# CZECH TECHNICAL UNIVERSITY IN PRAGUE

## Faculty of Mechanical Engineering, Dept. of Mechanics, Biomechanics and Mechatronics


Report title:

# Brief manual of scripts for thermo-mechanical fatigue predictions

Software ID:

**12105 S01 2015.12**


Grant:

**TE 01020020**



Leader:

**Doc. Ing. Miroslav Španiel, CSc.**

Head of Dept.:

**Prof. Ing. Milan Růžička, CSc.**

Authors:

**Doc. Ing. Miroslav Španiel,CSc., Ing. Michal Bartošák**


Prague, December 2015


| | | | |
|---|---|---|---|
| Adress: | **Technická 4** | tel.: **2 2435 2509** | IČO: **68407700** |
| | **166 07 Praha 6** | fax: **2 3332 2482** | DIČ: **CZ 68407700** |

**CTU in Prague, Faculty of Mechanical Engineering, Dept. of Mechanics, Biomechanics and Mechatronics**

Technická 4, 166 07 Praha 6

---

## Anotační list

| | |
|---|---|
| *Název zprávy:* | **Stručný návod ke skriptům pro hodnocení nízkocyklové teplotně mechanické únavy.** |
| *Title:* | **Brief manual of scripts for thermo-mechanical fatigue predictions** |
| *Software ID:* | **12105 S01 2015.12** |
| *Grant:* | **TE 01020020** |
| *Leader:* | Doc. Ing. Miroslav Španiel, CSc. |
| *Authors:* | Doc. Ing. Miroslav Španiel, CSc. ..... |
| | Ing. Michal Bartošák ..... |
| *Head of dept:* | Prof. Ing. Milan Růžička, CSc. ..... |
| *Bibliography:* | 54 pages, 57 figures, 14 tables, 6 appendices |
| *Summary:* | The report describes script for calibration of Chaboche plasticity, script for evaluation of damage by LCTMF and creep within FEM program Abaqus. |
| *Anotace:* | Zpráva popisuje skript pro kalibraci Chabocheovy plasticity, skript pro vyhodnocení počkození od NCTMÚ a creepu v rámci MKP programu Abaqus. |

# Contents

# Chapter 1

# Elastic–plastic material model

For numerical description, elastoplastic model has been chosen. As plastic zones are expected in large areas, other way elastic model with appropriate notch formula can be used. Viscoplastic FEA is too much time consuming. Only local viscoplastic approximations will be made in future.

## 1.1 Temperature dependent Chaboche kinematic hardening

In FEM program Abaqus Chaboche model is implemented as the *nonlinear isotropic/kinematic hardening model with multiple backstresses* included under models of plasticity for metals subjected to cyclic loading. It is built on the basis of *Mises yield surface*

$$F = f\left(\boldsymbol{\sigma} - \boldsymbol{\alpha}\right) - \sigma_Y = 0\,, \tag{1.1}$$

where,

$$f\left(\boldsymbol{\sigma} - \boldsymbol{\alpha}\right) = \sqrt{\frac{3}{2}\left(\boldsymbol{\sigma}^{\mathrm{dev}} - \boldsymbol{\alpha}^{\mathrm{dev}}\right) : \left(\boldsymbol{\sigma}^{\mathrm{dev}} - \boldsymbol{\alpha}^{\mathrm{dev}}\right)}\,, \tag{1.2}$$

$\boldsymbol{\sigma}$ is stress tensor with deviatoric part

$$\boldsymbol{\sigma}^{\mathrm{dev}} = \boldsymbol{\sigma} - \frac{1}{3}\mathrm{trace}\left(\boldsymbol{\sigma}\right)\,\mathbf{I}\,,$$

$\boldsymbol{\alpha}$ is backstress tensor with deviatoric part

$$\boldsymbol{\alpha}^{\mathrm{dev}} = \boldsymbol{\alpha} - \frac{1}{3}\mathrm{trace}\left(\boldsymbol{\alpha}\right)\,\mathbf{I}\,;$$

*associative flow rule*

$$\dot{\boldsymbol{\varepsilon}}^{pl} = \dot{\bar{\varepsilon}}^{pl}\frac{\partial F}{\partial \boldsymbol{\sigma}}\,, \tag{1.3}$$

where equivalent plastic strain rate depends on plastic strain tensor rate $\dot{\boldsymbol{\varepsilon}}^{pl}$, as

$$\dot{\bar{\varepsilon}}^{pl} = \sqrt{\frac{2}{3}\dot{\boldsymbol{\varepsilon}}^{pl} : \dot{\boldsymbol{\varepsilon}}^{pl}}\,;$$

and *nonlinear isotropic/kinematic hardening model* to determine backstress history

$$\boldsymbol{\alpha} = \sum_{k=1}^{N} \boldsymbol{\alpha}_k \,, \tag{1.4}$$

where

$$\dot{\boldsymbol{\alpha}}_k = C_k \frac{1}{\sigma_Y} (\boldsymbol{\sigma} - \boldsymbol{\alpha}) \, \dot{\bar{\varepsilon}}^{pl} - \gamma_k \boldsymbol{\alpha}_k \dot{\bar{\varepsilon}}^{pl} \,. \tag{1.5}$$

