.1 above.

The legacy keyword files are still supported and can be used to load a fatigue analysis definition, and the keywords can still be used as command line parameters for running the analysis from a macro or a command line, see section 23.

#### General format

Keyword files are ASCII text files.



Each line in the file has the format

```
123456789nnKey=Value
```

The first 9 characters on each line are ignored.

The next two characters *nn* define the group index to which the parameters belong. If blank this applies to the whole analysis. Some parameters such as the material properties and Kt can vary from group to group.

Group indices start at 01. Each group index has a special keyword `GroupNum` which defines the number of the group in the loaded model that this group index relates to, when the `GroupNum` parameter is set to -1, the previous group index was the last one.

Comments can be added into the keyword file. On comment lines column 10 must be the character # and the last character must be =.

For example:

```
123456789# This line indicates the columns used =
      # This line is a comment =
      APPENDTOFED=0
      CURRENTGROUP=AL
      FEDNAME=/fesafe.work/dump/FESAFE.FED
      FINALOPFIL=/fesafe.work/dump/keyholeResults.odt
      UNVIPFIL=/fesafe/version.3.10/fegui/input.unv
      UNVOPFIL=/fesafe.work/dump/keyholeResults.odt
      # Group 1 Parameters =
.....01GROUPNUM=7
.....01MATLNAME=SAE1020
.....01STRESSCON=1.1
      # Group 2 Parameters =
.....02GROUPNUM=4
.....02MATLNAME=SAE1040
.....02STRESSCON=1.2
123456789# End of file =
```

The following table shows a list of keywords:

Appendix E

205.8.3 Table of settings (by category)

<u>Syntax</u>	<u>Datatype</u>	<u>Default</u>	<u>Display Text</u>	<u>Comments</u>	<u>Legacy Keyword</u>
<b>Project Settings</b>	-	-	-	-	-
<b>File references</b>					
[project.job.loading file]	File	N/A	Loading file	The name of the load definition file if the loading mode is LDF	LDFFILE
[project.job.output file]	File	N/A	Contour destination file	The destination file for the results contours. Can be a .fer name.	FINALOPFIL
[project.model.mesh source file]	File	N/A	Mesh source FE model	This is the name of the FE model with the mesh	
[project.model.source file]	File	N/A	Source FE model	This is the name of the first opened FE model - it will never be a FED name	SOURCE_FEMODEL
[project.model.appended files(n)]	File	N/A	Appended FE models	The list of additional models appended to the first model - these will never be FED names	
[project.job.material databases.nodal properties]	File	N/A	Nodal Properties File		NODALPROPS
NOTE: See User settings for data file options					
[project.generated results directory]	Directory	N/A	Generated Results Directory		
<b>FEA Solutions</b>					

Pro/Mechanica Interface Options (Pro/ENGINEER CREO or Wildfire)					
[project.interfaces.pro-e.default dataset]	Boolean	TRUE	Default Dataset for Fatigue Results	Insert the fatigue results into an existing placeholder dataset or append one on the end of the stresses results	PROE_USEHOLDER
[project.interfaces.pro-e.h-node policy]	Boolean	FALSE	H-node variable values	Interpolate h-node data from p-nodes variable values (1) Full analysis read h-node data (2)	TREAT_HNODES_AS_PNODES
[project.interfaces.pro-e.Pro-E solid export vars]	String		List of Pro-E variables to export solids to		PROE_SOLID_EXPORT_VARS
[project.interfaces.pro-e.Pro-E shell export vars]	String		List of Pro-E variables to export shells to		PROE_SHELL_EXPORT_VARS
Abaqus ODB Interface Options					
<i>NOTE: See User settings for the Determine ODB version and Allow multiple parts/instances settings</i>					
[project.interfaces.ODB.merge results with input]	Boolean	FALSE	Default to merge results with input ODB	Default the output results to the input .odb database	ODB_INEQOUT
[project.interfaces.ODB.data position]	Enumeration		Extract data at* ...?	Position of data to read from the ODB database; 'centroidal' (Centroidal), 'integration' (Integration points), 'nodal' (Nodal averaged), 'element_nodal' (Element nodes (recommended))	ODB_POSTYPE
[project.interfaces.ODB.last increment only]	Boolean	TRUE	Extract just the last increment in a step*	Extract just the last increment in a step	ODB_LASTFRAME

Appendix E

[project.interfaces.ODB.read SET groups]	Boolean	TRUE	Read groups from ESETs and NSETs	When reading groups, groups will be created from ESETs and NSETs	
[project.interfaces.ODB.read section groups]	Boolean	TRUE	Create groups based on section types	When reading groups, groups will be created based on section types	
[project.interfaces.ODB.read material groups]	Boolean	TRUE	Create groups based on material	When reading groups, groups will be created for each material	
[project.interfaces.ODB.vectors as tensors]	Boolean	FALSE	Export vectors as tensors	Export vectors as tensors	TENSOR_ODBWRITE
[project.interfaces.ODB.cache forces]	Boolean	FALSE	Cache nodal forces	Reduce memory overhead needed for reading nodal forces. This option should be used only if problems have been encountered reading nodal force datasets	ODB_CACHE_FORCES
[project.interfaces.ODB.use legacy step name]	Boolean	FALSE	Use the legacy fatigue load step	Use the legacy fatigue load step (stepId, date, time)	ODB_LEGACY_STEP_NAME
[project.interfaces.ODB.step name]	String	fe-safe_	Fatigue Load Step	The name of the fatigue load step. Default is 'fe-safe_' with a number	ODB_STEP_NAME
[project.interfaces.ODB.odb version]	String		ODB interface version for this job	ODB interface version for this job	
<b>Abaqus FIL Interface Options</b>					
[project.interfaces.FIL.export field]	Enumeration		Export Lives / FOS to FIL as ...?	Export code for the fatigue results. 'user' (User-defined ...), 'TEMP' (TEMP - 2), 'UVARM' (UVARM - 87 (most viewers))	FIL_CODE

[project.interfaces.FIL.output format]	Integer	2	Format	.fil files are ASCII (0), Binary (1) or auto-detect (2)	FIL_ASCIIORBINARY
[project.interfaces.FIL.omit stress steps]	Boolean	TRUE	Do not write stress steps to output	Only export the analysis results rather than all steps from the original .fil file	FIL_SKIPSTEPS
[project.interfaces.FIL.data position]	Enumeration	Elemental	Extract Stresses at* ...?	Type of stress data to extract from the .fil file; 'integration_points' (Integration points), 'centroidal' (Centroidal), 'element_nodal' (Element nodes (recommended)), 'nodal_averaged' (Nodal averaged)	FIL_POSTYPE
<b>FEMAP / CSV Interface Options</b>					
[project.interfaces.CSV.export shortest value]	Boolean	TRUE	Treatment of multiple results per ID	Export the Life as the shortest (TRUE) or the mean (FALSE) for an element to the file	CSV_SHORTEST
[project.interfaces.CSV.export all values]	Boolean	FALSE	Treatment of multiple results per ID	Export one value per node	CSV_PERNODE
[project.interfaces.CSV.all ASCII as default]	Boolean	FALSE	Default all results to ASCII	Default outputs to CSV files	CSV_ASDEFAULT
[project.interfaces.CSV.include headers]	Boolean	FALSE	Include a header with results	Add header to CSV file describing the Fatigue results	CSV_HEADERS
[project.interfaces.CSV.separator character]	String	,	Separator character	Separator to use for exports of CSV files, Either SPACE, TAB or COMMA (,)	CSV_SEP
<b>Save Loaded FE Models</b>					
[project.interfaces.CSV.save group]	Boolean	TRUE	Save group information		CSV_GROUPS

Appendix E

[project.interfaces.CSV.save geometry info]	Boolean	FALSE	Save geometry information		CSV_GEOM
[project.interfaces.CSV.accurate output]	Boolean	TRUE	Use 5 significant figure exponent format		CSV_ACCURATE
[project.interfaces.CSV.include trailing zeros]	Boolean	FALSE	Include +/- and trailing 0s (i.e. +3.200E+03)		CSV_EXPANDZEROS
[project.interfaces.CSV.fixed width fields]	Boolean	FALSE	Make Ids fixed width fields (add spaces)		CSV_FIXEDWIDTH
[project.interfaces.CSV.export list]	String	All	List of elements/nodes to export	This is a list of elements to export if your stresses are elemental, or nodes if your stresses are nodal.(e.g. 1-57, 1024, 567, 'All' for all) Add a . for a particular node on an element (e.g. 6.1) and a : for a shell layer number. (e.g. 6.1:2).	CSV_IDS
[project.interfaces.CSV.export worst case]	Boolean	FALSE	Export worst case temperatures as a dataset		CSV_EXPORTTWT
<b>Nastran Interface Options</b>					
[project.interfaces.NASTRAN.export only analysed]	Boolean	TRUE	Export only analysed elements	Export only analysed elements	OP2_FER_ELS_ONLY
[project.interfaces.NASTRAN.stress default]	Real	16	Unanalysed stress default:	Unanalysed stress default value	OP2_DEF_VAL
[project.interfaces.NASTRAN.later than MSC 2001]	Boolean	FALSE	OP2 is in MSC 2001+ format	OP2 is in MSC 2001+ format	OP2_2001
[project.interfaces.NASTRAN.enable label]	Boolean	FALSE	OP2 is labelled	Labelled implies POST=-1, e.g. PatranUnlabelled implies POST=-2, e.g. I-DEAS	OP2_LABELED

[project.interfaces.NASTRAN.data position]	Integer	256	Extract Stresses at* ...?	Extract Stresses at Element Nodes (0), Element Centroid (3), or Both (256) or (4), Default=256	NAS_POSTTYPE
[project.interfaces.NASTRAN.groups from SETS]	Boolean	TRUE	Create groups from OP2 SETs	The SET lists will be used to create groups	
[project.interfaces.NASTRAN.groups from properties]	Boolean	TRUE	Create groups from OP2 element properties	Element properties will be used to create groups	
[project.interfaces.NASTRAN.groups from materials]	Boolean	TRUE	Create groups from OP2 materials	Element materials will be used to create groups	
<b>ANSYS .RST Interface Options</b>					
[project.interfaces.ANSYS.extract temperatures]	Boolean	FALSE	Read temperatures from .RST model	Read temperatures from .RST model	EXTRACT_TEMPS
[project.interfaces.ANSYS.extract DOF temperatures]	Boolean	TRUE	Read DOF solution from thermal analysis	Read DOF nodal temperatures when reading temperatures	EXTRACT_DOF_TEMPS
[project.interfaces.ANSYS.extract boundary conditions]	Boolean	TRUE	Read elemental boundary conditions	Read elemental boundary conditions when reading temperatures	EXTRACT_EL_TEMPS
[project.interfaces.ANSYS.use worst for critical plane export]	Boolean	TRUE	For critical plane export use worst on element		RST_WORST
<b>HyperMesh Results Options</b>					
[project.interfaces.Hypermesh.hypermesh as default]	Boolean	FALSE	Default all results to HyperMesh	Default results to Hypermesh format	HM_ASDEFAULT
[project.interfaces.Hypermesh.storage type]	Integer	1	Storage Type per Element	Export element type results as the shortest on an element (1) or as the mean on an element (0)	HM_SHORTEST

Appendix E

[project.interfaces.Hypermesh.data position]	Enumeration		Fatigue Results Position*	Export fatigue results as 'nodal' (Nodes), 'element_nodal' (Elements), 'default ' (Default (recommended))	HM_ASELEMENTS
<b>I-DEAS Masterseries UNV Interface Options</b>					
[project.interfaces.I-DEAS.data position]	Integer	3	Extract Stresses at* ...?	Extract element stresses (3) or nodal stresses (1)	UNVNODAL
[project.interfaces.I-DEAS.load node group]	Boolean	FALSE	Don't load node group	For element data element groups are extracted. For nodal data nodal groups and colour codes are extracted.	UNVSKIPGROUPS
[project.interfaces.I-DEAS.extra groups dataset id]	Integer	2435	Extra Groups Dataset #	This allows new Permanent Group records to be read. They must have the same format as UNV dataset number 2435.	UNVEXTRAGROUPCODE
[project.interfaces.I-DEAS.vectors as tensors]	Boolean	FALSE	Export vectors as tensors		UNV_WRITETENSOR
[project.interfaces.I-DEAS.merge results with input]	Boolean	FALSE	Include input UNV in the UNV results file	The output results will be duplicated at the export location and results appended	
<b>POSTPR Interface Options</b>					
[project.interfaces.POSTPR.data position]	Integer	1	Extract data at* ...?	Extract nodal data (1) or integration-point data (2)	POSTPR_DATA_POS
[project.interfaces.POSTPR.elemental strain-type]	Integer	0	Read integration-point strains from ...?	Read total strain (0) or elastic + plastic strain (1)	POSTPR_GAUSS_STRAIN_TYPE
<b>Open finite element model using rotational</b>					



symmetry (fe-safe/Rotate)					
[project.model.rotate model]	Boolean	FALSE	Rotated FE model	Indicates the loaded FE model was opened using rotate	OPEN_MODE
[project.model.rotation.number of files appended]	Integer	0	How many Source Files have been appended		
[project.model.rotation.axis]	Enumeration	x_axis	Axis of rotational symmetry x_axis, y_axis, or z_axis		ROTATIONAL_AXIS
[project.model.rotation.number of segments]	Integer	8	Number of rotations of master segment needed (n)		ROTATIONAL_NUMSEGMENTS
[project.model.rotation.number of solutions]	Integer	1	Number of solutions in master segment (n2)		ROTATIONAL_NUMSOLUTIONS
[project.model.rotation.tolerance]	Real	-1	Warning tolerance for finding rotated elements		ROTATIONAL_TOLERANCE
[project.model.rotation.material groups]	String		List (space delimited) of group names defining the master segment		ROTATIONAL_MATERIAL
[project.model.rotation.auto-add master groups]	Boolean	FALSE	Automatically add groups starting with 'M_' to the master segment		ROTATIONAL_AUTO_MASTER
[project.model.rotation.include toggle]	Boolean	FALSE	Definition of groups in excluded materials will be excluded (false) or included (true)		
[project.model.rotation.excluded materials]	String		List of group names (to be excluded/included as per the include toggle)		ROTATIONAL_EXCLUDED MATS

Appendix E

[project.model.rotation.auto-filter groups]	Boolean	FALSE	Automatically include/exclude groups from rotation, starting with 'X_' or in RST files with material numbers of 100 or more		ROTATIONAL_AUTO_EXCLUDE
[project.model.rotation.mirror axis]	Enumeration		to be used in conjunction with the axis setting		ROTATIONAL_MIRROR_AXIS
[project.model.rotation.number of created datasets]			Number of datasets created on each open/append (n x n2)		
<b>Open finite element model for PSD analysis</b>					
[project.model.psd.modal participation factor files(n)]	File	N/A	Files that provide Modal Participation Factor (MPF) data		
[project.model.psd.psd_files(n)]	File	N/A	Files that provide Power Spectral Density (PSD) data		
[project.model.psd.modal participation factor complex number notation]	Enumeration	Rectangular	Complex number notation 'Rectangular' 'Polar (degrees)' or 'Polar (radians)'		
<b>Current FE models</b>					
[project.model.stress units.description]	String	Pa	Stress units	Stress units for loaded models	MODEL_SUNITS
[project.model.stress units.scale]	Real	1	User-defined Stress scale	Scale if Stress units are user defined	MODEL_SSCALE
[project.model.stress units.offset]	Real	0	User-defined Stress offset	Offset if Stress units are user defined	MODEL_SOFFSET
[project.model.strain units.description]	String	strain	Strain units	Strain units for loaded models	MODEL_EUNITS
[project.model.strain units.scale]	Real	1	User-defined Strain scale	Scale if Strain units are user defined	MODEL_ESCALE

[project.model.strain units.offset]	Real	0	User-defined Strain offset	Offset if Strain units are user defined	MODEL_EOFFSET
[project.model.temperature units.description]	String	deg.C	Temperature units	Temperature units for loaded models	MODEL_TUNITS
[project.model.temperature units.scale]	Real	1	User-defined Temperature scale	Scale if Temperature units are user defined	MODEL_TSCALE
[project.model.temperature units.offset]	Real	0	User-defined Temperature offset	Offset if Temperature units are user defined	MODEL_TOFFSET
[project.model.force units.description]	String	N	Force units	Force units for loaded models	MODEL_FUNITS
[project.model.force units.scale]	Real	1	User-defined Force scale	Scale if Force units are user defined	MODEL_FSCALE
[project.model.force units.offset]	Real	0	User-defined Force offset	Offset if Force units are user defined	MODEL_FOFFSET
[project.model.distance units.description]	String	mm	Distance units	Distance units for loaded models	MODEL_DUNITS
[project.model.distance units.scale]	Real	1	User-defined Distance scale	Scale if Distance units are user defined	MODEL_DSCALE
[project.model.distance units.offset]	Real	0	User-defined Distance offset	Offset if Distance units are user defined	MODEL_DOFFSET
[project.model.shell names (n)]	String				
<b>Analysis Options (Import tab)</b>					
<i>NOTE for 'Suppress project chooser at start-up' dialogue and 'Pre-scan options' see the user settings</i>					
[project.interfaces.load all groups]	Boolean	TRUE	Use loaded groups in Group Parameters table		LOAD_ALL_GROUPS
[project.model.extract strains]	Boolean	FALSE	Read strains from FE Models*	Extract strain as well as stress data	EXTRACT_STRAINS

Appendix E

[project.model.extract strain type]	Enumeration	Total	Strain type	Type of strains to read from the FE model; 'Total' (Total (default)), 'Logarithmic' (Logarithmic), 'Elastic_plus_Plastic' (Elastic plus Plastic), 'Nominal' (Nominal)	
[project.model.extract forces]	Boolean	FALSE	Read forces from FE Models*		EXTRACT_FORCES
[project.model.surface as nodal]	Boolean	TRUE	Surface element definition	For setting the Surface finder options radio button, select (True) for "Has one or more nodes on the surface" or (False) for "Has one or more faces on the surface"	SURFACE_AS_NODAL
<b>Analysis Options (Export tab)</b>					
[project.job.exports.logarithmic lives]	Boolean	TRUE	Export logarithmic lives to results file	Export log of lives (1) instead of lives (0) to the results file	LOGLIVES
[project.job.overflow value]	Real	0	Overflow Life value	Value to export for lives where non-fatigue failures occurred due to excessive stresses.	OVERFLOWVALUE
[project.job.infinite life value]	Real	-1	Infinite Life value	Value to export to represent infinite life (-1 indicates use the material's CAEL).	LIFE_INF
[project.job.exports.ascii header]	Boolean	TRUE	Save &ASCII data with a header		ASCII_HEADER
[project.job.exports.results.contour policy]	Integer	-1	Export Contour Options		CONTOUR_METHOD
<b>Analysis Options (General tab)</b>					

[project.job.ignore compressive cycles]	Boolean	FALSE	Assume infinite life for fully compressive cycles	Ignore fully compressive stress cycles, when plasticity is enabled the corrected stress cycle is used	IGNORE_COMP_CYCLES
[project.job.disable temperature analysis]	Boolean	FALSE	Disable temperature-based analysis	Disable temperature effects on fatigue properties	DISABLETEMPANALYSIS
[project.job.scale factor]	Real	1	Additional effects scale factor	Additional effects scale factor to apply to stresses	SCALEFAC
[project.job.gating.tensor gate enabled]	Boolean	TRUE	Gate tensors (as % of max tensor)	Gate tensors (as % of max tensor)	GATETENSORS
[project.job.gating.tensor gate]	Real	5	Gate value	Gate value for tensors as a % of the maximum value in the tensor	GATETENSORS_VALPC
[project.job.gating.load history gate enabled]	Boolean	FALSE	Pre-gate load histories with % gate	Pre-gate load histories with % gate	GATELH
[project.job.gating.load history gate]	Real	5	Load history gate value	Gate value for load histories as a % of the maximum value in the history	GATELH_VALPC
[project.job.gating.near zero stress]	Real	10	Near-zero stress value (MPa)	Gate value for near-zero stress tensor components when evaluating stress histories and principals	
[project.job.gating.near zero strain]	Real	10	Near-zero strain value (uE)	Gate value for near-zero strain tensor components when evaluating strain histories	

Appendix E

[project.job.gating.CAEL gate enabled]	Boolean	TRUE	Perform nodal elimination using material's CAEL	Gate nodes using the constant amplitude endurance limit of the material and the worst possible cycle in the loading	GATECAEL
[project.job.gating.trigonometric look-up tables]	Boolean	TRUE	Use trigonometric look-up tables (recommended)	Use trig function lookup tables	USELOOKUPS
[project.job.disable triaxiality]	Boolean	FALSE	Disable triaxial stress and strain treatment	Disable triaxial fatigue calculations	DISABLETRIAx
[project.job.disable failed directional cosine to XYZ]	Boolean	FALSE	Disable failed directional cosines to XYZ	Disable defaulting to XYZ orientation on error	DISABLEDCXYZ
[project.job.material databases.nodal property mapping enabled]	Boolean	TRUE	Enable nodal property mapping		ENABLE_NODAL_PROPERTY_MAPPING
[project.job.planes.critical plane search count]	Integer	18	Critical plane search count	The number of equally spaced planes to search from 2 to less than 180 degrees	
[project.job.max overflows]	Integer	1000	Maximum number of overflows to report	Outputting a large number of overflows can degrade performance (Minimum=0, Maximum=1e6)	MAX_OVERFLOWs
[project.job.solver limit]	Integer	-1	Number of nodes to be solved simultaneously	Number of nodes to be solved simultaneously; values higher than the maximum core count will be ignored. A value of -1 indicates to use the maximum core count for the system.	Solver_Limit

[project.job.thread limit]	Integer	-1	Maximum number of threads available to all solvers	Maximum number of threads available to all solvers (as a combined total). A value of -1 indicates to use the maximum core count for the system.	Thread_Limit
[project.job.enforce thread affinity]	Boolean	FALSE	Force process CPU core affinity based on the thread and solver limits	Analysis and non-analysis threads will only use CPU cores they have affinity with. A program restart may be required for a thread limit increase to take effect.	
<b>Analysis Options (Properties tab)</b>					
[project.job.material databases.warnings enabled]	Boolean	TRUE	Material warning enabled	If enabled, material validation problems will be reported	MATLWARNING
[project.job.material databases.temperature interpolation]	Enumeration	linear	Interpolation method between defined temperatures:	Permitted values: 'linear' (the material data will use linear interpolation), 'log' (the material data will use log interpolation)	TEMP_INTERPOLATION
[project.job.material databases.temperature clamp warning]	Boolean	FALSE	Warn when temperature at a node is beyond the range defined for the material		MAT_TEMP_CLAMP_WARN
[project.job.material databases.material extrapolation warning]	Boolean	FALSE	Warn when SN data is extrapolated beyond the range of values defined for the material		MAT_EXTRAP_WARN
[project.job.material databases.R-ratio clamp warning]	Boolean	FALSE	Warn when a cycle has a stress ratio beyond the range defined for the material		MAT_R_CLAMP_WARN
<b>Analysis Options (Algorithms tab)</b>					

Appendix E

<b>Cast Iron</b>					
[project.job.material databases.Downing coefficient]	Real	2.549999952	Damage summation coefficient	Damage summation coefficient	P_COEFF
[project.job.material databases.Downing exponent]	Real	-0.800000012	Damage summation exponent	Damage summation exponent	P_EXP
<b>Modal Analysis</b>					
[project.job.modal samples per cycle]	Integer	10	Modal block samples per cycle	Number of samples to interpolate for each cycle in the generated loading	MODAL_NPERCYCLE
[project.job.modal gate percentage]	Real	0.1	Mode gate as % of maximum amplitude	Mode gate as % of maximum amplitude (Minimum=0, Maximum=100)	MODAL_GATEPC
<b>PSD</b>					
[project.job.psd.PSD response]	Enumeration		PSD response	Method used to calculate the PSD response; 'Von Mises' (Von Mises), 'Critical plane (normal)' (Critical plane (normal)), 'Critical plane (shear)' (Critical plane (shear)), 'Critical plane (shear & normal)' (Critical plane (shear & normal))	PSD_RESPONSE
[project.job.psd.Apply nodal filtering]	Boolean	FALSE	Implement Von Mises-based nodal filtering	Apply nodal filtering to a critical plane analysis	PSD_APPLY_NODAL_FILTERING
[project.job.psd.Number of stress range intervals]	Integer	1000	Number of stress range intervals	Number of stress range intervals for damage evaluation	PSD_NO_STRESS_RANGE_INTERVALS



[project.job.psd.RMS stress cut-off multiple]	Real	10	RMS stress cut-off multiple	RMS multiple that determines the PDF stress range for evaluation	PSD_RMS_STRESS_CUTOFF_MULTIPLE
[project.job.psd.RMS multiple]	Real	3	RMS multiple	RMS multiple that forms part of the denominator in the expression for FRF (Vertical)	PSD_RMS_FRF_MULTIPLE
[project.job.psd.Combined shear-normal K]	Real	0.25	Combined shear-normal K	Weighting factor on normal stress in combined shear and normal critical plane search	
<b>Plugin (CMF or Custom Module Framework Plugin)</b>					
[project.job.material databases.strain-life table enabled]	Boolean	TRUE	Default to using EN data for strain analysis		ENABLE_EN_TABLE
[project.job.material databases.stress-life table enabled]	Boolean	TRUE	Use SN curves	For stress-based analyses use SN curves rather than local strain material data	SNCURVE
[project.job.material databases.stress-strain table enabled]	Boolean	TRUE	Default to using SE data for cyclic curve		ENABLE_SE_TABLE
[project.job.planes.nodal plane mapping enabled]	Boolean	FALSE	Enable custom nodal plane mapping		ENABLE_NODAL_PLANE_MAPPING
<b>Safety Factors (Critical Distance and FOS Band Definition)</b>					
[project.job.critical distance.apply corrections]	Boolean	FALSE	Apply corrections to safety factors (FOS or FRF)	Perform critical-distance corrections	APPLY_SAFETY_FACTOR_CORRECTIONS
[project.job.critical distance.method]	Enumeration		Critical distance method	'point' for point method, 'line' for line method	SAFETY_FACTOR_CORRECTION_METHOD