Cyclically stable *cyclic stress-strain curves* were obtained experimentally at high strain rates (to exclude viscous effect) for a few distinct temperatures in condition of uniaxial strain-controlled loading with given strain amplitude $\varepsilon_a$ ,kept constant during the test up to fracture, and mean value $\varepsilon_m = 0$. These curves represent the dependence of equivalent strain amplitude $\varepsilon_a$, resp. equivalent plastic strain amplitude $\varepsilon_{a\,pl}$ on stress amplitude $\sigma_a$ of stable cycle in conditions of strain-controlled loading Cyclic stress-strain curves, or hysteresis loops , are needed for elastoplasticity calibration.

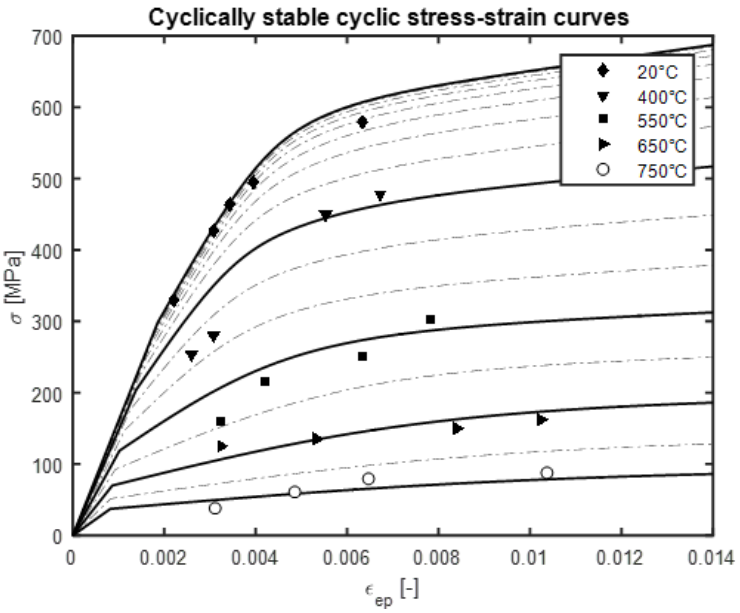


Figure 1.1: Cyclic stress- strain curves

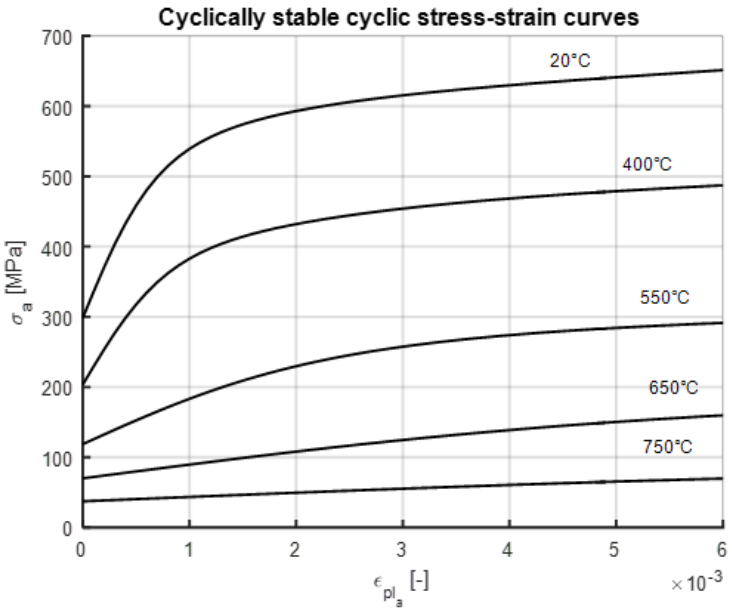


Figure 1.2: Cyclic stress- strain curves, plastic part only

Temperature dependent Chaboche non-linear kinematic hardening has been chosen to describe plastic response of the material. Stress ampllitude of stabilized cycle $\sigma_a$ in conditions of uniaxial isothermal strain-controlled loading in Chaboche model can be expressed, using infinite number of cycles while integrating equations 1.1–1.5, as

$$\sigma_a = \sigma_Y + \sum_{k=1}^{N} \begin{cases} \frac{C_k}{\gamma_k} \tanh\left(\gamma_k . \varepsilon_{a\,pl}\right) & \text{for} \quad \gamma_k > 0 \\ C_k \varepsilon_{a\,pl} & \text{for} \quad \gamma_k = 0 \end{cases} . \tag{1.6}$$