Appendix E

[project.job.critical distance.use safety-factor thresholds]	Boolean	TRUE	only when uncorrected factors are between		USE_SAFETY_FACTOR_CORRECTION_THRESHOLDS
[project.job.critical distance.safety-factor ceiling]	Real	10	Maximum	A ceiling, above which uncorrected safety-factors do not invoke a critical-distance correction	SAFETY_FACTOR_CORRECTION_CEILING
[project.job.critical distance.safety-factor floor]	Real	0	Minimum	A floor, below which uncorrected safety-factors do not invoke a critical-distance correction	SAFETY_FACTOR_CORRECTION_FLOOR
[project.job.critical distance.maximum surface convexity]	Real	180	Maximum surface convexity (degrees)		CRIT_DIST_MAX_SURF_CONVEXITY
[project.job.critical distance.maximum surface concavity]	Real	180	Maximum surface concavity (degrees)		CRIT_DIST_MAX_SURF_CONCAVITY
[project.job.critical distance.allow quadratic interpolation]	Boolean	TRUE	Allow quadratic interpolation	Allow quadratic interpolation in quadratic elements	ALLOW_QUADRATIC_INTERPOLATION
[project.job.fos.maximum]	Real	2	Maximum	Maximum coarse factor for FOS calculations	FOSCMAX
[project.job.fos.fine maximum]	Real	1.5	Maximum fine	Maximum fine factor for FOS calculations	FOSFMAX
[project.job.fos.fine minimum]	Real	0.8	Minimum fine	Minimum fine factor for FOS calculations	FOSFMIN
[project.job.fos.minimum]	Real	0.5	Minimum	Minimum coarse factor for FOS calculations	FOSCMIN
[project.job.fos.maximum coarse iterations]	Integer	4	Max coarse iterations	Maximum number of iterations for convergence of FOS	

				in the coarse band	
[project.job.fos.maximum fine iterations]	Integer	6	Max fine iterations	Maximum number of iterations for convergence of FOS in the fine band	
<b>Stress Analysis</b>					
[project.job.material databases.abort if no stress life data]	Boolean	TRUE	Block use of sf' and b if no SN datapoints	Flag to indicate if stress-based analyses should be aborted if a material has no SN data	ABORTIFNOSN
[project.job.material databases.correct elastic stress]	Boolean	FALSE	Plasticity correction	If enabled, apply Neuber plasticity correction (for HCF and LCF, requires K' and n')	Correct_Elastic_Stress
<b>TMF</b>					
[project.job.TURBOLife.non_dimensional_mean]	Boolean	FALSE	Mean Stress Treatment for Damage Partitioning of Cycles in fe-safe/TMF	Use non dimensional mean stresses when performing damage partitioning (default=no)	TMF_NONDIMMEAN
[project.job.Young's modulus enabled]	Boolean	TRUE	Youngs Modulus Used to Convert Stresses to Strains fe-safe/TMF	Use Young's modulus from room temperature rather than worst case temperature to convert stresses to strains	CREEP_USEYMODWC
<b>Von Mises</b>					
[project.job.Von Mises signed]	Boolean	FALSE	Apply sign from	Von Mises sign evaluated from largest principal rather than hydrostatic	VMSIGN

Appendix E

Load Equivalency (inside Loading Settings)					
[project.job.life units description]	String	Repeats	Life units description	Units for exporting life as (i.e Miles or Hours), also used to convert defined statistical lives and FOS lives to repeats	NF_UNITS
[project.job.life units factor]	Real	1	Life units scale factor	Factor to convert from repeats to life units, (e.g. for ?1 repeat is 27 miles? the factor is 27)	NF_CONVFACTOR
Material properties					
[project.job.material databases.materials(n).database]	File	N/A	Source database	Database from which material came. If this is omitted, the materials will not be refreshed from the database	MATL_DBASE
[project.job.material databases.materials(n).fatigue strength coefficient (sf')]	String	Undefined	Sf'	Fatigue Strength Coefficient (Sf')	SIGFP
[project.job.material databases.materials(n).fatigue ductility coefficient (Ef')]	String	Undefined	Ef'	Fatigue Ductility Coefficient (Ef')	EPSFP
[project.job.material databases.materials(n).fatigue strength exponent (b)]	String	Undefined	Basquin's Exponent (b)	Basquin's Exponent (b)	BASQUIN
[project.job.material databases.materials(n).fatigue ductility exponent (c)]	String	Undefined	Coffin's Exponent (c)	Coffin's Exponent (c)	COFFIN
[project.job.material databases.materials(n).K']	String	Undefined	K'	K'	KPRIME

[project.job.material databases.materials(n).Young's modulus]	String	Undefined	Young's modulus	Young's modulus	ELASMOD
[project.job.material databases.materials(n).ultimate tensile strength]	String	Undefined	UTS	Ultimate Tensile Strength	UTSSTRESS
[project.job.material databases.materials(n).n']	String	Undefined	n'	n'	NPRIME
[project.job.material databases.materials(n).CAEL]	String	Undefined	Constant Amplitude Endurance Limit	Constant amplitude endurance limit as a life value	CONSTAEL
[project.job.material databases.materials(n).0.2% Buch proof stress]	String	Undefined	Buch 0.2% proof stress	Buch 0.2% proof stress	2PCPSSSTRESS
[project.job.material databases.materials(n).material name]	String	Undefined	Material name	Material name	MATLNAME
[project.job.material databases.materials(n).temperature list]	String	Undefined	Temperature list	List of temperatures that parameters are defined at	TEMPLIST
[project.job.material databases.materials(n).R ratio list]	String	Undefined	R-Ratio List		R_RATIO_LIST
[project.job.material databases.materials(n).strain rate list]	String	-9999	Strain rate list	List of strain rates for which data is defined	StrainRateList
[project.job.material databases.materials(n).hours list]	String	-9999	Hours list	List of hours for which data is defined	HoursList
[project.job.material databases.materials(n).initial compressive stresses]	String	Undefined	List of compressive stresses	List of compressive stresses	initStressCompList
[project.job.material databases.materials(n).initial tensile stresses]	String	Undefined	List of tensile stresses	List of tensile stresses	initStressTensList
[project.job.material databases.materials(n).pre-soak factor]	String	Undefined	Pre-soak factor	Pre-soak factor	PreSoakFactor

Appendix E

[project.job.material databases.materials(n).Weibull slope]	String	-9999	Weibull slope	Statistical variability of material's parameters	WeibullSlope_BF
[project.job.material databases.materials(n).Weibull minimum]	String	-9999	Weibull minimum	Statistical variability of material's parameters	WeibullMin_QMUF
[project.job.material databases.materials(n).ultimate compressive strength]	String	Undefined	UCS	Ultimate Compressive Strength	UCS
[project.job.material databases.materials(n).tensile secant slope]	String	Undefined	Tensile Secant Slope	Tensile Secant Slope	M_T
[project.job.material databases.materials(n).compressive K']	String	Undefined	Compressive K'	Compressive K'	K_C
[project.job.material databases.materials(n).compressive n']	String	Undefined	Compressive n'	Compressive n'	N_C
[project.job.material databases.materials(n).compressive secant slope]	String	Undefined	Compressive Secant Slope	Compressive Secant Slope	M_C
[project.job.material databases.materials(n).modulus of unloading]	String	Undefined	Modulus Of Unloading	Modulus Of Unloading	MU
[project.job.material databases.materials(n).SWT coefficient]	String	Undefined	SWT Life curve coefficient	SWT Life curve coefficient (only required if MSC method is Smith- Topper-Watson)	SWT_COEFF
[project.job.material databases.materials(n).SWT exponent]	String	Undefined	SWT Life curve exponent	SWT Life curve exponent (only required if MSC method is Smith- Topper-Watson)	SWT_EXP
[project.job.material databases.materials(n).fatigue strength exponent above knee (b2)]	String	Undefined	b2	Basquin?s Exponent to be used above Knee_2nf (b2) (Optional)	BASQUIN2

[project.job.material databases.materials(n).life curve knee]	String	Undefined	Life curve knee	Life above which to change Basquin's exponent to b2 (Optional)	KNEE_2NF
[project.job.material databases.materials(n).in-phase loading factor]	String	-9999	Thermal factor 1	Thermal factor 1	PHD1
[project.job.material databases.materials(n).out-of-phase loading factor]	String	-9999	Thermal factor 2	Thermal factor 2	PHD2
[project.job.material databases.materials(n).compressive relaxed stresses]	String	Undefined	Stress relaxation table (compressive)	Stress relaxation table (compressive)	RelaxedStressComp
[project.job.material databases.materials(n).tensile relaxed stresses]	String	Undefined	Stress relaxation table (tensile)	Stress relaxation table (tensile)	RelaxedStressTens
[project.job.material databases.materials(n).Dang Van endurance limit stress]	String	Undefined	Dang Van Fatigue Limit stress list	Dang Van Fatigue Limit stress list	DangVanFLS
[project.job.material databases.materials(n).Dang Van stress ratio]	String	Undefined	Dang Van R values list	Dang Van R values list	DangVanR
[project.job.material databases.materials(n).Poisson's ratio]	String	-9999	Poissons ratio	Poissons ratio	MATL_POISSON
[project.job.material databases.materials(n).algorithm]	String	Undefined	Algorithm	Material?s default analysis algorithm	MATL_ALGORITHM
[project.job.material databases.materials(n).class]	String	Undefined	Type	Type of material	MATL_CLASS
[project.job.material databases.materials(n).units]	String	Undefined	Units	Units material uses. This is how the material is displayed. All values in this file are in SI units.	MATL_UNITS
[project.job.material databases.materials(n).quality]	String	Undefined	Data quality	Notes pertaining to the quality and usability of the material data	DATA-QUALITY

Appendix E

[project.job.material databases.materials(n).source]	String	Undefined	Data source	Notes pertaining to the original source of the material data	DATA-SOURCE
[project.job.material databases.materials(n).comment 1]	String	Undefined	Comment 1		COMMENT-1
[project.job.material databases.materials(n).comment 2]	String	Undefined	Comment 2		COMMENT-2
[project.job.material databases.materials(n).revision number]	String	Undefined	Revision Number		REVISION-NUMBER
[project.job.material databases.materials(n).revision date]	String	Undefined	Revision Date		REVISION-DATE
[project.job.material databases.materials(n).revision history]	String	-9999	Revision History		REVISION-HISTORY
[project.job.material databases.materials(n).default MSC]	String	Undefined	Default mean stress correction	Default mean stress correction for material. Used as infinite life envelope on Smith and Haigh diagrams if an analysis does not have a n envelope defined	DEFAULT_MSC
[project.job.material databases.materials(n).threshold stress intensity factor]	String	-9999	Threshold stress intensity factor	Threshold stress intensity factor	KTH
[project.job.material databases.materials(n).critical distance]	String	-9999	Critical distance	Critical distance	CRIT_DIST
[project.job.material databases.materials(n).alternating shear stress]	String	Undefined	Alternating Shear Stress		CPF_TW
[project.job.material databases.materials(n).alternating bending fatigue strength]	String	Undefined	Alternating Bending Fatigue Strength		CPF_SBW
[project.job.material	String	-9999	Walker Gamma (r is positive)		WALKER_GAMMA_R_POSITIVE



databases.materials(n).Walker gamma (r is positive)]					
[project.job.material databases.materials(n).Walker gamma (r is negative)]	String	-9999	Walker Gamma (r is negative)		WALKER_GAMMA_R_NEGATIVE
[project.job.material databases.materials(n).Default knock-down curve]	String	Undefined	Default Knock-Down Curve		DEFAULT_KNOCKDOWN
[project.job.material databases.materials(n).plugin integers(n)]	String				
[project.job.material databases.materials(n).plugin reals(n)]	String				
[project.job.material databases.materials(n).plugin real arrays(n)]	String				
[project.job.material databases.materials(n).plugin strings(n)]	String				
[project.job.material databases.materials(n).static fatigue strength coefficient (sf)]	String	Undefined	Sf	True Fracture Stress (Sf)	SIGF
[project.job.material databases.materials(n).Cast iron overload threshold]	String	Undefined	Cast iron overload threshold	Cast iron overload threshold	Cast_iron_overload_threshold
[project.job.material databases.materials(n).Cast iron overload SWT coefficient]	String	Undefined	Cast iron overload SWT Life curve coefficient	Cast iron overload SWT Life curve coefficient (only required for cast iron algorithm using overload curve when MSC method is Smith-Topper-Watson)	Overload_SWT_COEFF

Appendix E

[project.job.material databases.materials(n).Cast iron overload SWT exponent]	String	Undefined	Cast iron overload SWT Life curve exponent	Cast iron overload SWT Life curve exponent (only required for cast iron algorithm using overload curve when MSC method is Smith-Topper- Watson)	Overload_SWT_EXP
[project.job.material databases.materials(n).Cast iron overload fatigue strength coefficient (sf')]	String	Undefined	Cast iron overload Sf'	Cast iron overload Fatigue Strength Coefficient (Sf') (only required for cast iron algorithm using overload curve)	Overload_SIGFP
[project.job.material databases.materials(n).Cast iron overload static fatigue strength coefficient (sf)]	String	Undefined	Cast iron overload Sf	Cast iron overload Static Fatigue Strength Coefficient (Sf) (only required for cast iron algorithm using overload curve)	Overload_SIGF
[project.job.material databases.materials(n).Cast iron overload fatigue ductility coefficient (Ef')]	String	Undefined	Cast iron overload Ef'	Cast iron overload Fatigue Ductility Coefficient (Ef') (only required for cast iron algorithm using overload curve)	Overload_EPSFP
[project.job.material databases.materials(n).Cast iron overload fatigue strength exponent (b)]	String	Undefined	Cast iron overload Basquin's Exponent (b)	Cast iron overload Basquin's Exponent (b) (only required for cast iron algorithm using overload curve)	Overload_BASQUIN
[project.job.material databases.materials(n).Cast iron overload fatigue ductility exponent (c)]	String	Undefined	Cast iron overload Coffin's Exponent (c) (only required for cast iron algorithm using overload curve)	Cast iron overload Coffin's Exponent (c)	Overload_COFFIN
[project.job.material databases.materials(n).Cast iron overload fatigue strength exponent above knee (b2)]	String	Undefined	b2	Basquin's Exponent to be used above Knee_2nf (b2) (Optional) (only required for cast iron	Overload_BASQUIN2

				algorithm using overload curve)	
[project.job.material databases.materials(n).Cast iron overload life curve knee]	String	Undefined	Life curve knee	Life above which to change Basquin's exponent to b2 (Optional) (only required for cast iron algorithm using overload curve)	Overload_KNEE_2NF
[project.job.material databases.materials(n).fkm material]	String	-9999	FKM Material	Set according to the FKM Guideline	FKM_MAT
[project.job.material databases.materials(n).elongation]	String	-9999	Elongation	Used to calculate FKM safety factor	ELONG
[project.job.material databases.materials(n).grey iron index]	String	-9999	Grey iron index	Only used for FKM GG material type	FKM_GII
<b>Plug-in material properties</b>					
[project.job.material databases.materials(n).stress-life curve.control series]	String	Undefined	Stress Amplitude values	List of Stress Amplitude values in the curve	SN_S
[project.job.material databases.materials(n).stress-life curve.sample series]	String	Undefined	Life values	List of Life values in the curve	SN_N
[project.job.material databases.materials(n).torsion-life curve.control series]	String	Undefined	Torsional Stress Amplitude values	List of Torsional Stress Amplitude values in the curve	
[project.job.material databases.materials(n).torsion-life curve.sample series]	String	Undefined	Life values	List of Life values in the curve	
[project.job.material databases.materials(n).strain-life curve.control series]	String	Undefined	Strain Amplitude values	List of Strain Amplitude values in the curve	EN_E
[project.job.material databases.materials(n).strain-life curve.sample series]	String	Undefined	Life values	List of Life values in the curve	EN_N

Appendix E

curve.sample series]					
[project.job.material databases.materials(n).stress-strain curve.control series]	String	Undefined	Stress Amplitude values	List of Stress Amplitude values in the curve	SE_S
[project.job.material databases.materials(n).stress-strain curve.sample series]	String	Undefined	Strain Amplitude values	List of Strain Amplitude values in the curve	SE_E
<b>fe-safe/TMF material properties</b>					
[project.job.material databases.materials(n).creep.monotonic K']	String	Undefined	Monotonic K' Prime		MON_KP
[project.job.material databases.materials(n).creep.monotonic n']	String	Undefined	Monotonic N' Prime		MON_NP
[project.job.material databases.materials(n).creep.damage to harden]	String	-9999	Damage-to-Harden		NHARDEN
[project.job.material databases.materials(n).creep.strain rates]	String	Undefined	Strain Rates for Ductility Table		Ductility_erValues
[project.job.material databases.materials(n).creep.ductility values]	String	Undefined	Ductility for Ductility Table		Ductility_defValues
[project.job.material databases.materials(n).creep.strain list]	String	Undefined	Strain List		creepStrainList
[project.job.material databases.materials(n).creep.stress list]	String	Undefined	Stress List		creepStressList
[project.job.material databases.materials(n).creep.strain table]	String	Undefined	Strain Table		creepTableE
[project.job.material databases.materials(n).creep.stress table]	String	Undefined	Stress Table		creepTableS

[project.job.material databases.materials(n).creep.endurance limit]	String	-9999	Endurance Limit		CREEP-EL
[project.job.material databases.materials(n).creep.temperature threshold]	String	-9999	Temperature Threshold		CREEP_TTHRESH
[project.job.material databases.materials(n).creep.fatigue percentage values]	String	Undefined	Interaction Fatigue Damage (%)		interactionFatigue
[project.job.material databases.materials(n).creep.creep percentage values]	String	Undefined	Interaction Creep Damage (%)		interactionCreep
<i>fe-safe/TURBOlife material properties</i>					
[project.job.material databases.materials(n).fully reversed.fatigue ductility exponent (c)]	String	Undefined	Fatigue Ductility Exponent (c)		MATL_C_CC
[project.job.material databases.materials(n).creep reversed by plastic.fatigue ductility exponent (c)]	String	Undefined	Fatigue Ductility Exponent (c)		MATL_C_CP
[project.job.material databases.materials(n).plastic reversed by creep.fatigue ductility exponent (c)]	String	Undefined	Fatigue Ductility Exponent (c)		MATL_C_PC
[project.job.material databases.materials(n).fully reversed.fatigue ductility coefficient (Ef')]	String	Undefined	Fatigue Ductility Coefficient (Ef')		MATL_EFP_CC
[project.job.material databases.materials(n).creep reversed by plastic.fatigue ductility coefficient (Ef')]	String	Undefined	Fatigue Ductility Coefficient (Ef')		MATL_EFP_CP
[project.job.material databases.materials(n).plastic reversed by creep.fatigue ductility coefficient (Ef')]	String	Undefined	Fatigue Ductility Coefficient (Ef')		MATL_EFP_PC

Appendix E

Group Parameters					
[project.job.groups(n).number]	String	0	Number	The FED group which this set of parameters belong to. For the Default group this is 0	GROUPNUM
[project.job.groups(n).group name]	String	Default	Name	Text name of the group that parameters belong to. For the remainder group this is the text Remainder	GROUPNAME
[project.job.groups(n).display name]	String	Default	Display Name		DISPLAYNAME
[project.job.groups(n).subgroup]	Integer	0	Subgroup		SUBGROUP
[project.job.groups(n).surface finish]	Real	-1	Surface finish	The Ra or Rz value representing the surface finish	
[project.job.groups(n).surface description]	String	1	Surface description	The Kt value (if a number) or a description of the surface finish from the surface finish definition file	STRESSCON
[project.job.groups(n).surface description file]	String	UserDefinedKt	Surface finish definition file	Surface finish stress concentration definition file. (Only required if STRESSCON is non-numerical)	STRESSCON_FILE
<i>NOTE: Material is matched to Group Parameter using index n</i>					

[project.job.groups(n).algorithm]	String	DefaultMaterial	Algorithm	Type of analysis to perform. This should be one of : UniStrainLife, UniStressLife, vonMises, PrincipalStress, PrincipalStrain, BrownMiller, MaxShearStrain, WeldLife, Verity, or DefaultMaterial	ANALYSIS
[project.job.groups(n).mean stress correction]	String	Morrow	MSC Method	String defining either a Mean Stress correction or the token FRF. Valid values are : None, Smith-Topper-Watson, Morrow, Goodman, Gerber, Hydrostatic, Largest, Buch, User or FRF. (If the default material algorithm is used this is not required).	MEANSTCR
[project.job.groups(n).residual stress]	Real	0	Residual stress	In-plane residual stress (negative numbers are compressive)	RESIDUAL
[project.job.groups(n).residual units]	Enumeration		Residual stress units	In-plane residual stress units; 'metric' (Metric), 'imperial' (Imperial)	RESIDUAL_UNITS
[project.job.groups(n).stress life scale]	Real	1	Stress-life scale		SN_SCALE
[project.job.groups(n).use knock-down curve]	Boolean	FALSE	Use knock-down curve		KNOCK_DOWN
<b>Group Algorithm Selection options</b>					
<i>User Defined Mean Stress Correction (*.msc)</i>					

Appendix E

[project.job.groups(n).user MSC file]	File	N/A	MSC file	If Mean Stress correction is user-defined then this is the path of the correction file. If Mean Stress correction is FRF then this is the name of the FRF file. Otherwise not required.	USER_MSC_FILE
<b>Fatigue Reserve Factors (FRF)</b>					
[project.job.groups(n).frf life]	String		Design life for FRF	Design life in repeats for FRF calculation	FRFLIFE
<b>BS5400 Weld Finite Life (Weld)</b>					
[project.job.groups(n).weld class]	String		Weld class	Weld class for analyses where ANALYSIS is weldLife. This should be one of : B, C, D, E, F, F2, G. (Only required if ANALYSIS = WeldLife)	WELDCLASS
[project.job.groups(n).design criteria]	Real	-2	Design criteria (# standard devns)	Design Criteria as a number of standard deviations. (Only required if the algorithm is 'WeldLife')	DESIGNCRIT
<b>FKM Guidelines</b>					
[project.job.groups(n).FKM.surface treatment]	Enumeration	none	Type of surface treatment	Used to calculate surface treatment factor; 'none' (None), 'nitriding' (Nitriding), 'case-hardening' (Case hardening), 'carbo-nitriding' (Carbo-nitriding), 'cold-rolling' (Cold rolling), 'shot-peening'	FKM_TREAT



				(Shot peening), 'inductive-hardening' (Inductive/Flame hardening)	
[project.job.groups(n).FKM.technological size factor]	Real	1	Technological size factor	Sets technological size factor for FKM algorithm	FKM_SIZE
[project.job.groups(n).FKM.coating factor]	Real	1	Surface coating factor	Sets coating factor for Al alloys	FKM_COAT
[project.job.groups(n).FKM.surface notched]	Boolean	FALSE	Is component surface notched?	Used to calculate surface treatment factor	FKM_NOTCH
[project.job.groups(n).FKM.component casting quality]	Enumeration		Casting Quality	Used to calculate safety factor; 'no-ndt' (Casting not subject to NDT), 'is-ndt' (Casting subject to NDT), 'high-quality' (High quality cast component)	FKM_NDT
[project.job.groups(n).FKM.component inspected]	Boolean	FALSE	Is component regularly inspected?	Used to calculate safety factor	FKM_INSPECT
[project.job.groups(n).FKM.component failure severe]	Enumeration		Consequence of failure	Used to calculate safety factor; 'severe' (Loss of human life), 'mean' (Loss of entire structure), 'moderate' (Loss of secondary component)	FKM_SEVERE
<b>Virtual Gauges and Influence Coefficients</b>					
[project.job.IC.export_histogram]	Boolean	FALSE	Export Inf. coeffs as plottable histograms (.icp, .icn)	Export influence coefficients as plottable histograms	IC.exportICH

Appendix E

[project.job.IC.gauges(n).id]	Integer	-1	Placement ID	Id to place gauge at. I.e. element or node number	id
[project.job.IC.gauges(n).sub_id]	Integer	-1	Placement sub-ID	Sub Id. I.e. 5th node on element	subId
[project.job.IC.gauges(n).angle]	Real	0	Angle	Angle to local X direction on element	angle
[project.job.IC.gauges(n).type]	Integer	1	Type	1 = Single Strain Gauge 2 = 45 deg. Strain rosette 3 = Single Stress 4 = Stress tensor output	gaugeType
[project.job.IC.gauges(n).section]	String	Not a shell	Section number for gauge (shells)	Section number for gauge (shells)	section
[project.job.IC.gauges(n).instance]	String		Instance name for gauge item	Instance name for gauge item	
[project.job.IC.influence coefficients(n).dataset]	Integer	-1	Dataset		dataset
[project.job.IC.influence coefficients(n).load_id]	Integer	-1	ID		loadId
[project.job.IC.influence coefficients(n).description]	String	No description	Description		descr
[project.job.IC.influence coefficients(n).units]	String	None	Units		units
<b>Gauges</b>					
[project.job.gauge_setting.factor]	Integer	0	Gauge sample interpolation factor	Interpolation factor for creating gauge outputs. Provides more defined loops	Gauges.int_factor
[project.job.gauge_setting.save_gauge_summary]	Boolean	TRUE	Save gauge analysis summary results to .csv file	Save gauge analysis summary results to .csv file	SaveGaugeSummary
[project.job.gauge_setting.gauges_array(n).id]	Integer	-1	Placement ID	Id to place gauge at. I.e. element or node number	id