Parameters $C_k$ and $\gamma_k$ at given temperature describe the cyclically stable stress-strain curve (see Fig 1.2). For instance, model with three backstresses ($\gamma_3$ is assumed equal 0) presented below, coluld be used to calibrate parameters $C_k\left(T_i\right)$, $\gamma_k\left(T_i\right)$, where $T_i \in \{20, 400, 550, 650, 750\}$,

$$\sigma = \sigma_y + \frac{C_1\left(T_i\right)}{\gamma_1\left(T_i\right)} \tanh(\gamma_1\left(T_i\right) \epsilon_{a\,pl}) + \frac{C_2\left(T_i\right)}{\gamma_2\left(T_i\right)} \tanh(\gamma_2\left(T_i\right) \epsilon_{a\,pl}) + C_3\left(T_i\right) \epsilon_{a\,pl} \tag{1.7}$$

temperature by temperature. This way of calibration is insufficient, because temperature dependence of parameters $C_k\left(T_i\right)$, $\gamma_k\left(T_i\right)$ is oscilating and cannot be used to describe loading with variable temperature. Rather than on single cyclic stress-strain curves at single temperatures, model parameters have been calibrated to follow monotonic downward trend, dependent on temperature. After some experimental research, appropriate form of temperature dependence of Chaboche parameters was mathematically described as Boltzmann function

$$C_k, resp.\gamma_k = \frac{a_1 - a_2}{1 + exp\frac{T - a_3}{a_4}} + a_2 \tag{1.8}$$

where $a_1, a_2, a_3$, and $a_4$ are constants, $T$ is temperature [°C]. Script based on Python was programmed to perform such a calibrations. It is described in

chapter 3 on page 22. Example calibration is presented in figures 1.3–1.5, and in table 1.1.