[project.job.gauge_setting.gauges_array(n).sub_id]	Integer	-1	Placement sub-ID	Sub Id. I.e. 5th node on element	subId
[project.job.gauge_setting.gauges_array(n).angle]	Real	0	Angle	Angle to local X direction on element	angle
[project.job.gauge_setting.gauges_array(n).type]	Integer	1	Type	1 = Single Strain Gauge 2 = 45 deg. Strain rosette 3 = Single Stress 4 = Stress tensor output	gaugeType
[project.job.gauge_setting.gauges_array(n).section]	String	Not a shell	Section number for gauge (shells)	Section number for gauge (shells)	section
[project.job.gauge_setting.gauges_array(n).instance]	String		Instance name for gauge item	Instance name for gauge item	
<b>Factor of Strength (FOS)</b>					
[project.job.fos.design_life]	Real	0	Design life for FOS	Design Life for FOS calculations. Units are as defined for fatigue life. 0 indicates do not calculate FOS.	SERVLIFE
<b>Probability (Failure rate and Survival/Reliability rate)</b>					
[project.job.probability.enabled]	Boolean	FALSE	Perform Failure Rate for Target Lives calculations?	Perform statistical analysis?	DOTARGETLIVES
[project.job.probability.target_lives]	String		Target lives list	List of target lives. The list is comma or space delimited, (e.g. 2000 5000 7000) ? units are as defined for fatigue life.	TARGETLIVES
[project.job.probability.probability_as_reliability]	Boolean	FALSE	Calculate Reliability Rate instead of Failure Rate	Export reliability rates rather than failure rates?	exportRF
[project.job.probability.load_variability]	Real	5	Load variability (%)	The load variability as a % for the statistical	LOADSD

Appendix E

				analysis	
Exports and Outputs (Contours)					
[project.job.exports.contours.life]	Boolean	TRUE	Life or LOG10(Life)	Export life or log of life as a contour	EXP_CONT_LIFE
[project.job.exports.contours.damage]	Boolean	FALSE	Damage	Export total damage as a contour	EXP_CONT_DAMAGE
[project.job.exports.contours.block damage]	Boolean	FALSE	Damage per loading block	Export per-block damage as a contour. This includes all repeats, even if transitions are on	
[project.job.exports.contours.FRF horizontal]	Boolean	TRUE	FRF (Horizontal)	Export Horizontal FRF as a contour	EXP_CONT_FRFH
[project.job.exports.contours.FRF vertical]	Boolean	TRUE	FRF (Vertical)	Export Vertical FRF as a contour	EXP_CONT_FRFV
[project.job.exports.contours.FRF radial]	Boolean	TRUE	FRF (Radial)	Export Radial FRF as a contour	EXP_CONT_FRFR
[project.job.exports.contours.FRF worst]	Boolean	TRUE	FRF (Worst of Horizontal, Vertical and Radial)	Export Worst of 3 FRFs as a contour	EXP_CONT_FRFW
[project.job.exports.contours.largest loading stress]	Boolean	FALSE	Largest stress in loading (SMAX)	Export Maximum Stress seen by an item as a contour	EXP_CONT_SMAX
[project.job.exports.contours.SMAXYS]	Boolean	FALSE	SMAX/0.2% Proof Stress	Export ratio of Maximum Stress and Proof Stress as a contour	EXP_CONT_SMAXYS
[project.job.exports.contours.SMAXUTS]	Boolean	FALSE	SMAX/UTS	Export ratio of Maximum Stress and UTS as a contour	EXP_CONT_SMAXUTS
[project.job.exports.contours.SM]	Boolean	FALSE	Worst cycle mean stress and damage parameter	Export worst damage amplitude and mean stress as a contour	EXP_CONT_SM

[project.job.exports.contours.SMPreNumber]	Boolean	FALSE	Worst cycle uncorrected mean stress and stress amplitude	Export worst cycle uncorrected mean stress and stress amplitude as contours	
[project.job.exports.contours.SRP nodal temperatures]	Boolean	FALSE	TURBOLife/SRP Evaluated Nodal Temperatures		EXP_CONT_SRPTEMP
[project.job.exports.contours.critical planes all]	Boolean	FALSE	Critical planes (all blocks)		EXP_CONT_CRIT_ALL
[project.job.exports.contours.critical planes worst]	Boolean	FALSE	Critical planes (worst block)		EXP_CONT_CRIT_WORST
[project.job.exports.contours.critical planes averaged]	Boolean	FALSE	Critical planes (average of blocks)		EXP_CONT_CRIT_AVG
[project.job.exports.contours.surface checks]	Boolean	FALSE	Eigenvector vs surface (all blocks)	Exports the dot-product of the reference tensor's eigenvector with the surface normal (all blocks)	
[project.job.exports.contours.triaxiality]	Boolean	FALSE	Triaxial status	Indicates whether the stress/strain history was proportional or triaxial	DIAG_CONT_TRIAX
[project.job.exports.contours.group index]	Boolean	FALSE	Group index	The index of the analysis group used when analysing the item	
[project.job.exports.contours.critical distance success]	Boolean	FALSE	Critical Distance success		
[project.job.exports.contours.critical distance diagnostics]	Boolean	FALSE	Critical Distance diagnostics		
[project.job.exports.contours.critical distance cycle]	Boolean	FALSE	Critical Distance cycle	Stress amplitude and mean stress (in MPa) of the largest cycle at the critical distance	

Appendix E

[project.job.exports.contours.turbolife accumulated stress strain]	Boolean	FALSE	TURBOlife/SRP accumulated stress and strain contours	Export TURBOlife/SRP accumulated stress and strain contours	
[project.job.exports.contours.temperature-dependent UTS and FS]	Boolean	FALSE	Temperature-dependent UTS and FS	Export worst-cycle nodal temperature-dependent UTS and Fatigue Strength	
[project.job.exports.contours.maximum nodal temperature]	Boolean	FALSE	Maximum nodal temperature	Export maximum nodal temperature	
[project.job.exports.contours.worst cycle dataset indices]	Boolean	FALSE	Dataset indices of worst FRF cycle	Export the dataset indices of the worst FRF cycle (min and max)	
[project.job.exports.contours.fatigue strength lower life]	Real	10000			
[project.job.exports.contours.fatigue strength upper life]	Real	10000000			
[project.job.exports.contours.NASA life2]	Boolean	TRUE	NASALife life without MDMC	Export alternative life or its logarithm as a contour	
[project.job.exports.traffic lights]	Boolean	FALSE	Traffic lights	A contour can be exported as a 'quick-look' feature. This will be 0 for a definite failure, 0.5 for a suspected failure (in the specified life range) and 1 for safe.	DIAG_TRAFFICLIGHTS
[project.job.exports.traffic lights lower]	Real	1000000	Lower life band for traffic lights	Lower life band for traffic lights	DIAG_TRAFFICLIGHTS_NF1
[project.job.exports.traffic lights upper]	Real	10000000	Upper life band for traffic lights	Upper life band for traffic lights	DIAG_TRAFFICLIGHTS_NF2
[project.job.exports.unit system]	Enumerator	Metric	Unit system for contours, histories	Defines the units of	

			and logs	<p>stress, strain, temperature etc. used in contours, logging and plottable histories.</p> <p>“Metric” for “Stress: MPa; Strain: microstrains; Temperature: degC; Force: N”</p> <p>“Imperial” for “Stress: ksi; Strain: microstrains; Temperature: degF; Force: lbf”</p> <p>“Model” to match the unit system of the FE solution</p>	
<b>Exports and Outputs (Histories)</b>					
[project.job.exports.plots.worst cycle.Haigh]	Boolean	FALSE	Worst-cycle Haigh diagram	Scatter-plots mean stress against damage parameter (e.g. stress/strain amplitude) for the worst rainflow cycle on each item.	EXP_PLOT_WHAIGH
[project.job.exports.plots.worst cycle.Smith]	Boolean	FALSE	Worst-cycle Smith diagram	Scatter-plots mean stress against the minimum and maximum stress for the worst rainflow cycle on each item.	EXP_PLOT_WSMITH
<b>Exports and Outputs (Worst Node Histories)</b>					
[project.job.exports.plots.worst node.Haigh critical plane]	Boolean	FALSE	Haigh diagram for critical plane	Scatter-plots mean stress against damage parameter (e.g. stress/strain amplitude) for each	EXP_PLOT_NHAIGH

Appendix E

				rainflow cycle on the critical plane.	
[project.job.exports.plots.worst node.Smith critical plane]	Boolean	FALSE	Smith diagram for critical plane	Scatter-plots mean stress against the minimum and maximum stress for each rainflow cycle on the critical plane, for stress-based algorithms.	EXP_PLOT_NSMITH
[project.job.exports.plots.worst node.Von Mises stress]	Boolean	FALSE	Von Mises Stress	Plots von Mises stress against gated history samples	EXP_PLOT_NVON
[project.job.exports.plots.worst node.ignore overflows]	Boolean	TRUE	Ignore non-fatigue failure items (overflows)	Ignore non-fatigue failure items when evaluating worst item	EXP_PLOT_NIGNOREOFLOW
<b>Exports and Outputs (Log)</b>					
[project.job.material databases.material diagnostics]	Boolean	FALSE	Export material diagnostics?	Export material diagnostics?	DEBUGMATL
[project.job.exports.logs.worst n lives]	Boolean	FALSE	Items with worst n lives, where n =	Lists the worst n items' lives and worst-cycle amplitudes	DIAG_LOG_WORSTN
[project.job.exports.logs.n]	Integer	100	Number of items to put into worst-n-items table	Number of items to put into worst-n-items table	DIAG_LOG_WORSTN_N
[project.job.exports.logs.critical distance summary]	Boolean	FALSE	Critical Distance summary	Summarises the diagnostic codes returned by Critical Distance analysis	
[project.job.exports.logs.ranked elimination table]	Boolean	FALSE	Ranked item elimination table	Tabulates items eliminated from analysis in each scale-and-combine loading block by comparing their largest cycle with the endurance limit	DIAG_LOG_RANKEDELIM



Exports and Outputs (List of Items)					
[project.model.debug items]	String	None	List of items to diagnose	List of elements/nodes to dump diagnostics for, in the format 1-27, 45.1, 56-58, 102:4.	DEBUGELS
[project.job.exports.only analyse listed items]	Boolean	FALSE	Only analyse listed items	Analyse only listed items	JUSTDEBUGELS
Exports and Outputs (Histories for Items)					
[project.job.exports.plots.stress tensors]	Boolean	FALSE	Load histories	Export stress/strain/temperature load histories prior to gating.	DIAG_PLOT_PRET
[project.job.exports.plots.stress tensors after gating]	Boolean	FALSE	Load histories after gating	Export stress/strain/temperature load histories after gating	DIAG_PLOT_POSTT
[project.job.exports.plots.principals]	Boolean	FALSE	Evaluated principals	Export in-surface principals. 'In-surface' means that, once the tensors have been rotated to the local coordinate system, any out-of-plane shears are neglected when evaluating principals.	DIAG_PLOT_PRINCS
[project.job.exports.plots.critical plane normals]	Boolean	FALSE	Uncorrected normal stress/strain on critical plane	Export normal stress and strain on the critical plane (without plasticity correction)	DIAG_PLOT_NORMSCP
[project.job.exports.plots.plasticity corrected critical plane normals]	Boolean	FALSE	Plasticity-corrected normal stress on critical plane	Export plasticity-corrected normal stress on the critical plane (at cycle start and end points).	
[project.job.exports.plots.all plane normals]	Boolean	FALSE	Uncorrected normal stress/strain on all planes	Export normal stress/strain on all	DIAG_PLOT_NORMSAP

Appendix E

				analysis planes (without plasticity correction).	
[project.job.exports.plots.Dang Van]	Boolean	FALSE	Dang Van	Export Dang Van plots	DIAG_PLOT_DANGVAN
[project.job.exports.plots.damage]	Boolean	FALSE	Damage vs plane	Export damage-versus-plane plots	DIAG_PLOT_DAMAGE
[project.job.exports.plots.TURBOlife]	Boolean	FALSE	TURBOlife plots		DIAG_PLOT_TURBO
[project.job.exports.plots.modal]	Boolean	FALSE	FFT	Modal analysis diagnostics to plot files	DIAG_PLOT_MODAL
[project.job.exports.plots.Von Mises stress]	Boolean	FALSE	Von Mises Stress	Plots von Mises stress against gated samples.	EXP_PLOT_VON
[project.job.exports.plots.Haigh]	Boolean	FALSE	Haigh diagram for critical plane	Scatter-plots mean stress against damage parameter (e.g. stress/strain amplitude) for each rainflow cycle on the critical plane.	EXP_PLOT_HAIGH
[project.job.exports.plots.Smith]	Boolean	FALSE	Smith diagram for critical plane	Scatter-plots mean stress against the minimum and maximum stress for each rainflow cycle on the critical plane, for stress-based algorithms.	EXP_PLOT_SMITH
[project.job.exports.plots.critical distance plots]	Boolean	FALSE	Critical distance stress vs depth	Export critical distance stress-vs-depth plots	
[project.job.exports.plots.critical distance tensor histories]	Boolean	FALSE	Critical distance stress tensors vs depth	Export critical distance stress tensor-vs-depth plots	

[project.job.exports.plots.PSD Frequency Response Function plots]	Boolean	FALSE	PSD frequency response function	Export PSD frequency response function for each block	
<b>Exports and Outputs (Log for Items)</b>					
[project.job.exports.logs.nodal information]	Boolean	FALSE	Item information and critical-plane orientation	For each loading block, lists the item's temperature, stress- state, the number of planes searched, critical-plane angle, shear-type and vector, and history lengths pre- and post- gating.	DIAG_LOG_NODALINFO
[project.job.exports.logs.block life table]	Boolean	FALSE	Block-life table	Tabulates the number of repeats n for each loading block and the fatigue life for n repeats of that block.	DIAG_LOG_BLOCK
[project.job.exports.logs.plane life table]	Boolean	FALSE	Plane-life table	Tabulates the planes on which the item was analysed, listing the search-axis, angle, shear-type and fatigue life.	DIAG_LOG_PLANE
[project.job.exports.logs.critical plane life table]	Boolean	FALSE	Cycle-life table for critical plane	Tabulates the 100 most damaging rainflow cycles on the critical plane, including sample numbers, elastic/elastic-plastic stress/strain and fatigue life.	DIAG_LOG_CYCLECP
[project.job.exports.logs.all planes life table]	Boolean	FALSE	Cycle-life table for all planes	Tabulates the 100 most damaging rainflow cycles on each analysed plane, including the search axis, angle and	DIAG_LOG_CYCLEAP

Appendix E

				shear-type, sample numbers, elastic/elastic-plastic stress/strain and fatigue life.	
[project.job.exports.logs.fos planes]	Boolean	FALSE	FOS plane-tracking table	Tabulates (i) the FOS on each analysed plane and (ii) for the critical plane, the life for each iteration of the FOS scale.	DIAG_LOG_FOS
[project.job.exports.logs.grey iron damage]	Boolean	FALSE	Cast iron damage accumulation table	For each block, lists damage accumulation on the critical plane for 20 segments of the fatigue life.	DIAG_LOG_GI
[project.job.exports.logs.dataset stresses]	Boolean	FALSE	Dataset stresses	Tabulates FE stress/strain/temperature datasets in both model and export units	DIAG_LOG_DATASETS
[project.job.exports.logs.ep residuals]	Boolean	FALSE	Elastic-plastic residuals	Tabulates the residual stress and strain normal to each analysis plane, for each loading block	DIAG_LOG_RESIDS
[project.job.exports.logs.stress and strain tensors]	Boolean	FALSE	Loading stress, strain and temperature	Tabulates the stress, strain and temperature loading before any gating	DIAG_LOG_TENSORS
[project.job.exports.logs.principals]	Boolean	FALSE	In-surface principals	Tabulates stress and strain principals for each (post-gating) sample of each loading block. 'In-surface' means that, once the tensors have been rotated to the local coordinate system, any out-of-plane shears are neglected when	DIAG_LOG_PRINCS

				evaluating principals.	
[project.job.exports.logs.load history]	Boolean	FALSE	Loading stress, strain and temperature after gating	Tabulates the stress, strain and temperature loading after gating. Gating of loading signals and/or tensor histories excludes samples which cannot form an end-point of a rainflow cycle.	DIAG_LOG_24
[project.job.exports.logs.TURBOLife]	Boolean	FALSE	TURBOLife tables		DIAG_LOG_TURBO
[project.job.exports.logs.critical distance items]	Boolean	FALSE	Critical Distance items	Lists the elements intersected by the inward surface-normal and the interpolated stress at various points along it.	
[project.job.exports.logs.PSD items]	Boolean	FALSE	PSD items	Export PSD items	
[project.job.exports.logs.load_sensitivity]	Boolean	FALSE	Enable load sensitivity analysis	Do sensitivity analysis	DIAG_LOG_SENSITIVITY
[project.job.exports.logs.history indexing]	Enumeration	One	History index basis (the number of the first sample of a history log output)	Indices refer to samples in the loading history or to loaded finite element datasets; 'Zero' (Zero-based indexing is consistent with plottable history files and the Non-Fatigue Failure log.), 'One' (One-based indexing (the default) is consistent with earlier versions of the software.)	
<b>TURBOLife Options</b>					

Appendix E

[project.job.TURBOLife.plasticity model]	Integer	0	Plasticity Method	Neuber(0), Negative E (1), Glinka (2), or Vertical (3)	TURBO_PMODEL
[project.job.TURBOLife.stress parameter]	Integer	0	Generalised Stress Parameter	Von Mises(0) or Tresca (1)	TURBO_SEQUIV
[project.job.TURBOLife.creep precedence]	Integer	0	Creep Table that Takes Precedence	Table A (0) or Table B (1)	TURBO_CREPPREC
[project.job.TURBOLife.elastic follow-up factor]	Real	0.33	Elastic follow-up factor, 1/Z (0->1)		TURBO_INVZ
[project.job.TURBOLife.max diagnostic table size]	Integer	10000	Maximum diagnostic table size		TURBO_MAXTABLESIZE
[project.job.TURBOLife.max loading iterations]	Integer	5	Maximum iterations of loading		TURBO_MAXITS
[project.job.TURBOLife.max iterations per fos]	Integer	3	Maximum iterations per FOS		TURBO_MAXFOSITS
[project.job.TURBOLife.zero damage contour percentage]	Real	50	Creep contour value for 0 damage (%)		TURBO_PCNODMG
[project.job.TURBOLife.SRP Tmax weight]	Real	0.5	T(max)	Minimum=0	TURBO_SRP_TMAX_WEIGHT
[project.job.TURBOLife.SRP Tvm max weight]	Real	0.5	T(vm-max)	Minimum=0	TURBO_SRP_TVMMAX_WEIGHT
[project.job.TURBOLife.SRP T constant]	Real	0	T constant	Minimum=-1000, Maximum=1000	TURBO_SRP_TCONST
<b>Define and Analyse Weld Geometry settings</b>					
[project.job.welds.merge groups]	Boolean	TRUE	Merge all groups into 'WELD' group		VERITY_MERGE_GRP
[project.job.welds.weld diagnostics enabled]	Boolean	FALSE	Output extensive diagnostics		VERITY_DEBUG
[project.job.welds.stress mode]	Enumeration	multiaxial	Select stress calculation mode	Select the Verity stress calculation mode; 'normal' (Normal), 'shear' (Shear), 'multiaxial' (Multiaxial)	VERITY_STRESS_MODE

[project.job.welds.shear beta]	Real	1.607	Shear weighting in multiaxial metric	This parameter is set to best match the shear S-N curve to the normal and weights shear in the WB-MOI metrics	
[project.job.welds.proportional variance threshold]	Real	0.98	Proportional variance threshold	Proportional variance threshold on the first eigenvalue of the normal-shear covariance matrix above which the multiaxial loading will be treated as proportional	
[project.job.welds.use modified wang_brown]	Boolean	TRUE	Use Modified Wang and Brown initialisation in multiaxial analysis		
[project.job.welds.use gough ellipse]	Boolean	TRUE	Use Gough ellipse to map each multiaxial reversal to equivalent normal amplitude		
<b>User Settings</b>					
[application.autoname maximum length]	Integer	64	Maximum characters for source name		AUTONAME_MAXLEN
[application.non-standard modules enabled]	Boolean	FALSE	Enable non-standard fatigue modules		ENABLE_NSF
[application.data module.cache extra info]	Boolean	TRUE	Store load history extra information with data		ST_DF_MODE
[application.data module.append timestamp]	Boolean	TRUE	Append timestamp to filename for generated signals	Append a unique time/date stamp to each generated signal filename. If disabled, an automatic numbering scheme will be used to ensure unique filenames.	

Appendix E

[application.materials module.default units]	String		Default units		MATL_DEFAULTUNITS
[application.models module.pre-scan.enabled]	Boolean	TRUE	Pre-scan enabled		PRESCAN_ENABLED
[application.models module.pre-scan.confirm]	Boolean	TRUE	Prompt for pre-scan		CONFIRM_PRESCAN
[application.interfaces.ODB interface.enable auto selection]	Boolean	TRUE	Determine ODB file version automatically	When enabled, this will try to determine the version of an ODB file automatically. This can lead to a slowdown when reading ODB models.	
[application.interfaces.ODB interface.enable instance support]	Boolean	TRUE	Allow ODB files with multiple parts and instances	When disabled, this prevent use of multiple ODB parts and instances and will strip any instance association.	



205.84 *fe-safe* FE data (FED) folder (.fed)

See section 205.4.3.

205.85 *fe-safe* FE fatigue results (FER) file (\*.fer)

The FER file is an ASCII format working file, into which fatigue results are saved before exporting the results to the desired third-party file format.

The format comprises of 2 sections.

- The results header.

This section is marked by the token HEADER. It contains a summary of the analysis, a definition of which contour parameters have been exported and an indication of the position of the data.

An example is shown below :

```

HEADER
type=0, nVars=4
title1=fe-safe 5.2-00 [mswin];current.ldf;Remainder:BM:Mo--SAE_950C-Manten-local.dbase-
title2=-User defined Kt:1;KEYHOLE:BM:Mo--SAE-1005-system.dbase--Kt:1;keyhole.fil;
var1=LOGLife-Repeats
var2=FOS@Life=1E7-Repeats
var3=SMAX:MPa
var4=SMAX/Yield

```

The keywords have the following meanings.

Keyword	Meaning
type	The position of the contours. 0=Element nodes 1=Nodes 2=Integration points 3=Element centroids 4=Element nodes and centroids.
nVars	Number of contours written to file
title1	Short description of analysis
title2	Continuation of short description
var?	List of descriptions for variables.

- The contour values.

This section is marked by the token HEADER.

This is a multi column list of the data at each of the ids (see type).

The first 3 columns define the ID (i.e element or node number), the sub ID (i.e the 3<sup>rd</sup> node on an element) and the shell layer number (-1 for not a shell). The 4<sup>th</sup> and subsequent columns are the contour data values for the each of the variables.

CONTOURS

#	Item	SubItem	Shell	LOGLife-Rep	FOS@Life=1E7	SMAX:MPa	SMAX/Yield
1	1	1	-1	4.0009	0.5000	244.7314	0.9341
1	1	2	-1	4.0582	0.5000	267.3041	1.0202
1	1	3	-1	3.7086	0.5000	280.5811	1.0709
1	1	4	-1	3.6781	0.5000	257.1984	0.9817
2	2	1	-1	5.9950	0.7063	241.3601	0.9212
2	2	2	-1	5.9777	0.6875	236.4204	0.9024
2	2	3	-1	5.3440	0.5562	253.9013	0.9691
2	2	4	-1	5.3602	0.5562	259.0145	0.9886
3	3	1	-1	5.3838	0.5562	255.0024	0.9733
3	3	2	-1	5.4690	0.5750	265.3949	1.0130
3	3	3	-1	6.1202	0.7250	243.1214	0.9279
3	3	4	-1	6.0115	0.7063	234.8230	0.8963
4	4	1	-1	5.9479	0.6875	236.9708	0.9045
4	4	2	-1	6.0695	0.7063	242.0948	0.9240
4	4	3	-1	6.5911	0.8562	222.3377	0.8486
4	4	4	-1	6.4484	0.8125	219.7820	0.8389
5	5	1	-1	6.3973	0.8000	221.5186	0.8455
5	5	2	-1	6.5476	0.8438	222.5760	0.8495
5	5	3	-1	6.9725	0.9937	204.7777	0.7816
5	5	4	-1	6.7995	0.9312	206.4032	0.7878
6	6	1	-1	5.3367	0.5562	253.2322	0.9665
6	6	2	-1	5.3255	0.5562	255.9656	0.9770
6	6	3	-1	5.9132	0.6875	241.8407	0.9231
6	6	4	-1	5.9282	0.6875	238.9675	0.9121
7	7	1	-1	5.3759	0.5562	254.9280	0.9730
7	7	2	-1	5.3792	0.5562	260.9824	0.9961
7	7	3	-1	6.0160	0.7063	241.4682	0.9216
7	7	4	-1	6.0084	0.7063	235.8769	0.9003

#### 205.8.6 *fe-safe* material database (DBASE) file (\*.dbase)

These files contain tab-separated lists of material data and metadata, and divide into two sections.

- A **metadata section** for describing what parameters are available for each material and static information about those parameters. The metadata section defines invariant information about each material parameter rather than defining a material-dependent value for the parameter. The metadata section contains one or more material parameter metadata lines; each such line defines a material parameter metadata. For standard material data, the metadata section can be contained in a ".template" file, or embedded at the start of a ".dbase" file.
- A **data section** containing a list of materials, usually with values for each parameter specified in the metadata section. The data section contains one or more material data lines, with each line divided into values for the material parameters.

Blank lines and comment lines starting with "#" are skipped over for the duration of the reading, in any section of the file. The top of the file consists of a header section which can contain anything at all; the reader skips over these lines until it finds the following opening tag:

`_METADATA_START`

Every line after this is interpreted as a **material parameter metadata line** (detailed below), until the reader finds the following closing tag:

`_METADATA_END`

Every line after this, bar comment lines and empty lines, is interpreted as a **material data line** (detailed below). The reader stops when it finds a line that consists of one or more "@" characters, which acts as a closing tag for the material data section. If no such line is found, the reader will continue interpreting each line as a **material data line** until the end of the file.

### Material parameter metadata line

Comprises string values for each of the six following fields in order, separated with a single <TAB> character. There is no maximum length for any particular field.

<b>Field 1 (Link Name)</b>	Rarely used - only used as part of on-screen display text for link-type parameters. Can contain any character other than TAB.
<b>Field 2 (Key)</b>	Material parameters that have special meaning to the analysis engine are identified using this key. Listing all the string keys within this document is inappropriate as they increase with each release of fe-safe. The best reference is to look in the databases that are shipped with fe-safe or ask the development team for an up-to-date list. It is not intended that users edit these keys.  Material parameters that have no key defined are not used within the analysis in fe-safe and are for display purposes only.
<b>Field 3 (Type)</b>	Now defunct but needs to remain for legacy compatibility - can be an empty string. If empty ensure that there are two tab stops between the Key and Display field.
<b>Field 4 (Display Category)</b>	Contains a string in the following format: <ul style="list-style-type: none"> <li>• Zero or more ~ characters, followed by</li> <li>• a category name, followed by</li> <li>• a : character, followed by</li> <li>• Zero or more ~ characters, followed by</li> </ul>

	<ul style="list-style-type: none"> <li>a display name</li> </ul> <p>The category name is used purely to group parameters for display purposes and has no other significance. The display name is used to represent the parameter throughout the GUI. The first sequence of ~ characters are used to increase the priority of the category in the GUI. The more ~ characters are used, the higher the priority and the category will be listed before others in the GUI. Two categories with the same priority are sorted alphanumerically based on category name. The same rules apply to the ~ characters immediately preceding the display name; they can be used to manipulate the display order of the parameter within a category.</p>						
<p><b>Field 5 (Units)</b></p>	<p>Units property - can be "MPa", "deg.C", "MPa*m1/2".</p>						
<p><b>Field 6 (Field Length)</b></p>	<p>Now defunct but needs to remain for legacy compatibility - can be an empty string. If empty, ensure that there are two tab stops between the Units and Additional Data field.</p>						
<p><b>Field 7 (Additional Data)</b></p>	<p><b>Optional Field.</b></p> <p>The format for this field is a comma-separated list of key-value pairs, in this format:</p> <pre style="text-align: center;">"key1=value1,key2=value2,...keyn=valuen"</pre> <p>It should be surrounded by quotation marks, and internal whitespace adjacent to the "," and "=" characters is ignored. Some boolean keys do not have an associated value, their presence in the list indicates they are enabled. Key-parsing in the reader is case-insensitive.</p> <p>Valid keys and values are:</p> <table border="1" data-bbox="459 1283 1390 1984"> <thead> <tr> <th>Key</th> <th>Purpose</th> <th>Valid Values</th> </tr> </thead> <tbody> <tr> <td><code>edit</code></td> <td>Used to suggest an editor for the data, or to suppress editability altogether.</td> <td> <code>readonly</code>                      The data cannot be edited.  <code>table1d</code>                      The data sequence is a one-dimensional list of real values.  <code>table2d</code>                      The data sequence is a two-dimensional list of real values. Requires also the values of <code>dim1</code> and <code>dim2</code> to be set; these refer to other material attributes that determine the row series and column series respectively.  <code>table3d</code>                      The data sequence is a three-dimensional list of real values.                 </td> </tr> </tbody> </table>	Key	Purpose	Valid Values	<code>edit</code>	Used to suggest an editor for the data, or to suppress editability altogether.	<code>readonly</code> The data cannot be edited. <code>table1d</code> The data sequence is a one-dimensional list of real values. <code>table2d</code> The data sequence is a two-dimensional list of real values. Requires also the values of <code>dim1</code> and <code>dim2</code> to be set; these refer to other material attributes that determine the row series and column series respectively. <code>table3d</code> The data sequence is a three-dimensional list of real values.
Key	Purpose	Valid Values					
<code>edit</code>	Used to suggest an editor for the data, or to suppress editability altogether.	<code>readonly</code> The data cannot be edited. <code>table1d</code> The data sequence is a one-dimensional list of real values. <code>table2d</code> The data sequence is a two-dimensional list of real values. Requires also the values of <code>dim1</code> and <code>dim2</code> to be set; these refer to other material attributes that determine the row series and column series respectively. <code>table3d</code> The data sequence is a three-dimensional list of real values.					

			<p>Requires also the values of <code>dim1</code>, <code>dim2</code> and <code>dim3</code> to be set; these refer to other material attributes that determine the row series and column series, and a third "sorting" series respectively.</p> <p><code>tablePartner</code></p> <p>The data sequence is a one-dimensional list of real values, partnered with a second one-dimensional list of real data and presented in the GUI in a two-column table with its partner. Requires the value of <code>partner</code> to be set.</p> <p><code>dialogunits</code></p> <p>This is a unit parameter and requires a units editor.</p> <p><code>dialogalgorithm</code></p> <p>This is a algorithm parameter and requires a specialised algorithm picker.</p> <p><code>file</code></p> <p>This is a file parameter and requires a file browser.</p>
	<code>defaultdirectory</code>	Used to suggest a default directory to look for files associated with this parameter	
	<code>dim1, dim2, dim3</code>	Used to associate other parameters as dimensions for this parameter.	The names must match with the key of another parameter. This can be used to stipulate, for example, that this parameter is temperature-dependent.
	<code>partner</code>	Used to associate another parameter as a partner for this parameter. A partnered parameter is edited in tandem with its partner.	The name must match with the key of another parameter.
	<code>size</code>	Integer value used to indicate	A positive integer.

		the maximum size of the list for editing one-dimensional or partnered parameters.	
	<code>checks</code>	Can be used to force a progression rule on the data. This is in turn used in the GUI editor to check the validity of the user input.	<p><code>increasing</code> The data sequence must strictly increase with each successive value.</p> <p><code>decreasing</code> The data sequence must strictly decrease with each successive value.</p> <p><code>none</code> or <code>0</code> No progression rule will apply.</p> <p>The special values of <code>checks1</code> and <code>checks2</code> can be used to enforce progression rules on the first and second columns of partnered data</p>
	<code>label</code>	Can be used to suggest a text name for the series of data. The name is displayed in the editor GUI, usually as a row header or column header.	The special values of <code>label1</code> and <code>label2</code> can be used to suggest column names for the first and second columns of partnered data.
	<code>title</code>	Text name, only used for parameters with an edit hint of <code>tableSNPartner</code> . The name is displayed in the editor GUI to give a more user-	

		friendly name to the relationship between several parameters (e.g. "Ductility Table").	
	<code>isindex</code>	Boolean property, only used for parameters with an edit hint of <code>tableSNPartner</code> . If false, this parameter will not be used as the "sorting" field when displayed in the GUI.	<code>true/false</code>
	<code>options</code>	Contains a list of semicolon separated strings which for an enforce list of options to select from for the parameter. Used for string parameters only (i.e. where there is no edit hint).	Spaces are supported, however most special characters ;,:" and tabs are not.
	<code>filters</code>	Contains a list of file filters to use in the open dialogue (only affects parameters with a <code>file</code> edit hint).	The filters are separated by semicolons, for each filter the description is followed by optional file filters separated by colons e.g. <code>'MSC File:*.msc;Text Files:*.txt:*.csv'</code> .  An "All files" filter is always included and should not be specified.

**Material data line**

Comprises a row of <TAB> separated string values for each material parameter defined in the metadata section. If the number of values does not match the number of metadatas, a warning is issued. The order of the values within the line is significant in that it must match the ordering of the metadatas in the metadata section.

Undefined values can be represented by any of the following text values (omit the quotation marks):

- whitespace (other than <TAB>)
- "none" (case-insensitive)
- "undefined" (case-insensitive)
- "NA" (case-insensitive)
- "-9999"

Multi-dimensional lists have a special syntax in the file; whitespace within the list is ignored.

For one-dimensional lists of numbers, the following syntax is expected:

```
1, 2, 3, 4
```

For two-dimensional lists of numbers, the following syntax is expected:

```
(1, 2, 3, 4) (5, 6, 7, 8)
```

For three-dimensional lists of numbers, the following syntax is expected:

```
[(1, 2, 3, 4) (5, 6, 7, 8)] [(1, 2, 3, 4) (5, 6, 7, 8)] [(1, 2, 3, 4) (5, 6, 7, 8)]
```

**205.8.7 *fe-safe* material database template (TEMPLATE) file (\*.template)**

A template file contains metadata which associates data values in a database (\*.dbase) file with a keyword that *fe-safe* can interpret. The file also contains information for correctly displaying and formatting material data in the user interface - see 205.8.6 and section 8 of the user manual.

This format is deprecated in favour of embedded templates in \*.dbase files. It is used in conjunction with a \*.dbase file to contain metadata appropriate to that file; the \*.dbase file in this case would only contain material data. The top of the file consists of a header section which can contain anything at all; the reader skips over these lines until it finds the following opening tag:

```
START
```



Every line after this is interpreted as a **material parameter metadata line** (detailed above). There is no closing tag for the file; if any line after the opening tag is not a valid material parameter metadata, the file will be rejected.

#### 205.8.8 *fe-safe* material database configurations

Along with other system settings, the material database configurations are stored in the in the user's home directory. The list of material databases that are loaded at the start of an *fe-safe* session is stored in the user settings (the `user.stli`) file. When the user opens or closes material databases in the *fe-safe* GUI, these choices are stored in the user settings; so different users can have different lists of material databases.

If the user settings is empty (i.e. the user has cleared the settings, or perhaps this is the first time the software has been run), the list of material databases in the user settings is initially populated by a file in the user's home directory called `material_databases.xml`.

Finally, if this file does not exist or the user's home directory does not exist, the `material_databases.xml` is copied from the `<install_directory>/local` directory within the *fe-safe* installation. This file can be used by a site administrator to control the initial list of material databases that a user sees when they start the *fe-safe* software. Depending on how the *fe-safe* software was installed, it may require elevated permissions to edit this file. This file is generated by the *fe-safe* installer from the choices made at installation time.

#### 205.8.9 *fe-safe* surface finish definition file, \*.kt

This ASCII format file is based on the ASCII data file format (see section 205.2.3). This file is used to define surface finish properties accessible from the drop-down list in the **Surface Finish Picker** dialogue box (see section 5.6). The Surface finish definition files are stored in the `\kt` subdirectory of the *fe-safe* installation directory. There is no limit to the number of surface finish definition files that can be defined in this directory.

These files can be opened in the **Loaded Data Files** window, and plotted if required.

##### **Surface finish definition file format**

The first channel is the UTS list.

The second and subsequent channels define how Kt varies with UTS for the specified finish.

The title for surface finish is the channel name.

The UTS units can be either MPa or ksi and *fe-safe* does the necessary conversions.

## Sample file

A sample surface finish definition file is shown below:

```
DassaultSystemesASCII
0.01
Kt Data
UTS      Mirror Polished - Ra <= 0.25 um      0.25 < Ra <= 0.6 um
0.6 < Ra <= 1.6 um      1.6 < Ra <= 4 um      (cont)
Fine Machined - 4 < Ra <= 16 um      (cont)
Machined - 16 < Ra <= 40um      (cont)
Precision Forging - 40 < Ra <= 75 um 75 um < Ra      (cont)
MPa

0      1      1      1      1      1      1      1      1
100    1      1      1.005  1.03   1.033  1.05   1.06   1.09
200    1      1      1.015  1.06   1.067  1.1     1.13   1.17
300    1      1.003  1.03   1.08   1.1     1.15   1.21   1.26
400    1      1.015  1.05   1.1    1.125  1.2    1.3    1.35
500    1      1.032  1.065  1.125  1.15   1.26   1.4    1.45
600    1      1.049  1.075  1.15   1.18   1.32   1.49   1.56
700    1      1.05   1.08   1.17   1.2    1.38   1.6    1.7
800    1      1.05   1.085  1.18   1.23   1.45   1.72   1.88
900    1      1.05   1.089  1.1875 1.26   1.51   1.89   2.05
1000   1      1.05   1.09   1.194  1.29   1.61   2.05   2.26
1100   1      1.05   1.09   1.2    1.32   1.7    2.25   2.51
1200   1      1.05   1.09   1.2    1.36   1.8    2.45   2.77
```

### 205.8.10 FE model pre-scan files (\*.scan)

These are found in the scan directory in the current project directory: `<ProjectDir>\model\scan`. If valid, they contain a list of datasets details and information used to speed up reading those datasets. Names of those files are based on the loaded FEA file path and name, e.g. loading `C:\My_Data\File.op2` file would result in a `C--My_Data-File.op2.scan` file.

If there are any problems with importing information from a file, in particular when files with the same names are used, e.g. containing undetectable changes, deleting the pre-scan files and then reloading the model is recommended.

### 205.8.11 Models selected dataset list file (\*.slct)

As for \*.scan files, see section 205.8.10 above, these are located in the scan directory in the current project directory: `: <ProjectDir>\model\scan`. The same naming syntax is used but the extension is \*.slct instead of \*.scan. These files contain information on the position type (e.g. elemental, integration points etc.) and the indices of the selected datasets in the corresponding \*.scan file. This is used when a model is being loaded with pre-scanning enabled.

## 205.9 *fe-safe* working directory structure

### 205.9.1 *fe-safe* user directory

The user directory contains settings files that allow a user to record their preferences. Multiple users cannot use the same user directory in simultaneous sessions of *fe-safe*.

#### Controlling the location of the user directory

The user directory is specified in a file within the installation directory called `<install_directory>/setup/global.stli`. This file is populated according to the choices made when the

software was installed, and depending on how the software was installed elevated permissions may be necessary to change it.

The default situation is to have one user directory per installation of *fe-safe*, but this file can be edited to allow each user of the machine to have a different user directory, by altering the “user\_directory” setting. There are two possibilities:

**`${username}`** – resolves to the user’s name:

e.g. `<string name="user_directory ">C:\Users\${username}\Documents\my_user_directory</string>`

**`${any_environment_variable}`** – a system environment variable, which will be resolved when the *fe-safe* session starts:

e.g. `set FESAFE_USER_DIR>//network_share/fesafe_user`  
`<string name="user_directory ">${FESAFE_USER_DIR}</string>`

#### Key files and directories in the user directory

output directory	Contains the <code>current.macro</code> (the journal of commands issued within the last session of <i>fe-safe</i> )  Contains the <code>debug.log</code> , for diagnostic purposes.
projects directory	The default suggested location for <i>fe-safe</i> projects, though projects can be stored anywhere.
data directory (shortcut)	This is a shortcut back to the directory in the installation directory which contains all the data resources installed with <i>fe-safe</i> . This is useful when running through the tutorials.
gui.stli file	This file records all the gui settings for <i>fe-safe</i> (e.g. last browsing location for file dialogs). It is not recommended to edit this file directly.
user.stli file	This file records all of the user settings for <i>fe-safe</i> . Due to its strict XML structure, it is not recommended to edit this file directly. The preferred route is to use a macro to control the settings. It is also possible to use a default settings file ( <code>.stld</code> ) to control the settings at a site level.
user.lock, user.session file	For internal use, prevents concurrent use of a single user directory. Do not edit or remove unless advised to by <i>fe-safe</i> support.
master_node_settings.xml file	For DMP users only. A list of remote server hostnames and ports for DMP analysis.
*.dbase files	Any material database files that are found in the <code>&lt;install_directory&gt;/local</code> directory will be copied automatically to the user directory.

#### 205.92 *fe-safe* project directory

The *fe-safe* project (or project directory) records the FE model data and fatigue scenario for one fatigue analysis. It is made up of data, settings and log files, and can be transferred to other machines for execution.

It can be located either on a local drive or a network drive. The default location for projects is under the user’s home directory.

The project is divided into three parts; the model (data from the FE solution), the job (the fatigue scenario as defined by the user), and the results (generated history data and other diagnostics). The fatigue analysis has its own results directory under the job.

**Key files and directories in the project directory**

jobs directory	<p>This directory is future-proofed to allow for multiple jobs in a project. At present, only a single job per project is supported.</p> <p>Contains the <code>job.stli</code> (stores settings for the job, like material assignments, surface finishes, location of working files etc).</p> <p>Contains the <code>current.ldf</code> (fatigue loading definition for the job).</p> <p>Contains the <code>fesafe.fer</code> (contour results from the fatigue analysis).</p>
model directory	<p>Contains the FESAFE.FED directory (contains FE datasets read from the FE solution)</p> <p>Contains FESAFE.GRP (element/node group information read from the FE solution)</p> <p>Contains FESAFE.GEOM (mesh connectivity information read from the FE solution)</p> <p>Contains any modal displacements resulting from reading the FE solution.</p>
results directory	<p>Contains signals generated by the user using the safe4 signal processing functionality.</p>
project.stli file	<p>Contains settings related to FE interfaces, and model settings (e.g units)</p>

## 206 Appendix F - Interfacing to third-party products – data files

### 206.1 Third-party data file formats

*fe-safe* and *safe4fatigue* support the following third-party data file types:

- Servotest SBF and SBR files (\*.sbf, \*.sbr)
- Snap-Master file (\*.sm?)
- MTS RPCIII binary data file (\*.rsp)
- Adams multi-column ASCII tabular data (\*.tab)
- ANSYS Modal Coordinates File (\*.mcf)
- ASAM MDF4 binary data file (\*.mf4)

The interfaces to these file formats operate without conversion. In other words, no translation is required, *fe-safe* works directly from the data file.

*fe-safe* endeavours to maintain interface support to the latest versions of supported third-party data files.

### 206.2 Servotest binary files SBF and SBR files (\*.sbf, \*.sbr)

These files are used in conjunction with Servotest servo actuator equipment.

### 206.3 Snap-Master™ file (\*.sm?)

These data files are captured using Snap-Master™ data acquisition software. At present only the standard binary data file format for Snap-Master™ files is supported. The exponential and fast binary formats are not supported.

### 206.4 MTS RPCIII binary data file (\*.rsp)

*fe-safe* can import RPC3 data for plotting and analysis. Exporting to RPC3 data from other formats is not yet supported.

#### 206.4.1 Limitations on the types of RPC3 data that can be read:

- Data must be load histories. This is specified using the RPC3 header keyword:

```
FILE_TYPE=TIME_HISTORY.
```

Histogram data is not supported.

- The data must be either binary big-endian or binary little-endian. This is specified using the RPC3 header keywords:

```
FORMAT=BINARY
FORMAT=BINARY_IEEE_LITTLE_END
FORMAT=BINARY_IEEE_BIG_END
```

RPC3 ASCII data is not supported in this interface.

- Half frames and multiple partitions are not supported. This is specified using the RPC3 header keywords:

```
HALF_FRAMES=0
PARTITIONS=1
```

- Data must be stored in the short-integer format. This is the default type for RPC3 data, but can be specified using the RPC header keyword:

```
DATA_TYPE=SHORT_INTEGER
```

## 206.5 Adams multi-column ASCII tabular data (\*.tab)

*fe-safe* can import Adams multi-column ASCII tabular data for plotting and analysis. Exporting Adams ASCII data is not yet supported.

206.5.1 The following assumptions are made about the ADAMS tabular data format :

- The file is split up into groups of channels.
- Each group begins with the an ASCII header, the first line of which starts with the token:

```
`ADAMS/
```

This is followed by an N column list of data for the channels in the group.

- If the first channel is labelled TIME then the sample rate is evaluated using the data from the first channel, assuming that the time increments are equal between samples.
- The sequence of header followed by data repeats for each group of channels.
- The labels for each channel are extracted from the first token in line 5 of the group.
- Column headers are extracted from line 7 of the header.

Example of a channel group:

Line 1	"ADAMS/Car Assembly (v11.0.0 Patch APN-100-100) "				
Line 2	"2000-10-" "9 17:31:26"				
Line 3					
Line 4	"Result ID" "Result Type" "I Marker ID" "J Marker ID" "Result Title"				
Line 5	"Suspension_of_Horse_Drawn_Carriage" " Joint" " (N/A)" " (N/A)" " (No Comment)"				
Line 6					
Line 7	"TIME "	"FX "	"FY "	"FZ "	"TX "
Line 8	8.000000e-003	2.262210e+000	1.862700e+001	-4.923020e+001	-7.591220e+003
Line 9	1.600000e-002	2.718530e+001	2.746260e+001	-8.412050e+001	-6.250640e+003
Line 10	2.400000e-002	5.922910e+001	2.190590e+001	-1.385050e+002	-5.614000e+003

## 206.6 ANSYS Modal Coordinates File (\*.mcf)

These files are used for modal transient dynamic loading in conjunction with ANSYS FE data files.

## 206.7 • ASAM MDF4 binary data file (\*.mf4)

*fe-safe* can import MDF4 data for plotting and analysis. Exporting to MDF4 data from other formats is not supported.

206.7.1 Limitations on the types of MDF4 data that can be read:

- Data blocks must not be compressed.
- Data must not be in Array data blocks.
- Data must be stored in floating point or integer formats. String data or conversion formats are not supported.

## 207 Appendix G - Interfacing to third-party products – FE analysis data

### 207.1 Third-party FE file formats

#### 207.1.1 Supported FE file types

*fe-safe* can import Finite Element analysis data from the third-party FE results file types shown in *Table 207.1-1*.

When an FE results file is imported, *fe-safe* reads stresses, strains, temperatures, nodal forces and group information. The options available for each file type are shown in Table 207.1-1.

Interface options for each file type are available from the **FEA Fatigue** menu.

The user can specify whether or not to pre-scan the model file/s before importing model data:

**FEA Fatigue >> Analysis Options >> Interface >> FEA Model Read Options >>**  
**>> {Always pre-scan / Do not pre-scan / Prompt to pre-scan}**

If a file *is not* pre-scanned:

- Model data is imported according to the configuration on the **Interface** tab of the in the **Analysis Options** dialogue [**FEA Fatigue >> Analysis Options >> Interface >> FEA Model Read Options**]. For example: strains will be imported only if the **Read strains from FE Models** option is selected in the **Analysis Options** dialogue
- File type specific interface options are invoked according to the settings in the individual interface options dialogues.

If a file *is* pre-scanned:

- Default options in the pre-scan dialogue are configured as above, but...
- User selections made in the pre-scan dialogue *will override* the options in the individual interface options dialogues, where applicable.

Additional FE datasets can be appended to the loaded model (the FED file) using **File >> FEA Solutions >> Append FE Model**.

When a user selects **File >> FEA Solutions >> Append FE Model**, the appropriate import interface extracts model data from the specified model file/s and writes it to the *fe-safe* FED (finite element data) file (see Appendix E). This is a proprietary high-performance FE-type-independent working file containing the requested model data.

Where an FE file type supports more than one data type (e.g. element-nodal, integration points, etc.), the data type to be read is configured in the appropriate interface options dialogue, as discussed in the following sections.

*fe-safe* endeavours to maintain interface support to the latest versions of supported third-party FE packages.

FE results file type			Section	Versions supported		Data type <sup>9</sup>		Information type							
FE package	File type	File extension		WINDOWS	Linux		Elemental / Nodal	Stress	Strain* <sup>1</sup>	Temperature	Forces* <sup>7</sup>	Element Groups	Node Groups	Shells* <sup>6</sup>	Geometry* <sup>12</sup>
Abaqus	FIL	.fil	207.2			Element-nodal <sup>9</sup>	E	●	● <sup>2</sup>	● <sup>3</sup>	●	●	●	●	●
						Element centroidal	E	●	● <sup>2</sup>	● <sup>3</sup>		●	●	●	●
						Element integration points	E	●	● <sup>2</sup>	● <sup>3</sup>		●	●	●	●
						Nodal averaged	N	●	● <sup>2</sup>	● <sup>3</sup>	● <sup>11</sup>	●	●	●	●
	ODB output database <sup>15</sup>	.odb	207.3	versions 6.11 to 2017	versions 6.11 to 2017 (For earlier versions see note 12 below)	Element-nodal <sup>9</sup>	E	●	● <sup>2</sup>	● <sup>3</sup>	●	●	●	●	●
						Element integration points	E	●	● <sup>2</sup>	● <sup>3</sup>		●	●	●	●
						Element centroidal	E	●	● <sup>2</sup>	● <sup>3</sup>		●	●	●	●
						Nodal averaged	N	●	● <sup>2</sup>	● <sup>8</sup>	● <sup>11</sup>	●	●	●	●
ANSYS	RST (results file)	.rst	207.5	versions 5.5, 5.6, 6.x, 7.x, 8.x, 9.x <sup>10</sup> , 10.x <sup>10</sup> , 11.x, 12.x, 13.x, 14.x, 15.x, 16.x, 17.x	Element-nodal <sup>9</sup>	E	●	●	● <sup>4</sup>	●	●	●	●	●	
NASTRAN	f06 (PRINT) file	.f06	207.8	CSA/NASTRAN v98 and MSC.NASTRAN v4.5, v4.6, 2001 NX Nastran	Element-nodal <sup>9</sup>	E	●	●			●		●		
					Element centroidal	E	●	●			●		●		
	OP2 output file	.op2	207.7	MSC.NASTRAN v4.5, v4.6, 2001 NX Nastran	Element-nodal <sup>9</sup>	E	●	●		●	●		●	●	
	Element centroidal	E	●	●		● <sup>11</sup>	●		●		●		●	●	
SDRC I-DEAS	UNV	.unv	207.4	SDRC I-DEAS / Master Series versions 1 to 12 NX I-deas 5 to 6.1 (fe-safe detects the file version)	Element-nodal <sup>9</sup>	E	●	●	●	●	●		●	●	
					On elements	E	●	●	●		●		●	●	
					Nodal averaged	N	●	●	●	● <sup>11</sup>	●	● <sup>5</sup>	●	●	

Table 207.1-1



## \* Notes:

- 1 By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue or allow pre-scanning of files.
- 2 For Abaqus FIL and ODB files, the variable from which strain data is imported can be selected – see 207.2.3 and 207.3.5. It is particularly important to select alternative strain variables where a thermal strain may be included in total strain.
- 3 For elemental ODB and FIL files, temperatures are read from the TEMP variable – see 207.2.2 and 207.3.4 below.
- 4 For ANSYS RST files, reading temperature data is disabled by default. This significantly reduces the time taken to read the model – see section 207.5.2 below.
- 5 For I-DEAS UNV files, extra node groups based on the node colours are also extracted.
- 6 The way in which data from shell elements is handled varies between FE packages and FE package versions.
- 7 By default, *fe-safe* does not read force datasets by default. To read force datasets, select the **Read forces from FE models** option in the **Analysis Options** dialogue or allow pre-scanning of files.
- 8 For nodal ODB and FIL files, temperatures are read from the NT variable – see 207.2.2 and 207.3.4 below.
- 9 Where element-nodal data is available, this is the recommended data type.
- 10 The format of the \*.rst file changed from ANSYS v9 to v10, from v11 to v12 and from v12 to v13.
- 11 While forces can be loaded with stresses of data types other than Element-Nodal, forces are only ever loaded in as Element-Nodal.
- 12 For options on using older version of Abaqus \*.odb files see section 207.3.13 below.