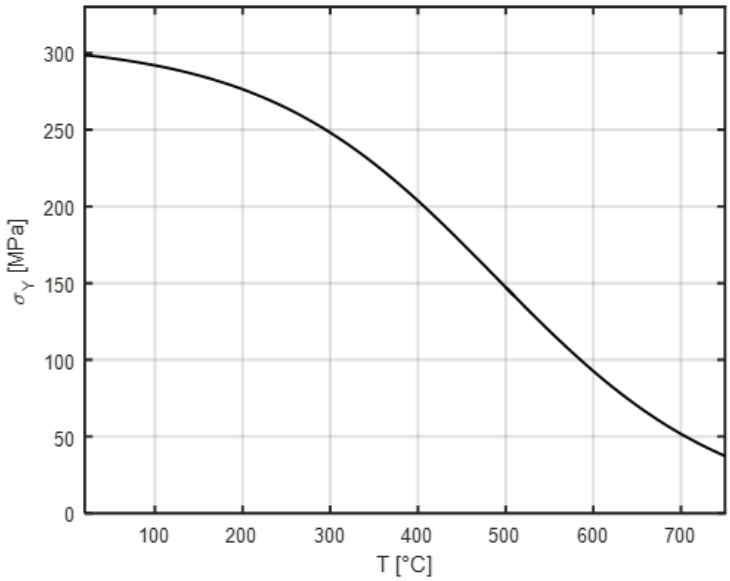


Figure 1.3: Yield stress

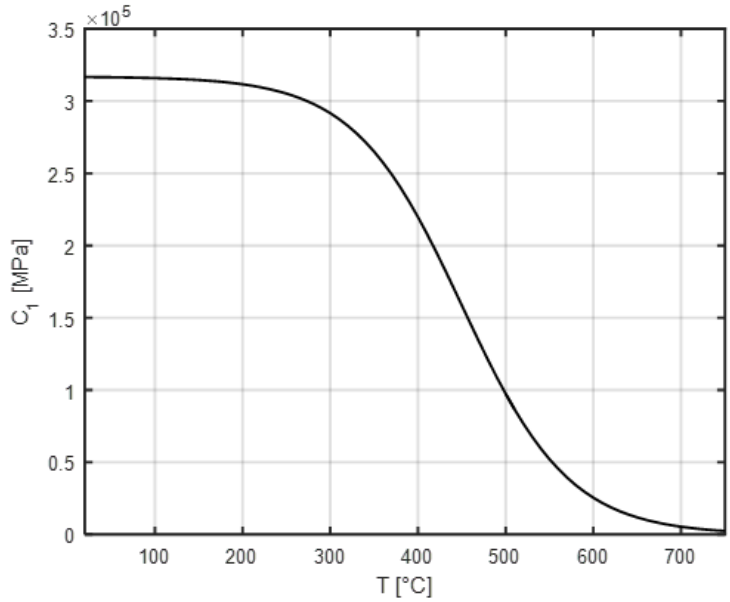


Figure 1.4: Temperature dependent backstress parameter $C_1$

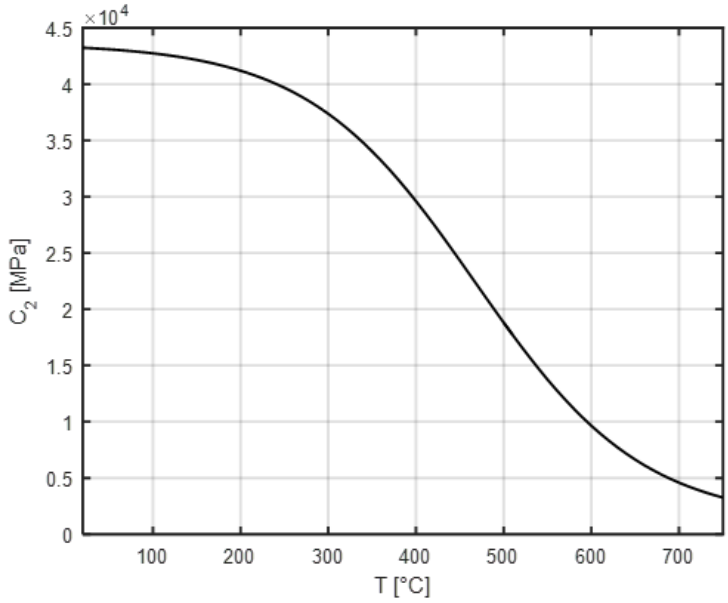


Figure 1.5: Temperature dependent backstress parameter $C_2$

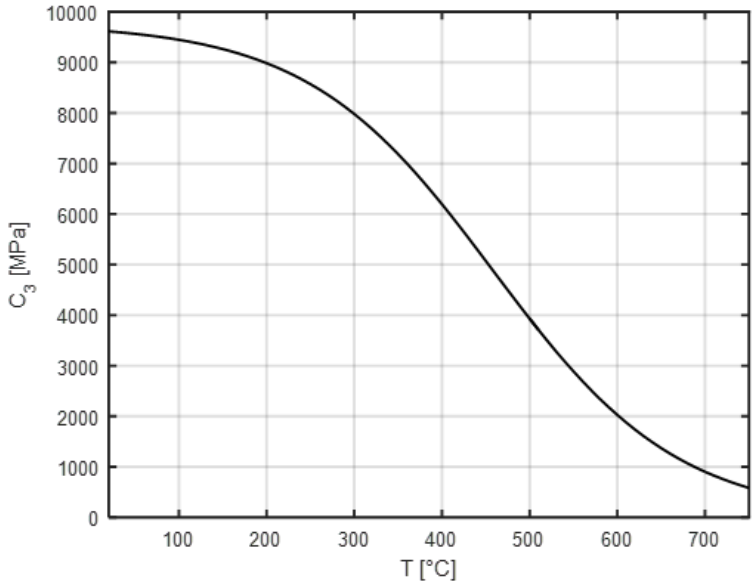


Figure 1.6: Temperature dependent backstress parameter $C_3$

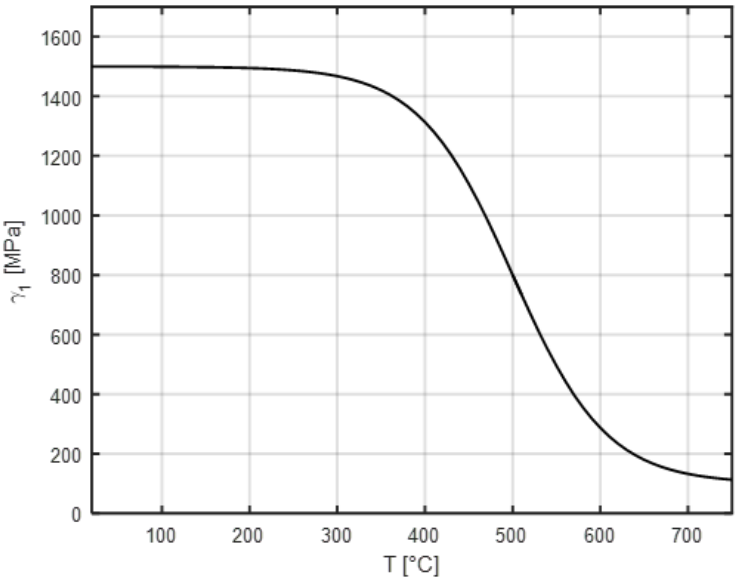


Figure 1.7: Temperature dependent backstress parameter $\gamma_1$

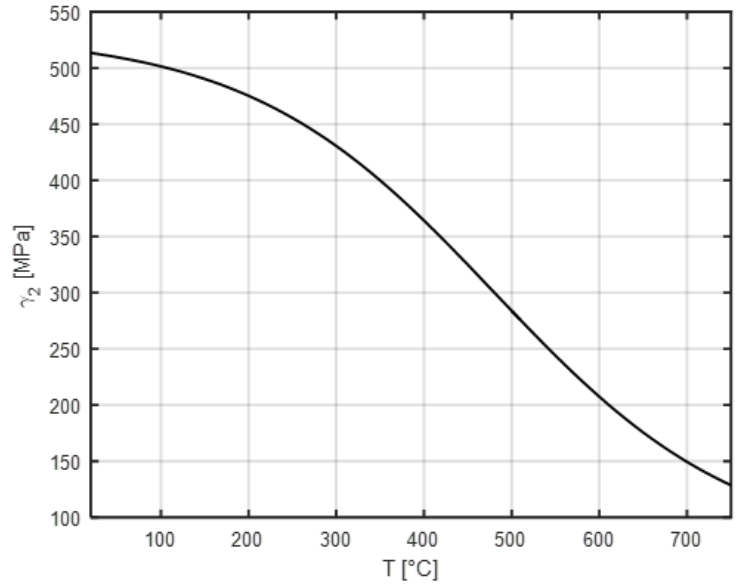


Figure 1.8: Temperature dependent backstress parameter $\gamma_2$

Table 1.1: Boltzmann function constants

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ |
|---|---|---|---|---|
| E | 160999,5 | 72,13443 | 651,079 | 105,5435 |
| $\sigma_y$ | 307,2062 | 3,65E-07 | 489,1807 | 131,9151 |
| $C_1$ | 316971,3 | 7,145086 | 450 | 61,41123 |
| $\gamma_1$ | 1500 | 100 | 500 | 53,40917 |
| $C_2$ | 43632,34 | 1173,358 | 467,4107 | 95,44679 |
| $\gamma_2$ | 530 | 70,24628 | 480,2177 | 139,9565 |
| $C_3$ | 9766,941 | 3,079251 | 458,1124 | 105,6972 |

# Chapter 2

# Lifetime predictions

## 2.1 Fatigue damage calculation

Before fatigue damage calculation is done. Input data needs to be prepared.

### 2.1.1 Fatigue-oxidation curves

#### Strain-life curves

Strain-life curves attained at high strain rate for few temperatures covering temperature range of a component under investigation are required for fatigue lifetime predictions. For now, due to limited number of tested specimens on current temperature, following extrapolation has been made