## 207.12 Defining element and node groups

*fe-safe* uses the term “group” to describe either a list of element numbers (i.e. an ‘element group’) or a list of node numbers (i.e. a ‘node group’).

The type of group information extracted from the FE results file depends on the selected data type:

- if *fe-safe* is importing elemental datasets, only element group information is extracted from the FE results file;
- if *fe-safe* is importing nodal datasets, only node group information is extracted from the FE results file.

**Group names**

There are no limitations on group names. *fe-safe* reads the original group name from the model, then derives a group name for use in *fe-safe* by removing any spaces or illegal characters, then cropping the name if necessary so that it has no more than 30 characters. If this results in more than one instance of the same derived group name then *fe-safe* tags a number to the end of the group name to make it unique (see section 5.11).

The semantics used to describe element groups differ in different FE packages:

### Abaqus

In Abaqus, element groups are referred to as “Element Sets” and node groups are referred to as “Node Sets” – see sections 207.2.7 and 207.3.11 below.

### ANSYS

ANSYS does not export element and node groups directly to the RST file. Therefore, groups are supported in ANSYS by the use of the material number – see section 207.5.6 below.

### NASTRAN

In Nastran, groups are referred to as “Sets” and the context of referencing the sets indicates whether the sets are element or node sets. Pre-processors can be used to manage sets. See section 207.5.6 below.

## 207.2 Abaqus FIL (.fil) files

Options for importing and exporting Abaqus FIL (.fil) files are configured in the **Abaqus FIL Interface Options** dialogue – see Figure 207.2-1. These options may be overridden by selections made in the pre-scan dialogue.

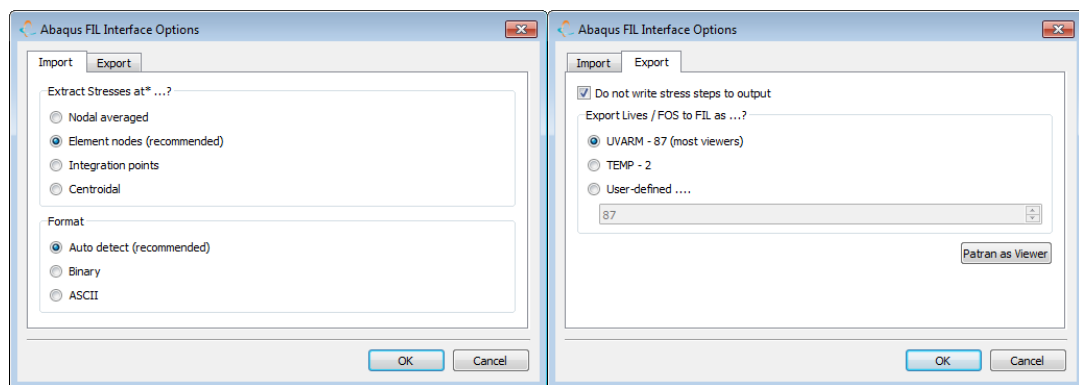


Figure 207.2-1

The ASCII FIL file format is portable between platforms. The binary FIL file format is not portable between platforms.

### 207.2.1 Reading stresses

*fe-safe* reads stresses from the Abaqus output variable **S**.

It is recommended that element-nodal stresses are used, since these stresses are on the surface of the component for elements that are on the surface.

However, stress datasets can be imported from any of the following data formats:

#### Element Nodes (recommended)

Sometimes referred to as element-nodal data, this is the recommended data format, since the stresses at the nodes are not averaged with data from adjacent elements. Also stresses are at the surface for nodes that are at the surface.

In Abaqus, stresses are extrapolated to each node in the element. There are  $n$  stress values per element; where  $n$  is the number of nodes in the element. Stresses at element nodes are created using the following line in the Abaqus input file:

```
*EL FILE, POSITION=NODES
```

### Nodal Averaged

Abaqus extrapolates stresses to each node in an element to give element-nodal stresses. The stresses calculated for the node by all elements that share the node are averaged to give a nodal-average stress. This data type is nodal. Nodal averaged stresses are created using the following line in the Abaqus input file:

```
*EL FILE, POSITION=AVERAGED AT NODES
```

### Integration Points

Stresses are calculated at the integration point of each element. Integration points are the default elastic-stress dataset type in Abaqus.

Integration point stresses are created using the following line in the Abaqus input file:

```
*EL FILE, POSITION=INTEGRATION POINTS
```

However, since integration points are the default elastic-stress dataset type in Abaqus, the `POSITION=` argument can be omitted. In other words, the line can be reduced to:

```
*EL FILE
```

### Centroidal

Stresses are calculated at the centroid of the element. Only one stress value is produced per element. Centroidal stresses are created using the following line in the Abaqus input file:

```
*EL FILE, POSITION=CENTROIDAL
```

## 207.2.2 Reading temperatures

Temperature datasets for elemental data, written using the Abaqus variable **TEMP**, and for nodal data, written using the Abaqus variable **NT**, can be imported. These datasets can be in the same file as the stress datasets, or they can be imported from a separate FIL file using the **File >> FEA Solutions >> Append Finite Element Model** option.

## 207.2.3 Reading strains

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets.

By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

### Strain variables

By default, *fe-safe* imports strains from the Abaqus output variable **E** (the "integrated" total strain) or **LE** (the logarithmic strain).

In some situations the strain is unlikely to be wholly a stress-related strain. For example in an elastic-plastic high temperature analysis, the temperature may cause a 'free expansion' which will contribute to the strain. In these situations, the strain can be imported as the sum of the Abaqus output variables **PE** (plastic strain) and **EE** (total elastic strains). This option is set by checking checkbox labelled **Extract Strains from (EE+PE) rather than E** in the **Abaqus FIL Interface Options** dialogue box – see Figure 207.2-1, above.

#### 207.24 Reading nodal forces

Nodal force datasets written using the Abaqus variable **NFORC**, can be imported. While forces can be loaded with stresses of data types other than element-nodal, the forces themselves are only ever loaded as element-nodal. These datasets can be in the same file as the stress datasets, or they can be imported from a separate FIL file using the **File >> FEA Solutions >> Append Finite Element Model** option.

#### 207.25 Supported element types

*fe-safe* requires that stress tensors read from an FE model to use a Cartesian coordinate system. This excludes the use of element types that use polar coordinate systems, e.g. beams and pipes.

Axisymmetric element types can be supported if their coordinates can be mapped to a Cartesian coordinate system. This can be achieved in Abaqus using the **\*ORIENTATION** command in the input deck.

#### 207.26 Using shells

*fe-safe* supports multi-layer shells imported from FIL files. A single analysis will deal with all shell section points.

#### 207.27 Element groups and node groups

In Abaqus, element groups are referred to as “Element Sets” and node groups are referred to as “Node Sets”.

Element sets are defined in Abaqus using the **\*ELSET** option, node sets are defined using the **\*NSET** option. For details of how to define element sets in Abaqus see Abaqus/Standard User’s Manual or Abaqus Keywords Manual.

Some element sets are assigned automatically by Abaqus - these are called ‘internal element sets’.

For Abaqus FIL files, groups defined in Abaqus using the **\*ELSET** and **\*NSET** keywords that are longer than 8 characters do not retain the group name, but are sequentially numbered.

#### 207.28 Writing fatigue results

Fatigue results are written to an Abaqus FIL file if the specified output file has the extension `.fil`.

When fatigue results are exported to a FIL file, the data from the source FIL file is exported to the output FIL file first. The original steps/increments can be omitted using the **Do not write stress steps to output** option. Then the fatigue results are appended as additional steps.

Fatigue results can be exported to any Abaqus element variable (See Abaqus/Standard User’s Manual). The first variable in the first results step appended to the output file will contain either:

- the fatigue life, or  $\log_{10}$ (fatigue life), for fatigue life analyses; or
- the fatigue reliability factor (FRF), when an FRF analysis has been performed.

The data written to subsequent variables may contain contours of the following fatigue results:

- the Factor of Strength (FOS), if a design life is specified;
- the Failure Rate or Reliability Rate if a target life is specified - a number of target lives can be specified so there may be more than one set of results.

Most viewers can successfully read the **UVARM** variable from an Abaqus FIL file. If your viewer cannot, then export the fatigue results to the **TEMP** variable or a user-defined variable.

If you are using Patran as your viewer, then for best results click the **Patran As Viewer** button. This sets the variable to **TEMP**, and also ensures that fatigue lives are saved as  $\log_{10}$ (life) values.

## 207.29 Export options

The following export options are available in the Abaqus FIL Interface Options dialogue:

**Export fatigue results to the UVARMS variable (variable=87)**

A single step is appended to the FIL file. The first **UVARM** variable will contain either fatigue life,  $\log_{10}$ (fatigue life) or Fatigue Reliability Factor contours. Additional **UVARM** variables may be appended containing Factor of Strength, Failure Rate or Reliability Rate contours. The list of variables is displayed in the message log as the file is being written. For example:

```
Variables as UVARMS
Adding step 3
Variable 1 : LOGLIFE
Variable 2 : FOS
Variable 3 : RR_at_life=5000
Variable 4 : RR_at_life=8000
Variable 5 : RR_at_life=10000
```

**Export fatigue results to the TEMP variable (variable 2), and****Export fatigue results to a user-defined variable**

One step will be appended to the FIL file that contains either the fatigue life,  $\log_{10}$ (fatigue life) or Fatigue Reliability Factor contours. Additional steps may be appended containing Factor of Strength, Failure Rate or Reliability Rate contours. The list of variables is displayed in the message log as the file is being written. For example:

```
Variables as Temperatures
Adding step 3
Variable 1 : LOGLIFE
Adding step 4
Variable 1 : FOS
Adding step 5
Variable 1 : RR_at_life=5000
Adding step 6
Variable 1 : RR_at_life=8000
Adding step 7
Variable 1 : RR_at_life=10000
```

## 207.2.10 Pre-Scanning Support

When pre-scanning is used on a file the **Extract Stresses** option is ignored. The position is chosen in the **Select Datasets to Read** dialogue.

Note that the legacy interface does not pre-scanning.

## 207.3 Abaqus Output Database ODB (. odb) files

Options for importing and exporting Abaqus output database ODB (. odb) files are configured in the **Abaqus ODB Interface Options** dialogue – see

Figure 207.3-1. These options may be overridden by selections made in the pre-scan dialogue.

The Abaqus version that was used to create the source ODB file will be detected by *fe-safe* and the appropriate ODB interface will be automatically selected.

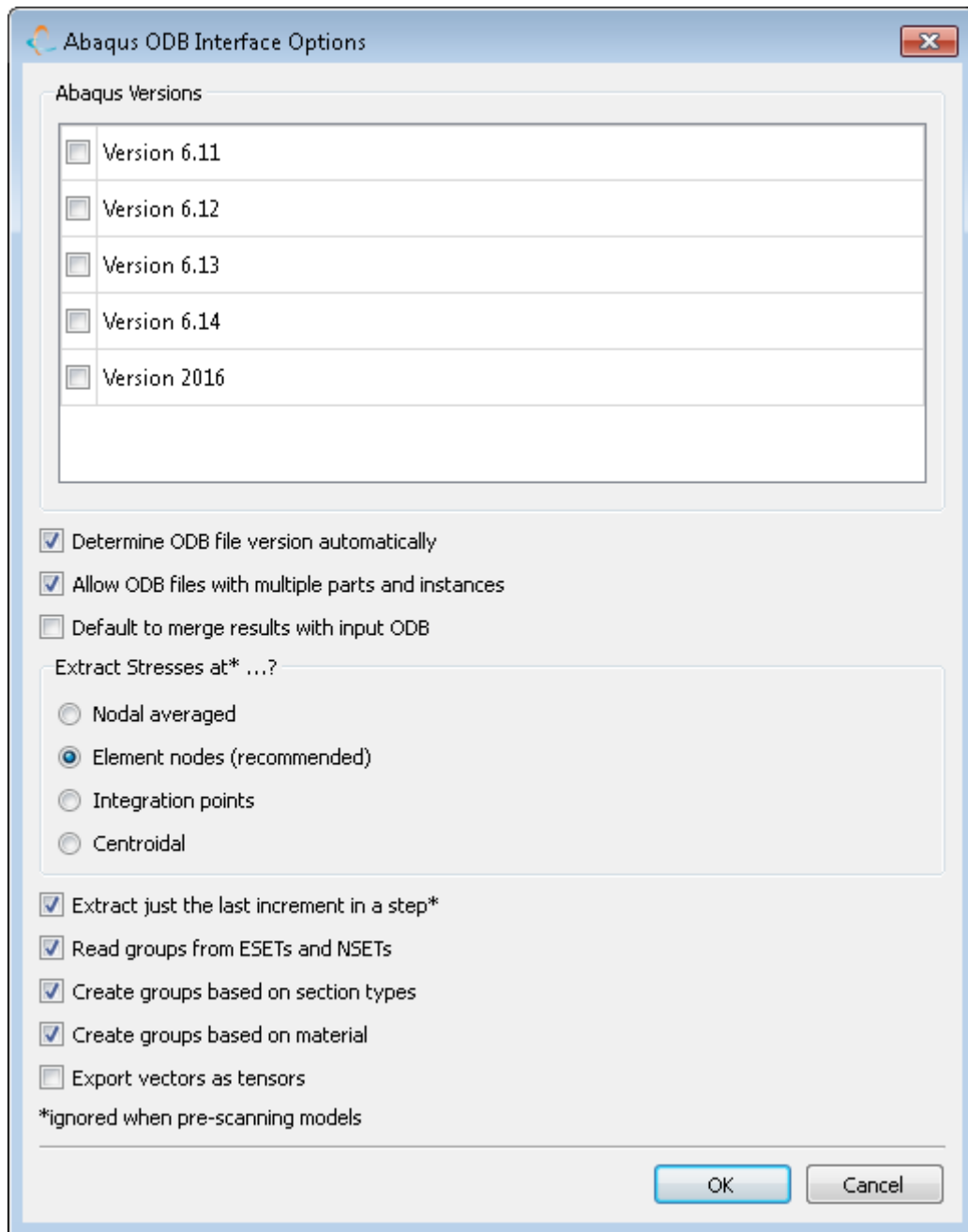


Figure 207.3-1

### 207.3.1 Supported Data Types

The ODB interface in fe-safe supports the following data types for tensor and scalar quantities:

- Integration points (elemental)
- Element-nodal (elemental)
- Centroidal (elemental)
- “Nodal-averaged”, or “averaged-at-nodes” (nodal)

Where an ODB file does not contain element-nodal, centroidal or nodal-averaged data, these can be derived automatically from integration point datasets. Both tensors and scalar quantities can be derived from integration point datasets.

The pre-scan dialogue window will show both:

- Datasets that already exist in the ODB file.
  - These can be loaded directly into fe-safe.
  - Variables that can currently be loaded directly into fe-safe include:
    - **S, E, LE, {EE and PE}, NE, TEMP, NT, NFORC**
- Datasets that can be derived from integration point data.
  - These datasets are clearly suffixed in the pre-scan dialogue with the string: “(derived)” - see Figure 207.3-2.
  - If a “derived” dataset is selected, the conversion process is performed as the model is being imported.
  - The following variable can currently be derived:
    - **S, E, LE, {EE and PE}, NE**
  - Currently it is not possible to derive nodal-averaged **TEMP** from integration points. The variable **NT** should be requested directly instead.

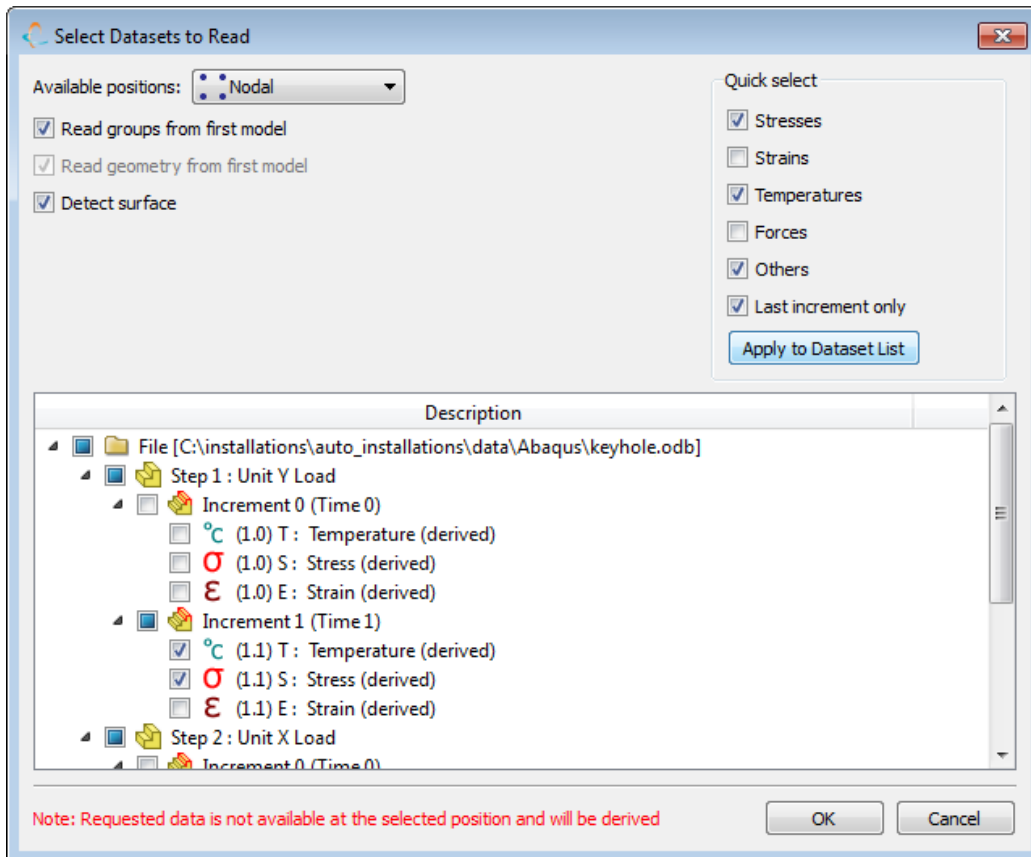


Figure 207.3-2

Note that Abaqus/Viewer currently cannot display unique (averaged) tensors at nodes. However, this limitation does not apply to fe-safe fatigue results as they are exported as scalar values.

Nodal-averaged data is derived by averaging contributions from multiple elements to create a single unique tensor or scalar quantity at the node. A node belonging to multiple elements where the elements have different properties may have multiple instances/values.

### 207.3.2 Working with nodal-averaged ODB data

Nodal-averaged data is supported in one of two ways:

**1. By importing nodal-averaged data directly from an ODB file.**

Nodal averaged data is currently not available as a standard ODB output in Abaqus. However, an ODB API utility has been published in the Simulia Online Support System (SOSS) allowing users to create ODB files containing nodal-averaged data. For more details please see Simulia Online Support System (SOSS) - Answer 4241. Note that the element types and data types supported by this utility are not comprehensive.

**2. By deriving nodal-averaged data from integration point data.**

If the ODB file does not contain nodal-averaged data, then fe-safe can derive nodal-averaged data from integration point data as described in section 207.3.1 above.

### 207.3.3 Reading stresses

It is recommended that element-nodal stresses are used, since these stresses are on the surface of the component for elements that are on the surface.

However, stress datasets can be imported from any of the following data formats:

#### Element Nodes

Stresses are extrapolated to each node in the element. There are  $n$  stress values per element; where  $n$  is the number of nodes in the element. Stresses at element nodes are created using the following line in the input file:

```
*ELEMENT OUTPUT, POSITION=NODES
```

#### Nodal Averaged

Stresses are extrapolated to each node in an element to give element-nodal stresses. The stresses calculated for the node by all elements that share the node are averaged to give a nodal-average stress. This data type is nodal.

See section 207.3.2 above

#### Integration Points

Stresses are calculated at the integration point of each element. Integration points are the default elastic-stress dataset type in Abaqus.

Integration point stresses are created using the following line in the Abaqus input file:

```
*EL FILE, POSITION=INTEGRATION POINTS
```

However, since integration points are the default elastic-stress dataset type in Abaqus, the `POSITION=` argument can be omitted. In other words, the line can be reduced to:

```
*EL FILE
```

#### Centroidal

Stresses are calculated at the centroid of the element. Only one stress value is produced per element. Centroidal stresses are created using the following line in the Abaqus input file:

```
*ELEMENT OUTPUT, POSITION=CENTROIDAL
```

### 207.3.4 Reading temperatures

Temperature datasets for elemental data, written using the Abaqus variable **TEMP**, and for nodal data, written using the Abaqus variable **NT**, can be imported. These datasets can be in the same file as the stress datasets, or



they can be imported from a separate ODB file using the **File >> FEA Solutions >> Append Finite Element Model** option.

See also section 207.3.2 above

### 207.3.5 Reading strains

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets.

By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

#### Strain variables

By default, *fe-safe* imports strains from the Abaqus output variable **E** (the "integrated" total strain) or **LE** (the logarithmic strain).

In some situations the strain is unlikely to be wholly a stress-related strain. For example in an elastic-plastic high temperature analysis, the temperature may cause a 'free expansion' which will contribute to the strain. In these situations, the strain can be imported as the sum of the Abaqus output variables **PE** (plastic strain) and **EE** (total elastic strains). This option is set by checking checkbox labelled **Extract Strains from (EE+PE) rather than E** in the

**Abaqus ODB Interface Options** dialogue box – see

Figure 207.3-1 above.

It is also possible to read in the nominal strain **NE** for use in elastomer fatigue analysis with *fe-safe*/Rubber. This option is only available through the pre-scan dialogue.

### 207.3.6 Reading nodal forces

Nodal force datasets written using the Abaqus variable **NFORC**, can be imported. While forces can be loaded with stresses of data types other than element-nodal, the forces themselves are only ever loaded as element-nodal. These datasets can be in the same file as the stress datasets, or they can be imported from a separate ODB file using the **File >> FEA Solutions >> Append Finite Element Model** option.

### 207.3.7 Increments and steps

Data can be imported for all increments within a step or just for the last increment. This option is set in the **Abaqus ODB Interface Options** dialogue.

### 207.3.8 Load-cases

If a step uses the load-case feature in Abaqus rather than increments then the last increment in step parameter is ignored and all of the load-cases within the step are extracted. The increment number will be marked as -1 if a dataset was extracted from a load-case step.

### 207.3.9 Supported element types

*fe-safe* requires that stress tensors read from an FE model use a Cartesian coordinate system. This excludes the use of element types that use polar coordinate systems, e.g. beams and pipes.

Axisymmetric element types can be supported if their coordinates can be mapped to a Cartesian coordinate system. This can be achieved in Abaqus using the **\*ORIENTATION** command in the input deck.

### 207.3.10 Using shells

*fe-safe* supports multi-layer shells imported from Abaqus ODB files.

Export of fatigue results to multiple section points is only supported in some releases of the ODB interface. If it is not supported then each set of results (LOGLife, FOS, etc.) will be written to the ODB step as separate variables - one for each section point. The name of the variable is constructed from the fatigue variable name and the section point name. For example:  $\log_{10}$ (fatigue life) will be written to the variables:

LOGLife\_SNEG(-1) and LOGLife\_SPOS(1)

### 207.3.11 Element groups and node groups

In Abaqus, element groups are referred to as “Element Sets” and node groups are referred to as “Node Sets”.

Element sets are defined in Abaqus using the \*ELSET option, node sets are defined using the \*NSET option. For details of how to define element sets in Abaqus see Abaqus/Standard User’s Manual or Abaqus Keywords Manual.

Some element sets are assigned automatically by Abaqus - these are called ‘internal element sets’.

Groups extracted from Abaqus ODB files will retain the group name in *fe-safe*, up to a maximum of 30 characters. Group names longer than 30 characters will be truncated (see section 207.1.2 above).

If a group name is not unique after truncation *fe-safe* will create a unique name for the group.

### 207.3.12 Writing fatigue results

Fatigue results are written to an Abaqus output database (ODB) file if the specified output file has the extension .odb.

If the specified output ODB file has the same name as the source database (input ODB file) then a new step is added to the source database.

If the specified output ODB file does not have the same name as the source database then:

- if the output ODB file does not exist the source database is copied to the output ODB file, and a new step is added containing the fatigue results;
- if the output ODB file already exists then the user is prompted (see Figure 207.3-3) to choose to either overwrite the output ODB file with a copy of the source database before appending the fatigue results, or append the fatigue results to the existing output ODB.

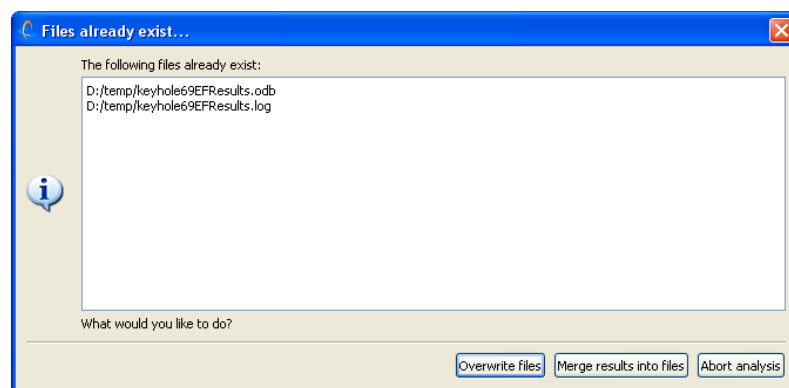


Figure 207.3-3

The new results step will have the name ‘fe-safe\_01’. However if a step already has that name, *fe-safe* will search for ‘fe-safe\_02’ then ‘fe-safe\_03’ and so on until an unused name is found.