$$\epsilon_{ep_a} = a.N_f^b, \tag{2.1}$$

where $N_f$ are cycles to failure, $a(t)$ and $b(t)$ are temperature dependent coefficients.

Table 2.1: Temperature dependent coefficients of strain-life curves

| $t[°C]$ | $a$ | $b$ |
|---|---|---|
| 20 | 0,0068 | -0,122 |
| 400 | 0,0137 | -0,226 |
| 550 | 0,0259 | -0,29 |
| 650 | 0,0626 | -0,407 |
| 750 | 0,1483 | -0,568 |

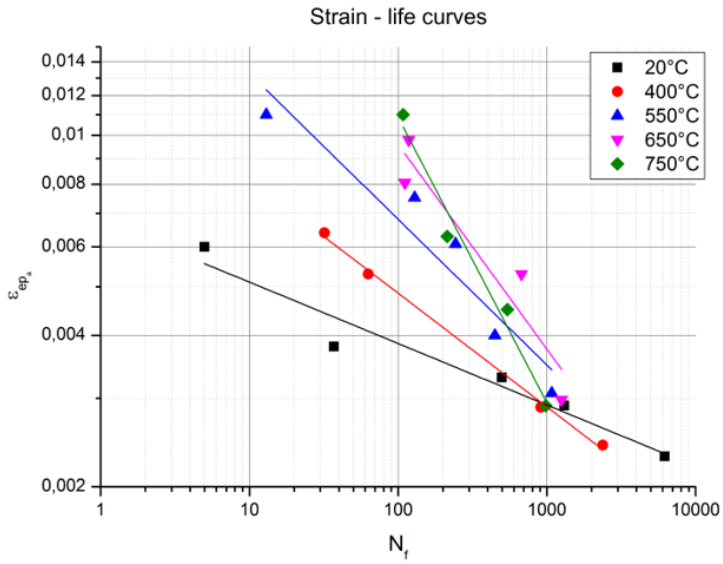


Figure 2.1: Strain-life curves

Coefficients for a temperature, where no tests were performed, can be gained by linear interpolation or analytical describtion for coefficients depending on the temperature can be found. However, piecewise cubic interpolation (PCHIP) seems to be most safe way.

### Energy-life curves

For lifiteme predictions based on dissipated plastic work, strain-life curves needs to be transformed to equivalent energy-life curves. Curves were transformed using hysteresis loops paramaters (stress amplitude) generated by elastoplastic model in ABAQUS

$$w = c_1.N_f^{c_2}, \tag{2.2}$$

where $w$ $[J.mm^{-3}]$ is dissipated energy of plastic work (for one full cycle),$N_f$ are cycles to failure, $c_1(t)$ and $c_2(t)$ are temperature dependent coefficients. For interpolation between temperatures, piecewise cubic interpolation has been used.

Table 2.2: Temperature dependent coefficients of energy-life curves

| $t[°C]$ | $c_1$ | $c_2$ |
|---|---|---|
| 20 | 8,6628 | -0,447 |
| 400 | 39,743 | -0,592 |
| 550 | 269,2 | -0,842 |
| 650 | 79,257 | -0,663 |
| 750 | 52,376 | -0,738 |