The first results set that is written to the output file will contain either:

- the fatigue life, or  $\log_{10}$ (fatigue life), for fatigue life analyses; or
- the fatigue reliability factor (FRF), when an FRF analysis has been performed.

Subsequent results sets may contain contours of the following fatigue results:

- the Factor of Strength (FOS), if a design life is specified;
- the Failure Rate or Reliability Rate if a target life is specified - a number of target lives can be specified so there may be more than one set of results.

The list of results being written to the ODB file is displayed in the message log as the file is being written. For example:

```
Results File : c:\safeResultsArchive\fesafe.fer
contains    : LOGLife-Repeats
contains    : FOS@Life=1E7-Repeats
contains    : RR_at_life=500 (RR_at_life=500)
contains    : RR_at_life=800 (RR_at_life=800)
contains    : RR_at_life=1000 (RR_at_life=1000)
ODB Database : p:\data\fullmodeltests\curResults\shaft.odb
Data at     : Element Nodal
```

**Note:** The following message may appear in the log if an attempt is made to export results for only a single node to the ODB file:

```
For element 1127 there are only 1 (out of 20 expected) values, -1 will be used.
There were 19 missing values when exporting contour LOGLife-Repeats. The missing
values have defaulted to -1.
```

This message can be ignored. Alternatively, select:

**FEA Fatigue >> Analysis Options >> Interface (tab) >> GUI Options >> Prompt before exporting results**

and ensure that at the prompt the results are not exported if only a single node has been analysed.

### 207.3.13 Supporting older versions.

The most recent versions of the Abaqus ODB interface are supported by default on each platform – the required interface files are supplied with the software. Because of the size of the run-time libraries required for each interface version, some of the older interfaces are not shipped with the software.

For interfacing to ODB files from Abaqus versions earlier than the interfaces shipped with the software, please contact your local support office to discuss the options available.

### 207.3.14 Support for parts and instances.

Parts and instances from ODBs are now fully supported by default. Loading in an ODB will now list the instance names in the **Current FE Models** window under the assembly item (see Figure 207.3-4 Display of parts and instances in the model tree)

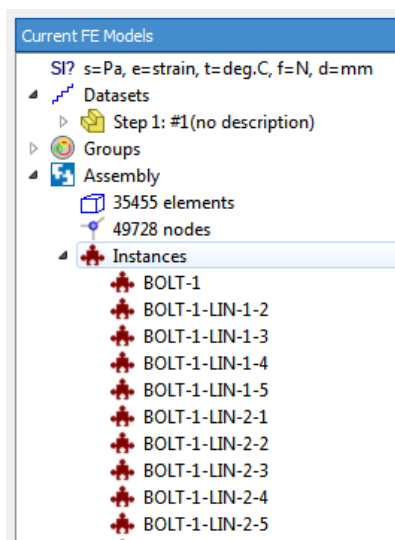


Figure 207.3-4 Display of parts and instances in the model tree

To specify a specific instance, prefix the item ID with ["NAME"] where NAME is the instance name (see Figure 207.3-5). This is available in all places where an item can be specified, e.g. creating groups in the **Group Manager**, the **List of items** export and with virtual strain gauges. If no instances is specified all elements/nodes from all instances will be affected.

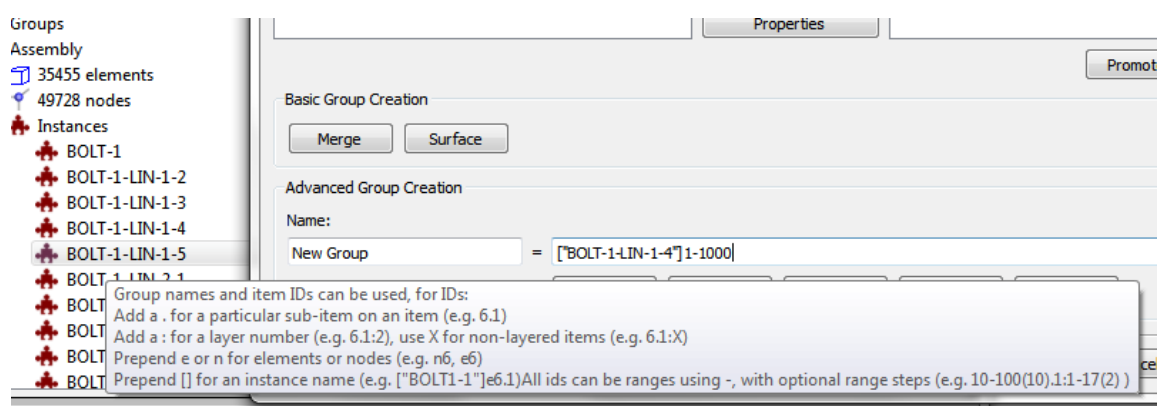


Figure 207.3-5 Group control for parts and instances

When saving a loaded ODB to a text file, to reduce space a table at the top of the file lists all instance and all further references are an index into this table e.g. [0]100 instead of ["BOLT 1"]100.

**Note:** FED files from previously loaded versions of fe-safe are not compatible with appending ODBs or exporting analysis results back to an ODB while ODB instances are enabled. To disable support for ODB instances uncheck the **Abaqus ODB Interfaces Options** checkbox labelled **Allow ODB files with multiple parts and instances**.

### 207.3.15 Pre-Scanning Support

When pre-scanning a file the **Extract Stresses** option and the extracting just the last increment are ignored. The position and increments are chosen in the **Select Datasets to Read** dialogue.

One of the indicators that a pre-scan is valid is that the time stamp on the source file has not changed. However with ODB files the time stamp changes when the file is read, this is a consequence of using the Abaqus ODB API. For this reason the user is asked if they wish to re-scan an ODB file. There is an option on this prompt to apply the users choice for future ODB reads. To re-enable prompting select the **Always prompt user for action confirmation** option in the **Tools >> Options** menu.

## 207.4 I-DEAS UNV (.unv) files

*fe-safe* will read UNV (.unv) files from I-DEAS v6, Master Series, and I-DEAS v7 to v10 (*fe-safe* automatically detects the version).

Options for importing and exporting UNV (.unv) files are configured in the **I-DEAS / Master Series UNV Interface Options** dialogue – see Figure 207.4-1 below. These options may be overridden by selections made in the pre-scan dialogue.

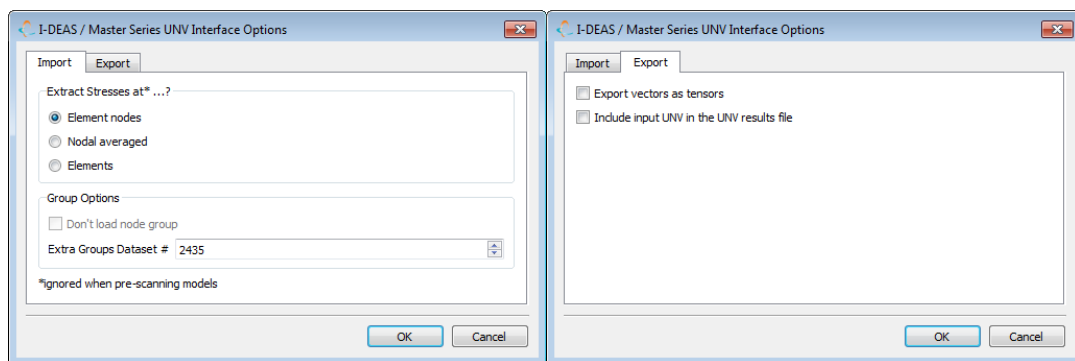


Figure 207.4-1

### 207.4.1 Reading stresses, strains and temperatures

Three types of stresses are supported:

- *Element Nodes*: These are stresses arranged in elements where each node in an element has its own set of stresses. Described in the I-DEAS documentation for dataset 2414 as '3: Data at nodes on elements'.
- *Nodal Averaged*: These are stresses averaged at each node. This option is only supported for Master Series 6 and I-DEAS v7 and above. Described in the I-DEAS documentation for dataset 2414 as '1: Data at nodes'.
- *Elements*: These are a single set of stress tensors per element. This option is only supported for Master Series 6 and I-DEAS v7 and above. Described in the I-DEAS documentation for dataset 2414 as '2: Data on elements'.

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets.

By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

Universal Dataset Numbers 57 and 2414 are supported in the current release of *fe-safe*. These records are documented in the I-DEAS help library. To locate them from the Bookshelf page of the help go to 'Miscellaneous' (bottom left) and then select 'File formats reference guide'. The universal datasets are documented in numerical order.

In order for a UNV file to be read successfully into *fe-safe*, the analysis data must meet the following requirements:

Item	Requirement	Dataset 2414 position
Dataset location	1:Data at nodes, 2:Data on elements or 3: Data at nodes on elements	Record 3, field 1
Model Type	1:Structural	Record 9, field 1
Analysis Type	1:Static, 4:Transient, or 9:Static non-linear	Record 9, field 2
Data characteristic	4: Symmetric global tensor	Record 9, field 3
Result type	2:Stress, 3:Strain or 5:Temperature	Record 9, field 4

The option to read UNV models with the legacy interface is for use as a reference only and will shortly be removed in a future release.

#### 207.4.2 Supported element types

*fe-safe* supports solid, 2D and shell elements - beam elements are ignored by *fe-safe*. This in practice is checked by confirming that there are 6 or 12 (for shells) stress or strain values per node.

#### 207.4.3 Using Shells

I-DEAS 2-layer shells are supported fully for Master Series 6 and I-DEAS v7 and above.

#### 207.4.4 Permanent Groups

Groups are extracted from the Permanent Groups Universal Datasets, at this release the supported dataset numbers are 752, 2417, 2429, 2430, 2432, 2435, 2452, and 2467. These records are documented in the I-DEAS help library. To locate them from the Bookshelf page of the help go to 'Miscellaneous' (bottom left) and then select 'File formats reference guide'. The universal datasets are documented in numerical order.

Two entity types are supported for these records '7: Nodes' and '8: Finite Elements'. If the stresses being read are on elements then the finite element permanent groups are read, if the stresses being read are nodal then the nodal permanent groups are read.

These records tend to become obsolete frequently so the **Extra Groups Dataset #** field as shown in figure 207.4-1 allows one extra dataset number to be processed. This will allow future versions of I-DEAS permanent groups to be temporarily supported if necessary. The new dataset must have the same format as Universal Dataset 2435; this is the case for all datasets that have so far superseded it.

If you have successfully used a new dataset type that is not in the existing list (above) then please inform Safe Technology Limited so that it can be permanently implemented in the code.

## 207.4.5 Node Colours

For nodal stresses extra groups based on the node colours are also extracted. The node's colours are extracted from Universal Dataset 2411. This record is documented in the I-DEAS help library. To locate them from the Bookshelf page of the help go to 'Miscellaneous' (bottom left) and then select 'File formats reference guide'. The universal datasets are documented in numerical order.

In the dataset record field 4 is a colour index into the table:

Index	Colour	Index	Colour
1	Blue	2	Gray_Blue
3	Light_Blue	4	Cyan
5	Dark_Olive	6	Dark_Green
7	Green	8	Yellow
9	Golden_Orange	10	Orange
11	Red	12	Magenta
13	Light_Magenta	14	Pink
15	White		

## 207.4.6 Writing fatigue results

To export the results as a UNV file specify the output filename with a `.unv` extension. Note the previously the UNV export interface wrote each contour to a separate variable, if this behaviour is required use the legacy interface mentioned below.

Note that some contours will be exported as a vector to the UNV file, e.g. 'Critical planes (all blocks)'. This can be changed to output the vector as a tensor instead, by selecting the correct option in the UNV interface options.

A legacy UNV write interface is provided as a reference, and will be removed in a future version. The legacy version does not support export of vector contours and each component of vector will be exported separately. Also multiple results in one file is not supported – the first calculated variable will be written to the specified output file, either the lives, log of lives or the FRF. If a design life was specified then this will be written to a file built up of the same filename as the lives with `fos` appended to it. If failure rates are calculated they will be written to files with the same name with `FR_at_life_xx` appended to it, (where `xx` is the design life).

*e.g. For an analysis with a FOS, and reliability rates calculated at 1000, 2000 and 5000 hours the following files will be created:*

```

Output file:    lives.unv
FOS file:      livesfos.unv
FRF files:     lives_FR_at_life_1000.unv
                lives_FR_at_life_2000.unv
                lives_FR_at_life_5000.unv

```

### 207.4.7 Pre-Scanning Support

When pre-scanning is used on a file the **Extract Stresses** option is ignored as the position is chosen in the **Select Datasets to Read** dialogue.

### 207.5 ANSYS RST (.rst) files

Options for importing ANSYS RST files are configured in the **ANSYS RST Interface Options** dialogue – see Figure 207.5-1. These options may be overridden by selections made in the pre-scan dialogue.

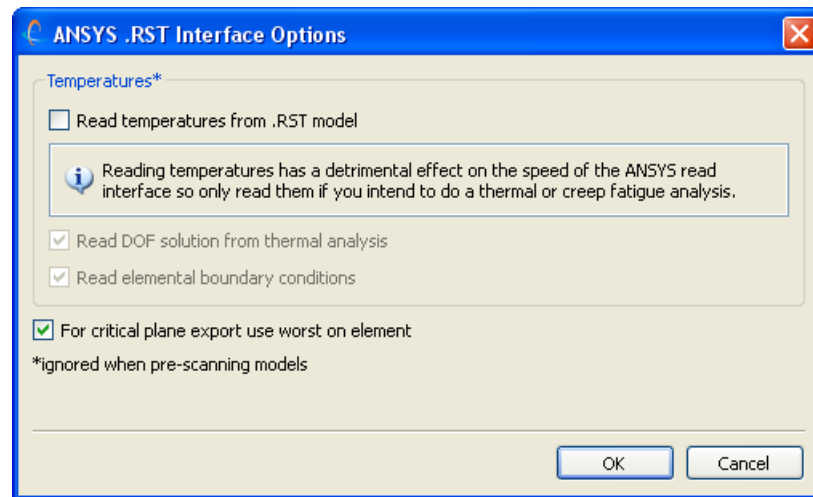


Figure 207.5-1

#### Note 1:

The ANSYS keyword `NO_RSTGM` (in the ANSYS configuration file: `config70.ans`), allows the ANSYS user to suppress writing the model geometry to the RST file (`NO_RSTGM =1`). RST files created using this option are not currently supported by *fe-safe*. For more information, see: ANSYS documentation [ANSYS 7.0 Basic Guide >> Chapter 19 - Memory Management and Configuration >> 19.4 - The Configuration File].

#### Note 2:

ANSYS includes a user option that allows files over a certain size to be split into sections. This option is not currently supported in *fe-safe*. For more information see: ANSYS documentation ANSYS 7.0 Operations >> Chapter 6. Customizing ANSYS and the GUI >> 6.2 Splitting files across platforms.

### 207.5.1 Large file support

All file sizes are supported directly.

### 207.5.2 Reading stresses and temperatures.

The required files from ANSYS are the structural results (.rst) and the thermal results (.rth) files. Opening either of these files will extract the temperatures and stresses contained in them.

By default, reading temperature data is disabled, as this speeds up read times significantly. If temperature data is required set the **Read Temperatures from .RST model** in the **ANSYS RST Interface Options** dialogue. Two types of temperatures are stored in the .rst file. DOF solution temperatures and boundary condition temperatures, reading of either can be disabled or enabled as required.



### 207.5.3 Reading strains

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets.

By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

### 207.5.4 Supported Element Types

The following element types are currently supported by *fe-safe*:

Type	No.	Type	No.	Type	No.	Type	No.	Type	No.
PLANE	2	PLANE	13	PLANE	25	PLANE	35	PLANE	42
PLANE	53	PLANE	55	PLANE	67	PLANE	75	PLANE	77
PLANE	78	PLANE	82	PLANE	83	PLANE	121	PLANE	145
PLANE	146	PLANE	162	PLANE	182	PLANE	183	PLANE	223
PLANE	230	SHELL	28	SHELL	41	SHELL	43	SHELL	51
SHELL	57	SHELL	61	SHELL	63	SHELL	91	SHELL	93
SHELL	99	SHELL	<b>131</b>	SHELL	<b>132</b>	SHELL	143	SHELL	150
SHELL	157	SHELL	163	SHELL	181	SHELL	208	SHELL	209
SHELL	281								
SOLID	5	SOLID	45	SOLID	46	SOLID	62	SOLID	64
SOLID	65	SOLID	69	SOLID	70	SOLID	72	SOLID	73
SOLID	87	SOLID	90	SOLID	92	SOLID	95	SOLID	96
SOLID	97	SOLID	98	SOLID	117	SOLID	122	SOLID	123
SOLID	<b>127</b>	SOLID	<b>128</b>	SOLID	147	SOLID	148	SOLID	164
SOLID	168	SOLID	185	SOLID	186	SOLID	187	SOLID	191
SOLID	226	SOLID	227	SOLID	231	SOLID	232	SOLSH	190

### 207.5.5 Using shells

ANSYS multi-layer shells are treated as elements with extra nodes.

In ANSYS, shells have their corner stresses exported. So, for a 4 node element the top shell will have nodes numbered 1 to 4, and the bottom shell will have nodes numbered 5 to 8.

### 207.5.6 Groups

ANSYS does not export element and node groups directly to the RST file. Therefore, groups are supported in ANSYS by the use of the material number. To define element groups set different regions in the model to have different material numbers (the material data can be the same).

### 207.5.7 Writing fatigue results

Only the fatigue results are exported to the .rst file.

To export the results as an ANSYS file specify the output filename with a .rst extension. The source model will be copied to the destination file and then a new set of structural results containing the fatigue results will be added. The set number, the set name, and the variables exported are written in the Message Log as the results are exported:

```
Adding fatigue parameters ...
Exporting set :      2
Variable      : LOGLife-Repeats
as           : Sx
Variable      : FOS
as           : Sy
Variable      : RR_at_life=500
as           : Sz
Variable      : RR_at_life=800
as           : Sxy
Variable      : RR_at_life=1000
as           : Syz
Variable      : RR_at_life=2000
as           : Sxz
Set title     : fe-safe x.yy[mswin];test1.ldr;e-p FEA;BM-Mo;SAE1020
```

The fatigue results are stored in the stress tensor. Sx is the fatigue life, the log of the life or the FRF, and, if available, Sy is the factor of strength or the first failure rate, and so on.

### 207.5.8 Pre-Scanning Support

When pre-scanning is used on a file the **Temperatures** option is ignored. Both temperature types will be selectable in the **Select Datasets to Read** dialogue.

Normally each solution in the \*.rst file is parsed until an element is found with stresses, strains, temperatures or a combination there of. If this behaviour is causing datasets to be missed, the system keyword PS\_QUICK\_SCAN can be used to add all three dataset types without scanning for elements.

### 207.6 NASTRAN .op2 files

*fe-safe* will read analysis results from MSC.NASTRAN and NX NASTRAN. Some packages allow .op2 files to be created without header information such as the modify date. These files are supported though a header will be written when results are exported to an .op2 file..

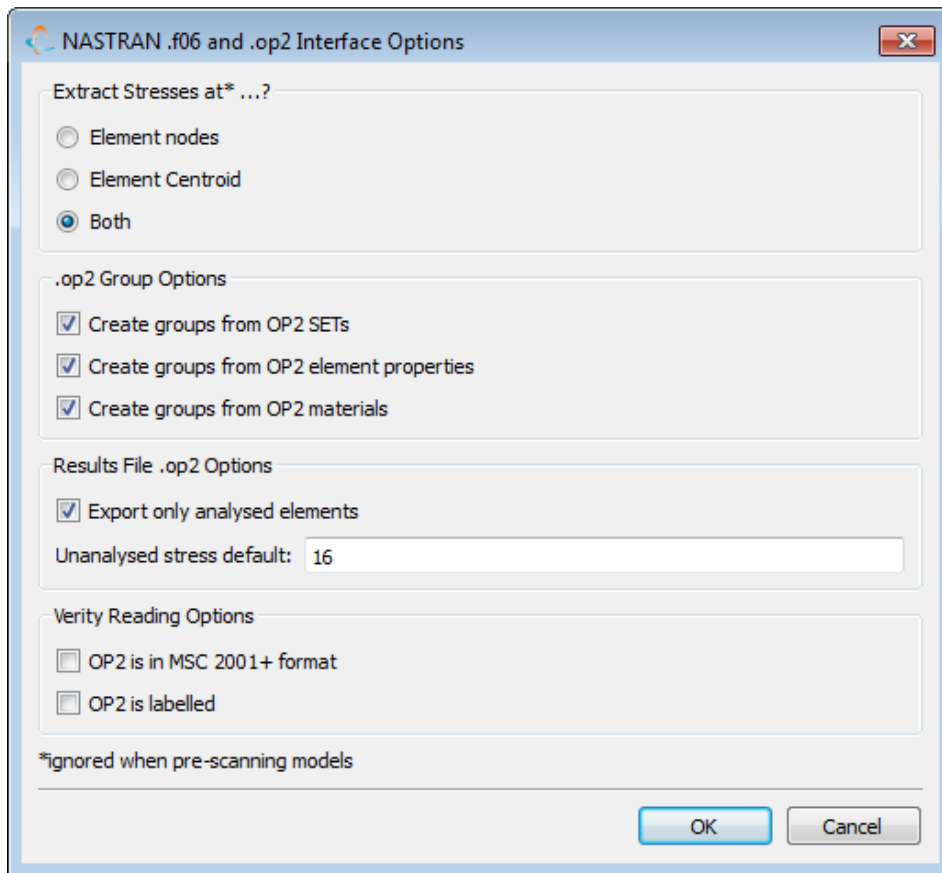


Figure 207.6-1

### 207.6.1 Reading stresses

NASTRAN refers to datasets as subcases.

The required binary file from NASTRAN is the `.op2` file. For importing into *fe-safe* the `.op2` file must include the elastic stresses for each data set. *fe-safe* will attempt to load stress data for all subcases that contains compatible stress data.

Three stress types can be loaded from the `.op2` file:

- *Element nodes*: the stresses for each node are read and stored with the element number.
- *Element Centroidal*: only the centroidal stresses are read and stored with the element.
- *Both*: both the centroidal and nodal stresses are read and stored with the element number.

**Note:** *fe-safe* only supports stresses written in sort mode `SORT1`.

### 207.6.2 Reading temperatures

Temperature datasets are not supported in NASTRAN.

### 207.6.3 Reading strains

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets. By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

## 207.64 Supported Element Types

*fe-safe* supports solid, 2D and shell elements - beam elements are ignored by *fe-safe*. The following table shows supported element types:

Element Type ID	Topology	Description
1	1 Node, (X & XY)	Rod, for stresses or strains
3	1 Node, (X & XY)	Tube, for stresses or strains
10	1 Node, (X & XY)	Rod, for stresses or strains
33	1 Node, (X, Y & XY)	Shell, for stresses or strains
39	4 Nodes, centre, (X, Y, Z, XY, XZ && YZ)	Tetrahedron, for stresses or strains
60	1 Node, (X, Y & XY)	2D crack tip element
61	1 Node, (X, Y, Z, XY, YZ, XZ)	3D crack tip element
64	4 Nodes, centre, (X, Y & XY)	Curved quad shell, for stresses or strains
67	8 Nodes, centre, (X, Y, Z, XY, XZ && YZ)	Hexa
68	6 Nodes, centre, (X, Y, Z, XY, XZ && YZ)	Penta
70	3 Nodes, centre, (X, Y & XY)	Triangular shell plate, for stresses and strains
74	1 Node, (X, Y & XY)	Shell corner, for stresses and strains
75	3 Nodes, centre, (X, Y & XY)	Curve triangular shell, for stresses and strains
82	4 Nodes, centre, (X, Y & XY)	Quad shell plate, for stresses and strains
139	4 Nodes, (X, Y & XY)	Hyper-elastic quad, for stresses. Requires both elemental and centroidal import option on
144	4 Nodes, centre, (X, Y & XY)	Quad shell plate, for stresses and strains

For further information on NASTRAN element types, see MSC.NASTRAN 2001 documentation.

## 207.65 Using shells

Two-layer shells are supported.

## 207.66 Groups

Groups are supported in NASTRAN. NASTRAN refers to groups as SETS, which are extracted from the first case control record in the case control data block in the .op2 file. Any additional case control records are ignored.

## 207.67 Writing fatigue results

When a NASTRAN .op2 file is loaded, the output type defaults to .op2 for export to viewers such as *FEMAP*. The exported results will be written as stress or strain tensors in a subcase, lives will always be written to **X Normal**. Any other results exported (e.g. factor of strength) will be written to other components, depending on which elements are used. In the event there are more results than available tensor components, a new subcase will be

created. All mappings are displayed in the log file for the analysis similarly to the following extract from the keyhole tutorial log:

```
Results File : c:\safeResultsArchive\fesafe.fer
  contains   : LOGLife-Repeats
  contains   : FOS@Life=1E7-Repeats

Creating subcase 1 for 2 results

LOGLife-Repeats will be mapped to Normal X
FOS@Life=1E7-Repeats will be mapped to Normal Y
LOGLife-Repeats will be mapped to Normal X
FOS@Life=1E7-Repeats will be mapped to Normal Y

...results transfer complete.
```

There are two options when writing fatigue results to an .op2 file. The check box **Export unanalysed elements** determines whether elements present in the source .op2 file that were not analysed will still be exported with values. The default value can be defined using the **Unanalysed stress default** edit control.

Unanalysed elements include :

- excluded groups;
- those without stresses.

When only the elements' central stress is read from the source file, the exported centroidal result will be copied to the nodal stress values.

When only the element nodal stresses are read from the source file, the exported nodal results will be averaged and exported to the centroidal stress.

For convenient viewing, the analysis results should normally be exported as log(10) of lives. This option can be set in the **General FE Options** dialogue box.