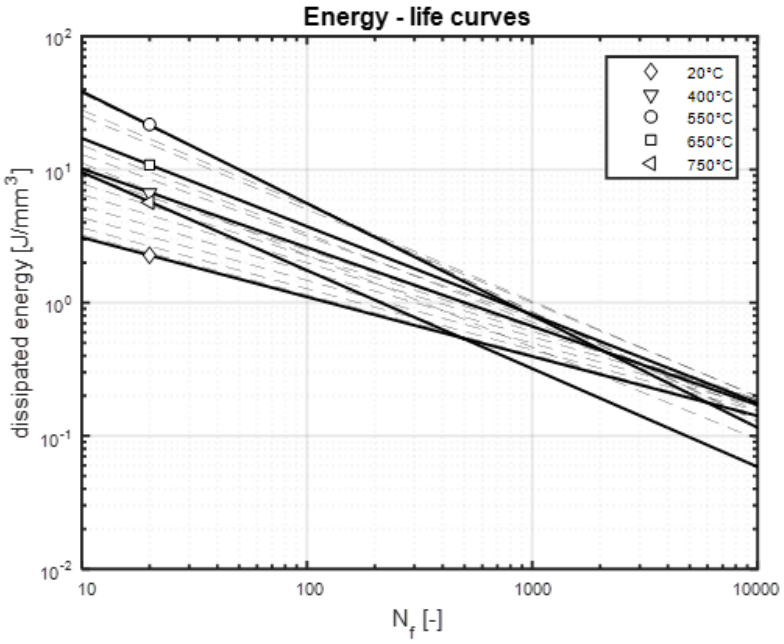


Figure 2.2: Energy-life curves, tested and interpolated

Relation between plastic strain amplitude and generated dissipated energy for one closed cycle is also needed for futher predictions. Dependency is described as

$$w = k_1(t).\epsilon_{pl_a}^{k_2(t)}. \tag{2.3}$$

For coefficients interpolation at temperature, at which was not FEM generation done, piecewise cubic interpolation has been used.

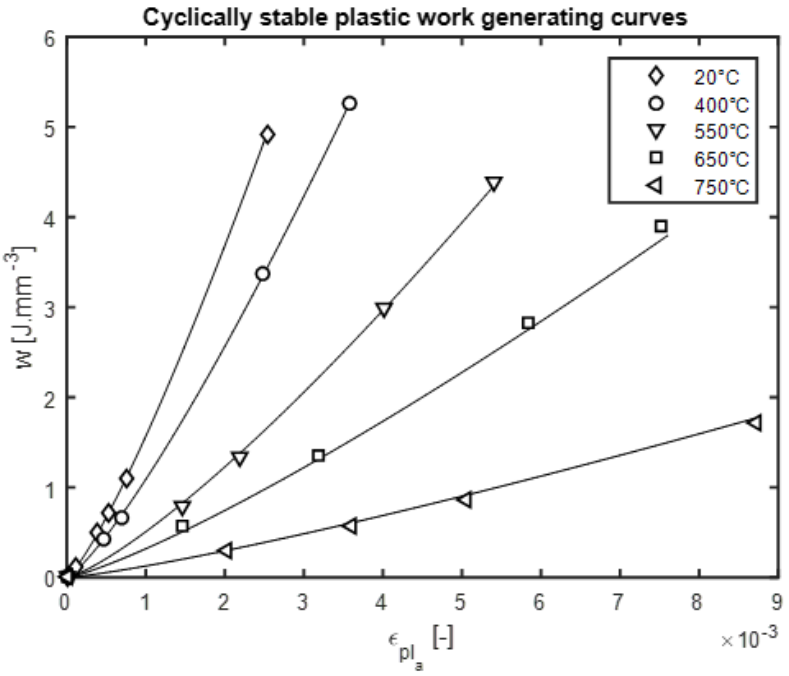


Figure 2.3: Energy-amplitude curves

Table 2.3: Energy-amplitude curves coefficients

| $t[°C]$ | $k_1$ | $k_2$ |
|---|---|---|
| 20 | 7745,778 | 1,231355 |
| 400 | 5526,505 | 1,234941 |
| 550 | 3321,872 | 1,271482 |
| 650 | 1459,237 | 1,21279 |
| 750 | 555,2598 | 1,219729 |

### 2.1.2 Fatigue lifetime predictions based on dissipated plastic work

Following fatigue life predictins will be done using dissipated plastic work as damage parameter. Required energy-life curves and energy-amplitude curves have been already presented. For correct and online damage calculation using following algorithm, instant plastic strain amplitude $\epsilon_{pl_a} = f(t)$ needs to be known. Let us presume that material (SiMo 4.06) obeys Masing and memory rules. Using these presumptions, extraction of amplitude of plastic strain (from FEM generated time history) can be done.

#### Masing behavior and memory rules

The construction of the stress-strain path resulting from a given stress-time or strain-time history only requires knowledge of the cyclic stress-strain curve (cssc). The path starts with an initial stress and strain value of zero. The segment resulting from the first peak simply follows the cssc. Upon load reversal, a half cycle of a hysteresis loop is formed that can be described by Masing behaviour. The model is completed by the definition of three memory rules:

- Memory 1 - if a hysteresis loop closes at the cssc, the further stress-strain path follows the cssc
- Memory 2 - if a hysteresis loop closes at a half cycle, the further stress-strain path follows this half cycle
- Memory 3 - a half cycle that was started at the cssc ends if the stress and strain value of the starting point with opposite sign is reached. The further stress-strain path then follows the cssc. The stress-time path shown in Figure 3.4 produces memory 3 at $X = -c$ because $|f| > |c|$, and at $Y = -f$ because $|g| > |f|$. Path between $X$ and $f$ is constructed using the cssc of the third quadrant.