To view the results of a fatigue analysis in FEMAP:

- Start FEMAP, and load the model file by clicking **File >> Open...**, choosing **NASTRAN** from the **Files of type** drop-down list and select the file (probably .nas or .dat).
- Create a new window for displaying the analysis results. From the **New View** dialogue box click **View >> New...**
- Show the Import Results from dialogue box by clicking **File >> Import >> Analysis Results....**
- In the **Import results from dialogue** box select **MSC.NASTRAN** from the **NASTRAN** drop-down list.
- Select the .op2 file produced by *fe-safe* and click **open**.
- Show the View Select dialogue box by clicking **View >> Select**.
- Set the deformed style to **None - Model Only**, and the contour style to **Contour**.
- Click on **Deformed and Contour Data**.
- In the **Select PostProcessing Data** dialogue box:

- select the desired subcase from the **Output Set** drop-down list (the name of the set will be X..MSC/NASTRAN Case Y, where X is a number and Y is the subcase). This will normally be the last option in the list.
- select **4..Stress** (or strain is applicable) from the **Category** drop-down list
- select from the **Contour** drop-down list the following:
  - either **Plate Top** or **Plate Bot** to view shell elements, or **Solid** to view solids
  - either **X, Y** or **Z Normal** or the shears depending on the outputs selected
- click **OK**.
- Click **OK** in the **View Select** dialogue box - the contour will plot.

*Tip:* When viewing the fatigue life contours, the areas of the component of most interest are those with the highest probability of failure (i.e. the areas that have the shortest lives). If the contour plot is using FEMAP's default colour palette, the areas of shortest life will be highlighted in blue, which can lead to confusion. Reverse the colours of the contour levels to highlight the areas of shortest life in red by:

- selecting the **View Options** window, by clicking **View >> Options...**;
- select the **Post Processing** option called **Contour/Criteria levels**;
- click **Set Levels...**;
- in the **Contour/Criteria Levels** dialogue box, click on **Reverse**, then **OK**;
- in the **View Options** dialogue box, click **Apply**, then **OK**.

## 207.6.8 Pre-Scanning Support

When pre-scanning is used on a file the **Extract Stresses** option is ignored. The position is selected in the **Select Datasets to Read** dialogue.

## 207.7 NASTRAN .f06 files

*fe-safe* will read analysis results from CSA/NASTRAN v98 and MSC.NASTRAN for Windows v4.5. See Figure 207.6-1 above.

### 207.7.1 Reading stresses

NASTRAN refers to datasets as 'subcases'.

The required file from NASTRAN is the .f06 file (an ASCII file). For importing to *fe-safe* the .f06 file must include the elastic stresses for each data set. NASTRAN does not save the results of the analysis to the .f06 file by default, so this option must be set explicitly (in MSC.NASTRAN this is done in the **File >> Analyse >> Advanced >> NASTRAN Case Control >> Output Requests** section by selecting the Print or Print and PostProcess options).

*fe-safe* will attempt to load stress data only for those subcases that specify stress as an output in the case-control deck of the .f06 file. (Note that since *fe-safe* numbers these data sets sequentially, the number of the data set in *fe-safe* may not be the same as the number of the subcase in the .f06 file.)

Three stress types can be loaded from the .f06 file:

- *Element nodes*: the stresses for each node are read and stored with the element number.
- *Element centroidal*: only the centroidal stresses and strains are read and stored with the element.

- *Both*: both the centroidal and nodal stresses are read and stored with the element number.

#### 207.7.2 Reading temperatures

Temperature datasets are not supported in NASTRAN.

#### 207.7.3 Reading strains

For an elastic-plastic FE analysis, strain datasets can be imported in the same data format as the stress datasets.

By default, *fe-safe* does not read strain datasets. To read strain datasets, select the **Read strains from FE models** option in the **Analysis Options** dialogue.

#### 207.7.4 Supported Element Types

*fe-safe* supports solid, 2D and shell elements - beam elements are ignored by *fe-safe*.

#### 207.7.5 Using shells

Two-layer shells are supported.

#### 207.7.6 Groups

Groups are supported in NASTRAN. They are referred to as SETS in NASTRAN and are extracted from the case control statement in .f06 file.

#### 207.7.7 Writing fatigue results

When a NASTRAN .f06 file is loaded, the output type defaults to CSV (Comma Separated Variable) for export to viewers such as *FEMAP*.

For convenient viewing, the analysis results should normally be exported as log(10) of lives. This option can be set in the **General FE Options** dialogue box.

When the viewer reads the CSV file, the following information regarding the format of the file may be required:

- the first row *does not* contain row titles;
- the first column *does* contain ID's;
- the output type is *ELEMENTAL*.

In Windows NT, to view the results of a fatigue analysis in *FEMAP*, follow the following steps:

- Start *FEMAP*, and load the model file.
- Create a new window for displaying the analysis results, using the **New View** dialogue box by clicking **View >> New....**
- Show the Import Results from dialogue box by clicking **File >> Import >> Analysis Results....**
- In the **Import results from dialogue** box, select **Comma-Separated**.
- Select the .csv results file produced by *fe-safe* and click **open**.
- In the **Read Comma-Separated Table** dialogue box:
  - *unchecked* the **First Row Contains Titles** option;
  - *checked* the **First Column Contains IDs** option;
  - set output type to **Elemental**;
  - select **Create New Set**;

- click **OK**.
- Show the View Select dialogue box by clicking **View >> Select**.
- Set the deformed style to **None - Model Only**, and the contour style to **Contour**.
- Click on **Deformed and Contour Data**.
- In the **Select PostProcessing Data** dialogue box:
  - select the new results set from the **Output Set** drop-down list (the name of the set will be *X*.Table Output, where *X* is a number);
  - select **Table Output Vector Y** from the **Contour** drop-down list, where *Y* is an index number corresponding to the data in the CSV file - **Table Output Vector 1** always corresponds to the data column that contains the *Life* or the *Log of Life*. Other columns may include, for example, *Factor of Strength*, *Failure Rate*, etc.;
  - click **OK**.
- Click **OK** in the **View Select** dialogue box - the contour will plot.

*Tip:* When viewing the fatigue life contours, the areas of the component of most interest are those with the highest probability of failure (i.e. the areas that have the shortest lives). If the contour plot is using FEMAP's default colour palette, the areas of shortest life will be highlighted in blue, which can lead to confusion. Reverse the colours of the contour levels to highlight the areas of shortest life in red by:

- selecting the **View Options** window, by clicking **View >> Options...**;
- select the **Post Processing** option called **Contour/Criteria levels**;
- click **Set Levels...**;
- in the **Contour/Criteria Levels** dialogue box, click on **Reverse**, then **OK**;
- in the **View Options** dialogue box, click **Apply**, then **OK**.

## 207.7.8 Pre-Scanning Support

When pre-scanning is used on a file the **Extract Stresses** option is ignored. The position is selected in the **Select Datasets to Read** dialogue.

## 207.8 ASCII fatigue results files

*fe-safe* fatigue results can be exported to an ASCII file, delimited using tabs, spaces or commas. These options can be configured on the **ASCII FE Model Options dialogue** by selecting **FEA Fatigue >> FEMAP / CSV Interface Options**.



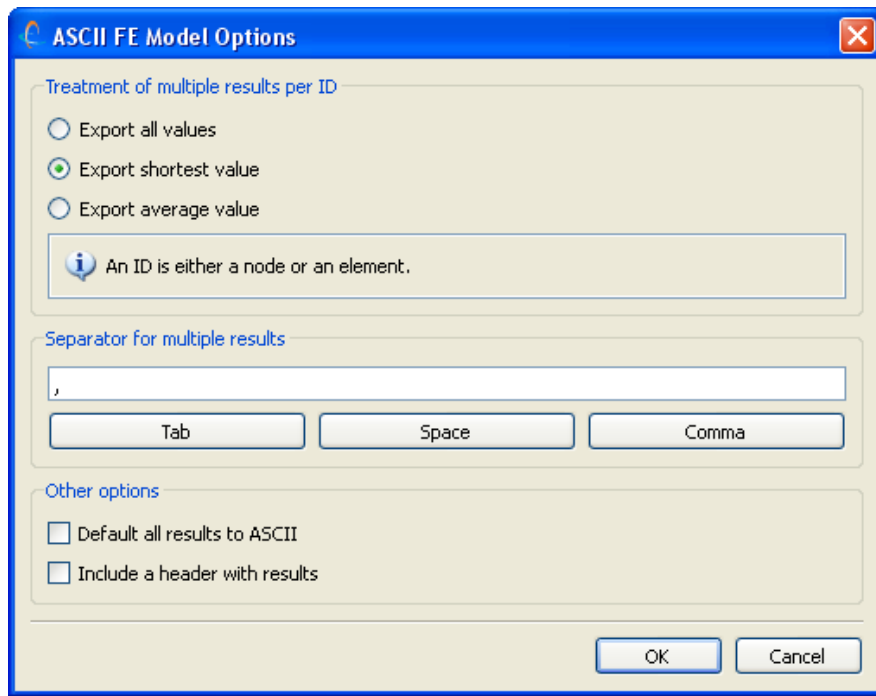


Figure 207.8-1

### 207.8.1 Reading fatigue results

Viewers such as FEMAP can be used to read ASCII data and plot it in conjunction with the original FE model. (Check that the viewer can read the source FE model, since the ASCII fatigue results file contains only the fatigue results.)

The ASCII fatigue results file can also be imported into a spreadsheet application, and the results viewed in a tabular format.

### 207.8.2 Using shells

#### For Abaqus ODB, Abaqus FIL, I-DEAS, NASTRAN and ASCII input models:

Multi-layer fatigue results can be exported. In this case one .csv file will be created for each of the section points in the fatigue results, the name will be based upon the specified output filename and the section number.

For example if the output was `results.csv` and there were fatigue results for sections 1 and 6 the results files would be named `results.-Section1.csv` and `results.-Section6.csv`.

#### For all other model types:

Shells are treated as elements with extra nodes – only the worst-case value for the element or node can be exported.

To instruct *fe-safe* to write fatigue results to an ASCII fatigue results file

To instruct *fe-safe* to use the ASCII fatigue results file format as the default output type for all FE model types, check the **Default all results to ASCII** checkbox in the **ASCII FE Model Options** dialogue.

### 207.8.3 Writing fatigue results

Fatigue results are written to a CSV file if the specified output file has the extension `.csv`.

## 207.84 ASCII fatigue results file format

The format of the ASCII fatigue results file is dependent upon the settings in the **ASCII FE Models Options** dialogue (see Figure 207.8-1 above).

For example, if comma-separation is specified, the results format will be:

```
<El_1>, <node_a>, <var_1>, <var_2>, <var_3>, <var_4>, ...
<El_2>, <node_b>, <var_1>, <var_2>, <var_3>, <var_4>, ...
<El_3>, <node_c>, <var_1>, <var_2>, <var_3>, <var_4>, ...
```

Where each line contains the results for one element/node, and

```
<EL_n>          is an element or node number.
<node_n>        is the node counter, this will only be present if Export All Values is selected.
<var_n>         is the value of a fatigue result variable.
```

On Linux, each line is terminated with a carriage-return character. In WINDOWS, each line is terminated with a carriage-return character plus a line-feed character. (Most FTP utilities will automatically substitute the correct line termination characters when transferring files between platforms.)

The following example shows a CSV file for an analysis of ten elements/nodes, with ten sets of fatigue results written per element:

```
1403,6.5,0.88,92.5,86.1,82.0,65.49,3.81,2.13,1.29,0.83
1477,5.85,0.67,61.07,47.01,40.2,21.3,0.07,0.02,0.01,0
2118,11.617,3.125,100,100,100,100,100,100,100,100
2096,12.284,3.1875,100,100,100,100,100,100,100,100
1846,12.631,3.4375,100,100,100,100,100,100,100,100
1244,13.639,4.1875,100,100,100,100,100,100,100,100
2164,13.813,4.4375,100,100,100,100,100,100,100,100
2440,13.084,3.75,100,100,100,100,100,100,100,100
1274,13.084,3.75,100,100,100,100,100,100,100,100
1325,12.468,3.3125,100,100,100,100,100,100,100,100
```

## 207.85 Saving FEA models as ASCII options

Loaded FEA models can be saved in an ASCII format using the **File** menu option **Save Loaded FE Models** and selecting an extension of .csv or .txt.

The group information is added to the ASCII model if the **Save groups information** check box is checked (default is checked) on the **Export Loaded FE Models to an ASCII file** dialogue.

## 208 Appendix H - Exporting and displaying *fe-safe* results in viewers

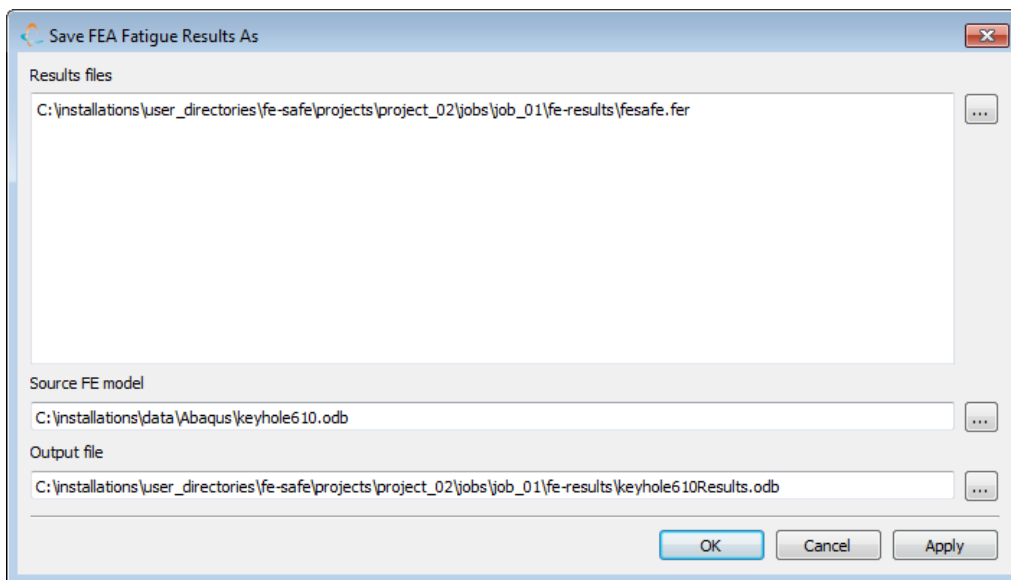
### 208.1 Exporting fatigue results

Fatigue results are exported as contours to be plotted with the original FE model, and may include the following parameters:

- the fatigue life, or  $\log_{10}$ (fatigue life), for fatigue life analyses; or
- the fatigue reliability factor (FRF), when an FRF analysis has been performed;
- the Factor of Strength (FOS), if a design life is specified;
- the Failure Rate or Reliability Rate if a target life is specified - a number of target lives can be specified so there may be more than one sets of results.
- Internally, *fe-safe* uses the .fer file for storing fatigue results. All results are written to the working .fer file (fesafe.fer) before being exported to the final output file.

#### 208.1.1 Using the “Save fatigue Results As...” option

Results stored as .fer files during analysis can later be exported to an alternative output format using the **Save FEA Fatigue Results As...** option.



Specify the .fer file that contains the fatigue life results, and the source FE model.

See Appendix G for the rules that apply when merging fatigue results with the source FE model.

### 208.2 General viewers

Most FE packages include a viewer that will display fatigue results as contours on the original model. Third-party viewers that support one of the export file types described in Appendix G, may also be used.

### 208.3 Patran as a viewer.

If Patran is used as the FE solver and viewer, then the fatigue results should be exported as an ABAQUS FIL (.fil) file. Patran cannot read fatigue lives exported using UVARM variables. However, *fe-safe* can export fatigue lives to the TEMP variable, as  $\log_{10}(\text{life})$  values, by clicking the **Patran As Viewer** button in the **Abaqus FIL Interfacing Options** dialogue box (see section G.2.8).

A further limitation of Patran is that it does not support Averaged at Nodes ABAQUS .fil data.

### 208.4 Cadfix/FAM 4 as a viewer.

This viewer is suitable for plotting fatigue evaluations from Abaqus .fil files.

From the **Abaqus FIL Interface Options** item on the **Options** menu ensure that the output type is a UVARM then save the fatigue results as an Abaqus .fil file.

To import the Abaqus .FIL results into FAM you need to perform the following operations in the viewer:

- Run the **famfromabaqus** utility specifying the default values apart from the first (the FAM file to create, extension *.frm*) and the input *.fil* file.
- Run **FAM**.
- From the **File** menu use the **Open** item to load the FAM file (*.frm* extension) created with famfromabaqus.
- Select the menu item **Generate Connectivity** from the **Mesh** main menu.
- Select the **Results Manager** from the **Results** main menu, A window will be displayed containing a tree structure, expand the tree structure for the last step and there will be a U-VAR item on the tree. Select it and display it. If there are two sub variables in this U-VAR variable then the first is the Life ( or the log of life) and the second is the Strength Factor.

### 208.5 ABAQUS/Viewer

ABAQUS/Viewer can be used to produce a contour plot or a tabular report of fatigue results stored in **ABAQUS Output Database ODB** (.odb) files.

#### 208.5.1 Plotting fatigue results from an ODB file

[See "*Contouring Analysis Results*" in the ABAQUS/Viewer Manual, for more information.]

To plot fatigue results from an ABAQUS ODB file follow these steps in the viewer:

- 1 Open the ODB file (**Main Menu >> File >> Open**).
- 2 Display the step and frame information for the file by opening the Step/Frame dialogue box (**Main Menu >> Result >> Step/Frame**).

The title of an *fe-safe* step has the format:

```
<date> <time> fe-safe <version> <platform>; <description start>
```

e.g.:

```
1-03-04 23:59:59 : fe-safe x.yy[mswin];
#1*test_mcg.amc:1; scale:10;UniE:No--SAE_950C-Manten
```

The name of the frame contains the continuation of the description, e.g.:

```
-Local.dbase--User defined Kt:1;keyhole613.odb;
```

Display the available fatigue output variables for a step by selecting the step and clicking OK.

- 3 Display the available fatigue output variables for a step by opening the Step/Frame dialogue box (**Main Menu >> Result >> Field Output**). The available variables will depend on the analysis that was performed, and may include:

<b>LOGLife-Repeats</b>	log10(fatigue life)
<b>FOS@1e6Repeats</b>	factor of strength

Select the variable to be plotted, and click **OK**.

- 4 Select either the **Contour** or **Symbol** icon to plot the results. (**Symbol** will plot the vector field associated with the critical planes given that the data is at integration points.)

## 208.5.2 Producing a tabular report of fatigue results from an ODB file

[See *"Generating Tabular Data Reports"* in the ABAQUS/Viewer Manual, for more information.]

To produce a tabular report of fatigue results from an ABAQUS ODB file follow these steps in the viewer:

- 1 Open the ODB file (**Main Menu >> File >> Open**).
- 2 Display the **Report Field Output** dialogue box (**Main Menu >> Report >> Field Output**).
- 3 Click the arrow next to the **Position** field and select **Element Nodal** from the list. The list of variable changes to show fatigue output variables. Use the checkboxes to select the variables to be included in the report.
- 4 Display the setup options by clicking the **Setup** tab.
- 5 Specify a filename for the report file in the **Name** field. Select **Append to file** to append to an existing file. If **Append to File** is *not* selected then any existing file contents will be overwritten.
- 6 Set sort options by clicking the arrow next to the **Sort by** field. Specify the sort order by clicking either **Ascending** or **Descending**.
- 7 Configure the remaining options in the dialogue box, then click **Apply** to generate the report.

## 208.6 FEMGV as Viewer

This viewer is suitable for plotting fatigue evaluations from Abaqus .fil files.

From the **Abaqus FIL Interfacing Options** item on the **Interfacing/Results Options** menu ensure that the output type is a **UVARM** then save the fatigue results as an Abaqus .fil file.

To import the Abaqus .fil results into FEMGV you need to perform the following operations in the viewer:

- Run the **postabaqus** utility specifying the name of the .fil file and the name of the resultant file, otherwise accept the default values.
- Run **FEMGV**.
- From the **FEMVIEW** option select the model you imported.
- Select **Results** then **Load Case**. This will list all of the steps in the model, the last step will contain the most recently added fatigue variables, so click on it.

- Select the **Results** then either **Nodal** or **Element** depending on the stresses used. You will be presented with **Uvarm**, select it. If you wish to plot the lives/log lives select **Uvarm1** and if there are FOS values then you can select **Uvarm2**.
- Finally select **Present** then **Contour** then **Levels**. This will display the contour plot of Lives/Log Lives or the FOS depending on which Uvarm was selected.

### 208.7 IDEAS / Master Series as Viewer

To plot UNV results in Ideas first load the .UNV file that contains the source model, then perform the following steps in the viewer:

- Import the results file using the **File** menu item **Import**.
- Move to the post processing of IDEAS.



- (Press the **Results...** button.) Change the **Display Results** to the life variable – this will be the result at index 1000 of the list. Alternatively select a different fe-safe result – these should appear with indices greater than 1000. Then press **Apply/OK**.



- (Press the **Display** button.) Press the right mouse button to select entities (if required) and the middle to draw.



- (Press the **Display Template...** button.) If a vector plot of the critical planes is required select the **Arrow** radio button. Then press **Apply/OK**. Press the **Display** button as before to plot the results.

### 208.8 ANSYS as a Viewer


This viewer is suitable for plotting fatigue evaluations from ANSYS .rst files. To plot the fatigue lives from a .rst file perform the following steps in the viewer:

#### 208.8.1 Using ANSYS Mechanical as a Viewer

##### Contour plots of fatigue results

Run ANSYS. On the **ANSYS Main Menu** select the **General Postproc** item **Results Viewer**. This will prompt for the model to plot. Select the .rst containing the fe-safe results. This will plot the model and pop up the Results Viewer menu.



From the left hand drop down box select **Element Solution >> Stress >> X-Component of Stress**. This will select the first contour created by fe-safe – usually the Log10(Lives) contour. Then press the **Plot Results** toolbar item  to plot the contour.

To plot the second and third fe-safe contours select the Y and Z components of Stress. The fatigue analysis .log will indicate which contour type is related to which fatigue contour.

### Vector plots of fatigue results

Follow the instructions in the previous section to load the model, select the dataset containing the critical planes. Exit the **Results Viewer**. Type the following commands into the ANSYS command line (alternatively save them as a macro, say critical.mac):

```
ETABLE, CritX, S, X
ETABLE, CritY, S, Y
ETABLE, CritZ, S, Z
PLVECT, CritX, CritY, CritZ, Crit, VECT, ELEM, ON, 0
```

Since the ETABLE command constructs results averaged over the element the **ANSYS .RST Interface Options** dialogue box provides the option “For critical plane export use worst on element”, which is selected by default.

## 208.9 Using ANSYS Workbench as a Viewer

### 208.9.1 When an ANSYS Workbench Project file is available

If the user has used ANSYS Workbench to create the FEA results file (\*.rst) that was used in fe-safe to calculate fatigue life, then the Analysis System may be duplicated, and the duplicate system used to post-process fe-safe results. The benefit of this method is that the Report options already used in Workbench for the Stress Analysis can be reused for the fatigue results.

- In your Results directory make a new sub-directory for your ANSYS results files\*.
  - \* NOTE: ANSYS Workbench will copy all of the files in the folder of the results selected back to its own directory structure, which could result in large and unnecessary data if the Ansys Result is accessed from the fe-safe Results Directory
- Copy your fe-safe results log file (\*.log) to the subdirectory.
- Change the names of the copied results file (\*.rst) to “file.rst” and change the name of the log file (\*.log) to “solve.out”.
- In ANSYS Workbench, open the Workbench Project File (\*.wbpj) corresponding to the FEA Results used for FEA Fatigue in fe-safe. The block(s) will appear in ANSYS Workbench.
- Right-click on one block and select **Duplicate**. The duplicate system will appear in the project schematic area. Rename it.
- Right-click on the “Solution” branch of the duplicate system and select **Edit**. The Mechanical Window will appear.

- In the Mechanical Window right-click on the “Solution” branch of the duplicate system and select **Clean**. Select **OK**.
- Select **Tools > Read Results File**. Select the fe-safe Results File (\*.rst)
- The “**Solution Information**” branch (under “Solution”) can be selected to view the fe-safe results log file (solve.out).
- The X component of stress SX is the field containing Life or Log-Life contours, and Sy, Sz, etc for other selected Contour Exports (see Appendix G section 207.5.8 for more details). Select Sx to view fatigue life results.
- **Note:** A red lightning bolt will appear in your duplicate system. Although certain files are missing, the results will be available in Model Viewer.

## 208.10 FEMAP as a Viewer

### 208.10.1 Plotting scalar fields

In Windows NT, to view the results of a fatigue analysis in FEMAP, follow the following steps:

- Start FEMAP, and load the model file.
- Create a new window for displaying the analysis results, using the **New View** dialogue box by clicking **View >> New...**
- Show the **Import Results from** dialogue box by clicking **File >> Import >> Analysis Results...**
- In the **Import results from** dialogue box, select **Comma-Separated**.
- Select the .csv results file produced by *fe-safe* and click **open**.
- In the **Read Comma-Separated Table** dialogue box:
  - *unchecked* the **First Row contains Titles** option;
  - *checked* the **First Column Contains IDs** option;
  - set output type to **Elemental**;
  - select **Create New Set**;
  - click **OK**.
- Show the **View Select** dialogue box by clicking **View >> Select**.
- Set the deformed style to **None - Model Only**, and the contour style to **Contour**.
- Click on **Deformed and Contour Data**.
- In the **Select PostProcessing Data** dialogue box:
  - Select the new results set from the **Output Set** drop-down list (the name of the set will be **X.Table Output**, where **X** is a number);
  - Select **Table Output Vector Y** from the **Contour** drop-down list, where **Y** is an index number corresponding to the data in the CSV file - **Table Output Vector 1** always corresponds to the data column that contains the *Life* or the *Log of Life*. Other columns may include, for example, *Factor of Strength*, *Failure Rate*, etc.;
  - Click **OK**.
- Click **OK** in the **View Select** dialogue box - the contour will plot.