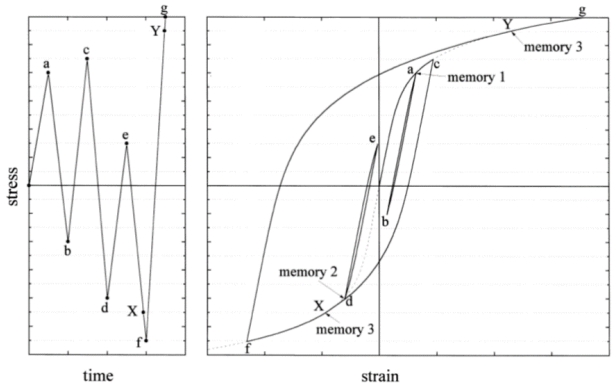


Figure 2.4: Memory rules (isothermal history) [2]

#### Plastic strain amplitude extraction

For non-isothermal histories, Masing and memory rules needs to be applied on strain history only. Loops don't necessarily close, when random thermomechanical loading occurs. This is because of temperature.

Online amplitude extraction will be presented using mentioned rules applied on strain history only, dealing with nested loops.

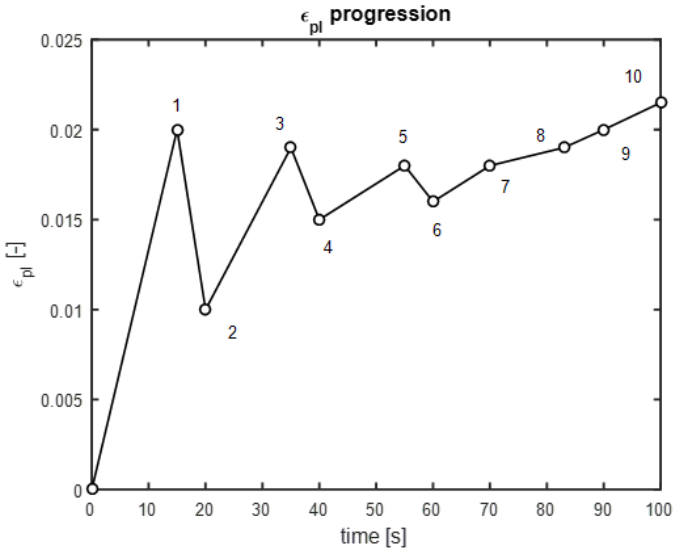


Figure 2.5: Plastic strain time history

- 0-1: Path starts at CSSC, amplitude is equal to plastic strain magnitude $\epsilon_{pl_a}(t) = \epsilon_{pl}(t)$.
- 1-2: Point 1 is strain reversal point, path continuous, tending to close hysteresis loop from 0-1, amplitude is equal amplitude in reversal point 1.
- 2-3: Point 2 is strain reversal point, path continuous with amplitude produced by last two reversals - 1 and 2 (equal half distance between reversals).
- 3-4: Point 3 is strain reversal point, amplitude is defined by reversals 2 and 3
- 4-5: Point 4 is strain reversal point, amplitude is defined by reversals 3 and 4
- 5-6: Point 5 is strain reversal point, amplitude is defined by reversals 4 and 5
- 6-7: Point 6 is strain reversal point, amplitude is defined by reversals 5 and 6
- 7-8: Point 7, memory is applied, amplitude is defined by reversals 3 and 4, loop 5-6 closed
- 8-9: Point 8, memory is applied, amplitude is defined by reversals 1 and 2, loop 3-4 closed
- 9-10: Loop 1-2 closed, Path continues on CSSC, amplitude is equal to plastic strain magnitude $\epsilon_{pl_a}(t) = \epsilon_{pl}(t)$.
- 10-...: Path continues on CSSC, amplitude is equal to plastic strain magnitude $\epsilon_{pl_a}(t) = \epsilon_{pl}(t)$.

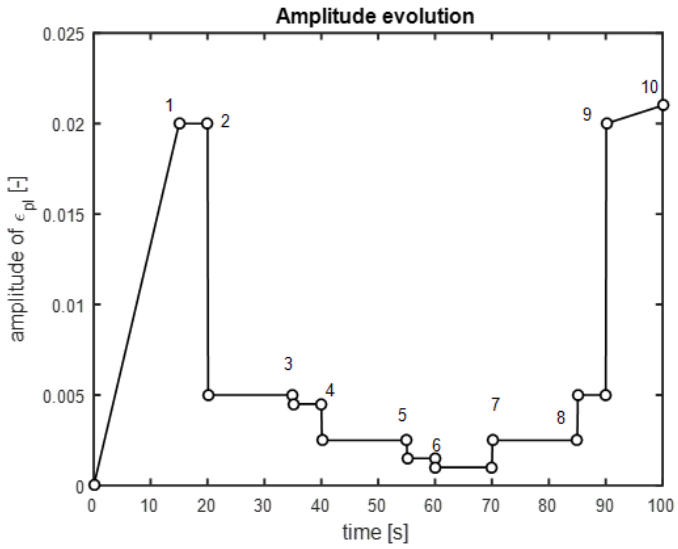


Figure 2.6: Extracted amplitudes

**Fatigue damage calculation**

Now, fatigue damage calculation is presented, using Palmgren-Miner linear damage accumulation rule. Nodal temperature, plastic strain and stress histories needs to be transfered from elastoplastic finite element method analysis. Incremental work done is defined as