*Tip:* When viewing the fatigue life contours, the areas of the component of most interest are those with the highest probability of failure (i.e. the areas that have the shortest lives). If the contour plot is using FEMAP's default colour palette, the areas of shortest life will be highlighted in blue, which can lead to confusion. Reverse the colours of the contour levels to highlight the areas of shortest life in red by:

- Selecting the **View Options** window, by clicking **View >> Options...**;
- Select the **Post Processing** option called **Contour/Criteria levels**;
- Click **Set Levels...**;
- In the **Contour / Criteria Levels** dialogue box, click on **Reverse**, then **OK**;
- In the **View Options** dialogue box, click **Apply**, then **OK**.

#### 208.10.2 Plotting vector fields

- Show the **View Select** dialogue box by clicking **View >> Select**.
- Set the contour style to **Vector**.
- Click on **Deformed and Contour Data**.
- Click on **Contour Vectors...**
- Under **Vector Type** select either the **2D** or **3D components** radio button.
- From the drop down list select **Table Output Vector 2**, etc. for the components of the vector.

#### 208.11 MetaPost as a Viewer

- Select **File>Open File...** Select the file containing model geometry to load. Select **OK**.
- The mesh will appear. Note the file appears in the **Filename** field of the **Geometry** tab on the **Read Results** dialogue
- Use the left mouse button to toggle to the **Results** tab.
- Browse to select the FEA results file in the **Filename** tab (if not automatically populated). Depending on the FEA results file format the geometry and results may be in the same file.
- Note the **scalar** tab contains contour variables. Use the drop-down boxes to select the fatigue life variable corresponding to the results file format (see Appendix G for more details)
  - For example: Select **Stresses>Normal-X** (in the case of \*.op2 files)
- Select the **Read** button on the **Read Results** dialogue.
- Use the left mouse button to toggle to the **States** tab next to **Read Results**.
- Select the solution state to contour and then select the **Fringe** button in the **Basic Buttons** window
- Use the right mouse button on the Fringe button to set **Fringe Colors** or **Fringe Range**



## 209 Appendix I – Recommended procedures for fatigue testing of materials

Materials tests to determine the local strain fatigue parameters are defined in ASTM Standard E606-92 (Re-approved 1998) 'Standard Practice for Strain-Controlled Fatigue Testing' (Ref. 1). This document references other ASTM Standards including those for verification and classification of extensometers (E83), and test methods for constant amplitude fatigue testing (E466, E467).

### 209.1 Standard equations for strain-based fatigue relationships

The following parameters are defined

$E$	the elastic modulus (Young's Modulus)
$K'$	the strain hardening coefficient
$n'$	the strain hardening exponent
$b$	the fatigue strength exponent (Basquin's exponent)
$\sigma'_f$	the fatigue strength coefficient
$c$	the fatigue ductility exponent (the Coffin-Manson exponent)
$\varepsilon'_f$	the fatigue ductility coefficient

For many metallic materials the cyclic stress strain curve can be approximated by the equation

$$\varepsilon = \frac{\sigma}{E} + \left( \frac{\sigma}{K'} \right)^{\frac{1}{n'}} \quad (1.1)$$

and the hysteresis loop stress-strain curve can be approximated by the equation

$$\Delta\varepsilon = \frac{\Delta\sigma}{E} + 2 \left( \frac{\Delta\sigma}{2K'} \right)^{\frac{1}{n'}} \quad (1.2)$$

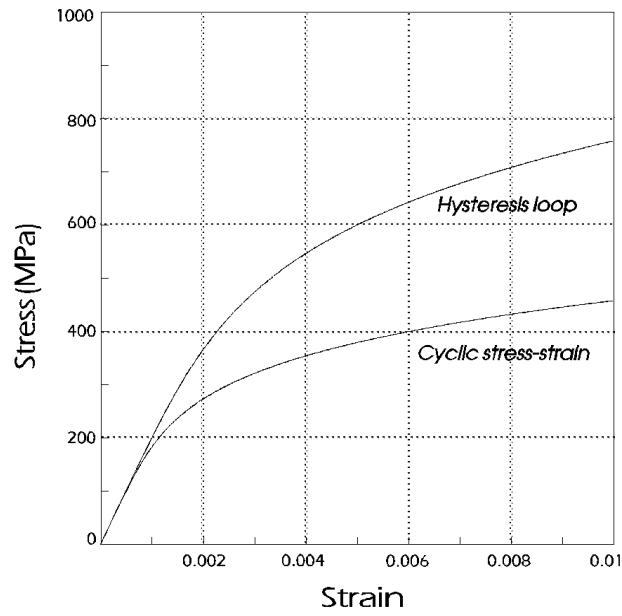


Figure 1 Stress-strain relationships

The relationship between true local stress amplitude and endurance is

$$\frac{\Delta\sigma}{2} = \sigma'_f (2N_f)^b \quad (1.3)$$

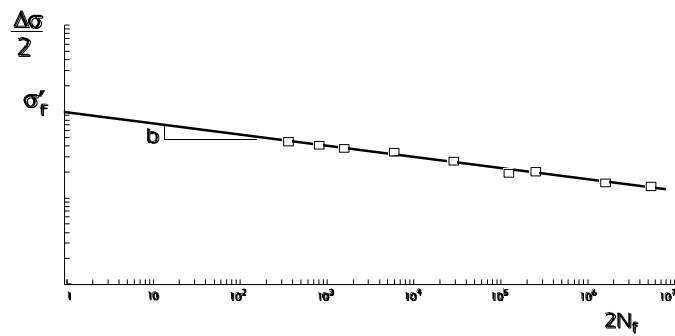


Figure 2 Stress-life relationship

The relationship between true local strain amplitude and endurance is

$$\frac{\Delta\varepsilon}{2} = \frac{\sigma'_f}{E} (2N_f)^b + \varepsilon'_f (2N_f)^c \quad (1.4)$$

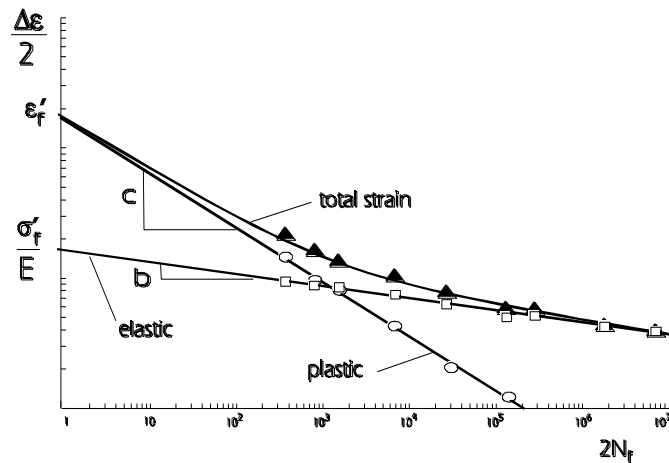


Figure 3 Strain-life relationship

## 209.2 Main features of ASTM E606-92, with comments

In the following section, Dassault Systemes comments and recommendations are shown underlined.

### 209.2.1 Test specimen

For bar materials, two specimen designs are provided (see *Reference 1, Figure 1, "Recommended Low-Cycle Fatigue Specimens"*), where specimen (a) has a uniform gauge length, and specimen (b) has an hourglass test section.

An axial clip gauge (extensometer) can be used with specimen (a) to measure axial strains. For specimen (b) diametral strains must be measured and converted into axial strains. It is recommended that specimen (a) be used whenever possible, because axial strains can be measured directly.

The minimum diameter of the gauge length is 6.35 mm, giving a cross-sectional area of approximately 31.67 mm<sup>2</sup>. For cast materials, we would recommend a 10mm diameter, otherwise the fatigue lives can be unduly influenced by a specific flaw or defect.

Test specimens should be fine machined, using a very small depth of cut to avoid introducing residual stresses. Longitudinal polishing should be used to finish the specimen. We would recommend final finishing with polishing tape. We recommend a system which rotates the specimen slowly while polishing the specimen longitudinally with polishing tape. The tape is held on a spool, and fed from one spool to another and back again repeatedly during the polishing, until the tape is worn. Using a new tape for each specimen and wearing it flat for each specimen also ensures a consistent finish. Finished specimens must be protected against physical and environmental damage.

### 209.2.2 Test set-up

The test machine should meet the requirements of ASTM Practices E 4 and E 467. The machine should be one in which specific measures have been taken to minimize backlash in the loading train. (See Section 6.1 in E606).

Testing machine controls should allow constant amplitude cycling between defined strain limits. If material behaviour permits (for example, aging effects are not significant), control stability should be such that the strain maximum and minimum limits are repeatable over the test duration to within 1 % of the range between maximum and minimum control limits. (See Section 6.2 in E606).

Bending strains in the test specimen should be as low as possible. E606 Section 6.3 recommends that bending strains should be less than 5% of the minimum axial strain range to be used in the series of tests. This should be checked using a strain-gauged specimen using the methods defined in E606.

Several designs for the specimen ends and the end fittings are given in E606.

Extensometers should qualify as Class B-2 or better in accordance with Practice E 83 and should be calibrated before and after each test (See E606 Section 6.4). (A schematic diagram showing the measurement of longitudinal displacement for a specimen with a uniform-gauge test section is shown in *Figure 5(a)* of Reference 1).

A load transducer should be placed in series with the test specimen to measure the magnitude and direction of the axial load transmitted through the specimen. Load transducer capacity should be selected to adequately cover the range of loads to be measured in the test. (See E606 Section 6.5).

At least 10 specimens should be tested successfully - i.e. after specimens which fail outside the gauge length, or from the extensometer knife edges, have been eliminated. We would recommend that test lives should cover a range from 100 to almost  $10^7$  cycles.

E606 specifies that all specimens should be tested under strain control. The only exceptions are the very high cycle specimens with lives between  $10^6$  and  $10^7$  cycles. For these specimens, testing may be started under strain control, then switched to load control once the stress-strain response has stabilised.

The test waveform should be as close to a triangular waveform as possible, to avoid stress relaxation at the loop tips. The same load direction should be used for the first load application to each specimen. Cycle frequency can be as high as practical without raising the temperature of the specimen. A 2° C limit on temperature rise is recommended. Test temperature should be recorded. Ambient temperature should not vary by more than 2° C for the test program.

### 209.2.3 Test recording

When a smooth test specimen is cycled between fixed strain limits, the stress response may show that the material is softening, producing lower stresses for each strain application. After a number of cycles the cyclic stress-strain curve stabilises, producing a stable hysteresis loop. (*Figure 4*)

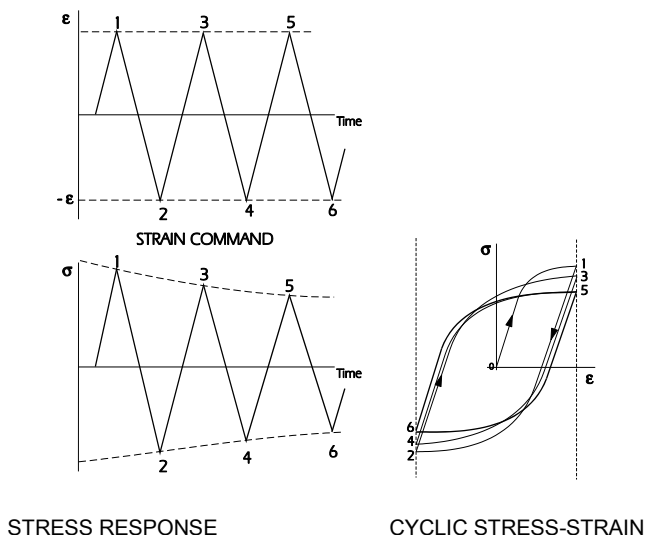


Figure 4 Cyclic softening

For other materials the stress response may show that the material is hardening.

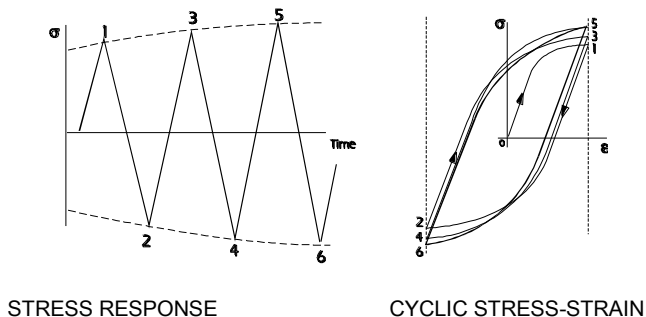


Figure 5 Cyclic hardening

In order to monitor any cyclic hardening or softening, stress-strain hysteresis loops should be digitised and recorded as pairs of stress-strain data points. We would recommend at least 512 points to define the hysteresis loop. If continuous recording of loops is not possible, then complete loops should be recorded on a pseudo-log increment scale, for example loops 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000 etc, throughout the test.

When calculating the true stress for the very low cycle specimens, the cross-sectional area of the specimen under load must be used. The procedure is as follows.

The diametral strain can be calculated from the axial strain, using both the elastic and plastic values of Poisson's ratio.

From an initial estimate of the stress based on the full cross-sectional area, the elastic component of strain  $\epsilon_e$  is determined

$$\epsilon_e = \frac{\sigma}{E}$$

The plastic component of strain is  $\varepsilon_p = \varepsilon - \varepsilon_e$

The diametral strain is  $-(\nu_e \varepsilon_e + \nu_p \varepsilon_p)$

The diametral strain is used to calculate a new cross-sectional area, which allows new estimates of the axial stress and hence  $\varepsilon_e$  and  $\varepsilon_p$ . A new estimate of the diametral strain can then be calculated, and hence a new estimate of axial stress. This process is repeated until a convergent value of axial stress is obtained.

This procedure can be automated using a computer program or spread sheet.

It is conventional to use  $\varepsilon_p = 0.5$

The strain command and the stress response should be monitored continuously. A consistent criterion must be adopted for defining the fatigue life. E606 suggests several options, including complete separation of the specimen, change of the unloading modulus, measured crack length, and a 50% reduction in load for a given applied strain.

Our experience is that when cycling under strain control, complete separation of the specimen is not a reliable criteria. This is because the reduction in load to maintain constant strain amplitude can result in very low crack propagation rates, and these conditions do not represent typical service loading.

For the materials data presented in Ref. 2, a wide variety of fatigue life criteria are used, including 0.1mm, 0.5mm and 1.0mm crack lengths, 10% and 50% stress reductions, and complete separation. We would recommend a stress reduction criterion, using 10% or 50% stress reduction. It is important that a consistent basis is used, and that the fatigue life criterion is documented.

### 209.3 Data analysis

This analysis procedure has been used by Dassault Systemes UK Ltd and is consistent with ASTM E606.

#### 209.3.1 Cyclic stress-strain relationship

The first strain excursion of a cyclically stabilised material follows the cyclic stress strain curve, and subsequent strain excursions follow the hysteresis stress-strain curve.

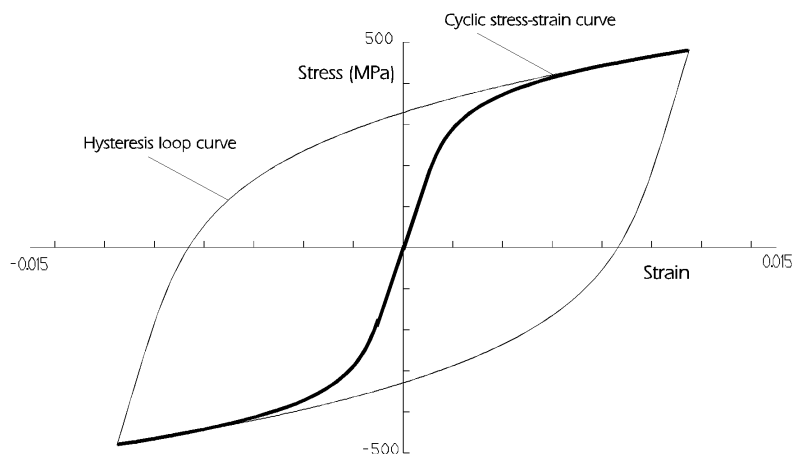




Figure 6 Cyclic and hysteresis loop curves

Other strain amplitudes will produce stable hysteresis loops of different sizes, but all the loops will have their tips on the cyclic stress strain curve.

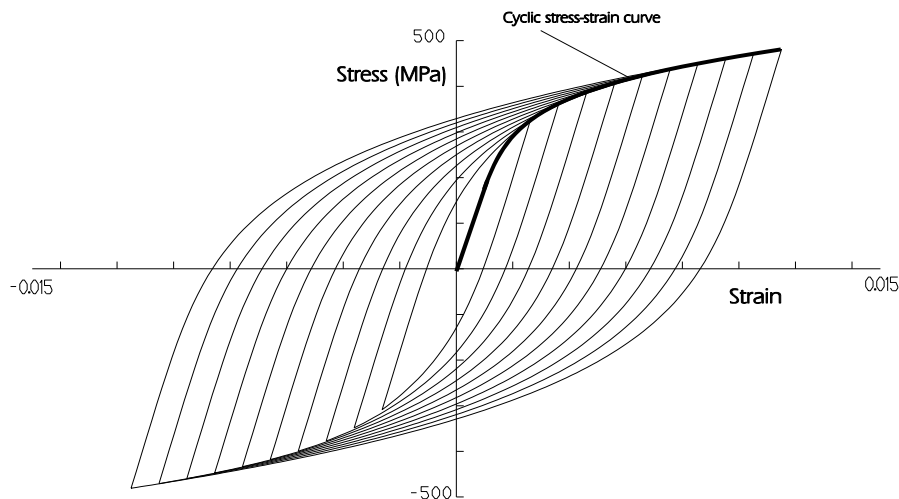


Figure 7 Cyclic stress-strain curve through the hysteresis loop tips

The stable stress-strain hysteresis loops from the specimens are plotted. A curve drawn through the loop tips defines the cyclic stress-strain curve.

The equation for the cyclic stress-strain curve is

$$\varepsilon = \frac{\sigma}{E} + \left(\frac{\sigma}{K'}\right)^{\frac{1}{n'}}$$

Taking the plastic component of strain,

$$\varepsilon_p = \left(\frac{\sigma}{K'}\right)^{\frac{1}{n'}}$$

Re-arranging this equation gives

$$\sigma = K'(\varepsilon_p)^{n'}$$

Taking logs of both sides of this equation

$$\log_{10} \sigma = \log_{10} K' + n' \log_{10} (\varepsilon_p)$$

This is the equation for a straight line on log-log axes.

The data points are plotted on axes of  $\log_{10}(\sigma)$  vs  $\log_{10}(\epsilon_p)$ . (Figure 8)

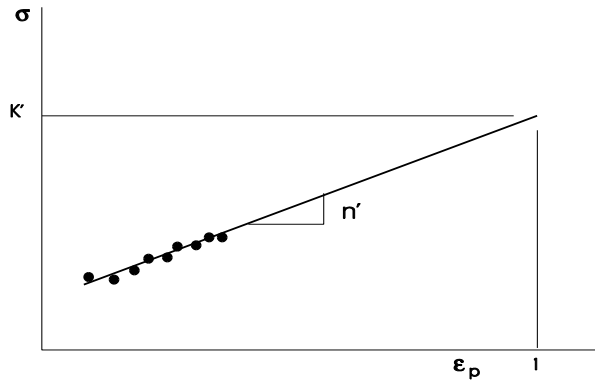


Figure 8 Calculation of  $N'$  and  $K'$

A least-squares regression analysis may be used to calculate the best-fit straight line through the data. The slope gives the value of  $n'$ .

For an extrapolated strain of  $\epsilon_p = 1$ , in the equation

$$\log_{10} \sigma = \log_{10} K' + n' \log_{10} (\epsilon_p)$$

if  $\epsilon_p = 1$ , then  $\log_{10}(\epsilon_p) = 0$  and so  $\sigma = K'$

The strain-life curve

The strain-life equation may be separated into the elastic and plastic components, each of which is a straight line on log-log axes.

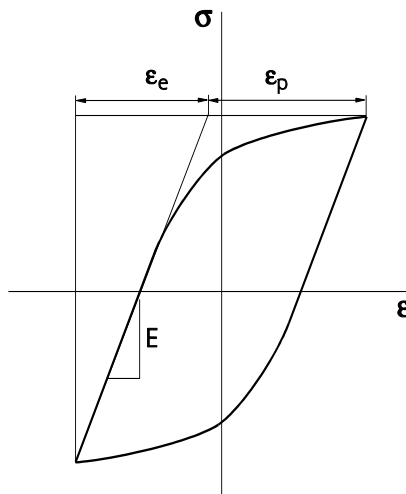


Figure 9 The cyclic stress-strain hysteresis loop as the sum of elastic and plastic strains

$$\frac{\Delta\epsilon_e}{2} = \frac{\sigma'_f}{E} (2N_f)^b$$

$$\frac{\Delta\epsilon_p}{2} = \epsilon'_f (2N_f)^c$$

The elastic and plastic components of strain can be determined for each strain amplitude (see *Figure 9*) using the equation for the hysteresis stress-strain curve (1.2). Regression analysis is used to give a best-fit straight line for the elastic and plastic curves, and provide the four parameters (*Figures 10 and 11*). The two curves are added to give the strain-life curve, as shown in *Figure 12*.

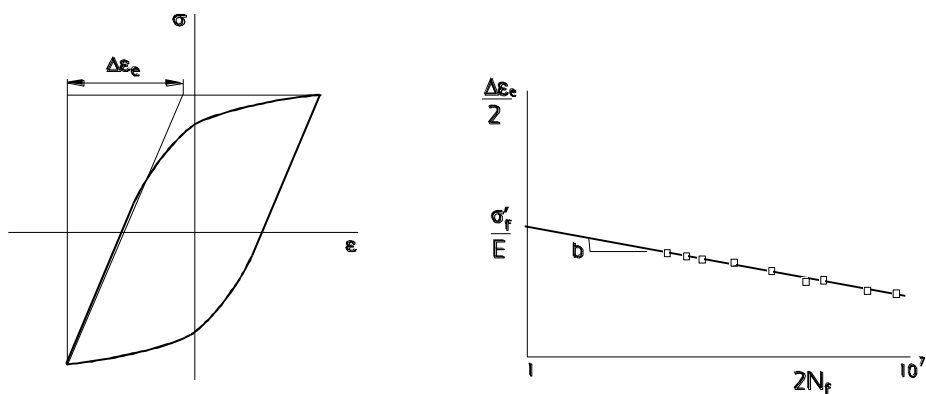


Figure 10 Relationship between elastic strain and endurance

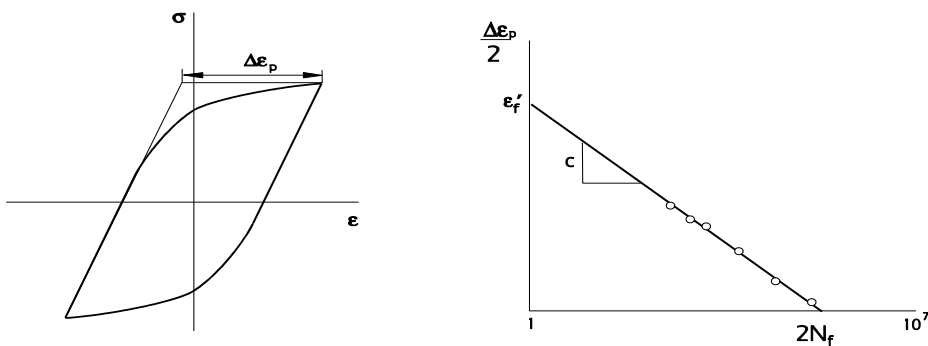


Figure 11 Relationship between plastic strain and endurance

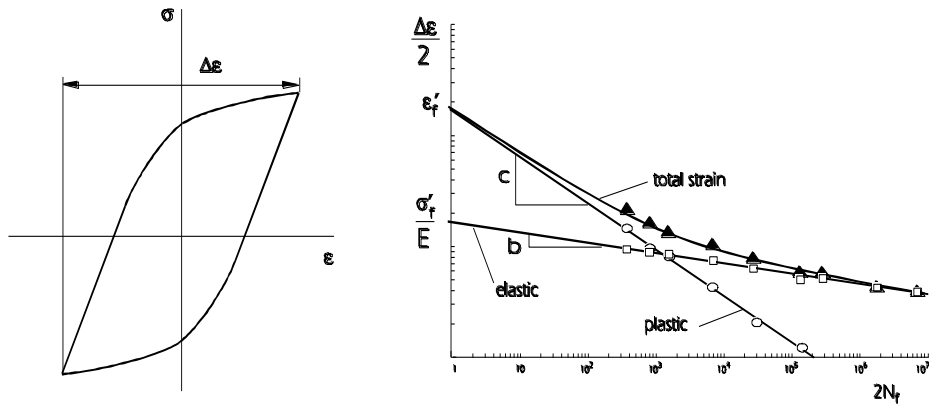


Figure 12 Example of test data to determine the strain-life curve.

For materials which produce symmetrical hysteresis loops but which are never really stable, it is usual to calculate  $n'$  and  $K'$  from hysteresis loops measured at the half-life point of each specimen.

Note: Theoretical stress-life curves take into account change in area, i.e.:

$$\Delta\sigma = \frac{P}{A/A_0}$$

However, S-N curves from measured laboratory data are usually  $\{\Delta\sigma/2 \text{ vs. } N_f\}$  or  $\{\Delta\sigma \text{ vs. } N_f\}$ , but do not take into account change of area, i.e.  $\Delta\sigma = P/A_0$ .

Measured laboratory data of this type (that does not take into account the change of area) will usually hold good down to an endurance of around 1E4 cycles. Below these lives some correction should be made to the fatigue strength coefficient,  $s_f'$ , derived from the SN curve, and hence the fatigue strength exponent,  $b$ . In the absence of measured data, this will need to be estimated, possibly using elongation data.

An initial estimate of the elongation could be made from the  $P/A_0$  stress by using Poisson's ratio to get the new area,  $A$ . Then find the new stress, make a new estimate of the elongation, and go round the loop until a stable solution is found.

## 209.4 References

ASTM Standard E606-92 (Re-approved 1998)

*'Standard Practice for Strain-Controlled Fatigue Testing'*

American Society for Testing and Materials

Chr Boller and T Seeger

*Materials Data for Cyclic Loading*

Materials Science, Monographs, 42A

Elsevier Science Publishers BV, 1987 (ISBN 0-444-42875-5)