$$dW = \sigma d\epsilon_{pl} \tag{2.4}$$

Actual $\epsilon_{pl_a}(t)$ is also know, using extraction algorithm presented. Amplitude defines actual dissipated energy $w(t)$, which cycle would have under current circumstances (dependency gained from plastic work generating curves (Figure 3.3)), temperature $T(t)$ needs to be also known, as $w$ also depends on temperature,

$$w = w(\epsilon_{pl_a}(t), T(t)). \tag{2.5}$$

Later will be shown, that $w$ defines output damage. Temperature for current time step in calculation is assumed constant, but only for current increment. Damage increment $dD_f$ for actual step can be obtained, using linear dependency of total work done $W_T$ on cycles to failure $N_f$ from current isothermal histories (Figure 3.7), as

$$dD_f = \frac{dW}{N_f.tan\ \alpha}, \tag{2.6}$$

where $dW$ is increment in plastic work done, $N_f = f(w((\epsilon_{pl_a}(t), T(t)))$, are cycles to failure for current energy $w(t)$, temperature $T(t)$ and amplitude $\epsilon_{pl_a}(t)$. Because total work done $W_T$ to failure can be obtained as

$$W_T = w.N_f, \tag{2.7}$$

then

$$tan\ \alpha = w, \tag{2.8}$$

and also

$$dW = dW_T = w.dN_f, \tag{2.9}$$

$w$ is energy of one cycle for current temperature and strain amplitude. Value of $N_f$ is obtained for current temperature and $w$ from energy-life curves (Figure 3.2). Linear depency is also applied for incremental values of work done. Now actual value of $dD_f$ is obtained as

$$dD_f = \frac{dN_f}{N_f(t)}, \tag{2.10}$$

$$dD_f = \frac{\sigma(t) d\epsilon pl}{N_f(t).w(t)}, \tag{2.11}$$

Value of $D_f(t)$ as time function is then

$$D_f(t) = \int_0^t dD_f. \tag{2.12}$$

Approach is applied on out-phase thermo-mechanical loading history in 4.2.

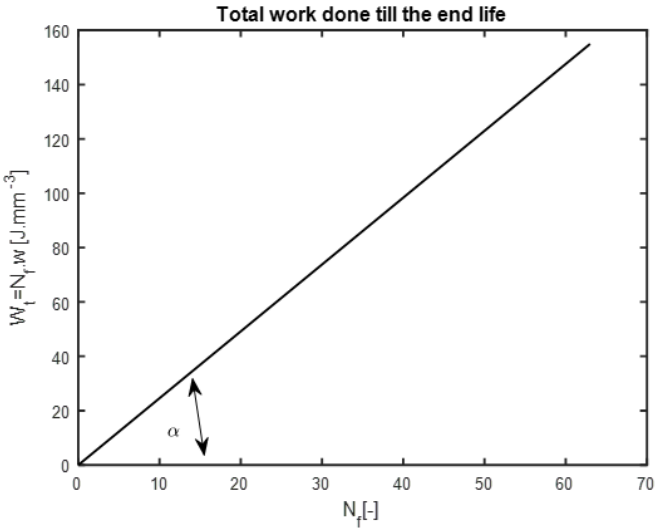


Figure 2.7: Total work done vs. cycles to failure

## 2.2 Creep damage calculation

Robinson's linear accumulation rule is used for further creep damage calculation

$$D_c(t_i) = \sum_{i=0}^{n_i} \sum_{j=0}^{m_{ij}} \frac{\Delta t_{ij} \sigma_{ij}}{t_{Rij}(T_i, \sigma_{ij})} + D_c(t_{i-1}), \tag{2.13}$$

where $\Delta t_{ij}$ and $t_{Rij}$ are the actual times under some loading and temperature and the corresponding time to rupture at the same loading and temperature, respectively. Index $i$ represents the number of the time step in load history and index $j$ represents additional subdivisions of time step $i$. Loading temperature T has to be higher or equal to creep temperature $T_C$ and loading stress $\sigma$ has to be higher than the temperature dependent elastic limit of the material $k(T)$ otherwise, no creep occurs. When the sum of individual creep damages reaches the defined limit value, usually 1, creep rupture occurs.

### 2.2.1 Creep master curves

Main objective is to determine master curves from the shortest creep-rupture tests, using extrapolation time to rupture for any combination of stress and temperature can be obtained.

Creep master curves were determined, using LM (Larson - Miller) parameter (Orr-Sherby-Dorn or Manson Brown paramaters or any other parameter can be used also). Larson - Miller parameter assumes that the logarithm of the time to rupture is inversely proportional to temperature

$$LM = T(logt_R + C). \tag{2.14}$$

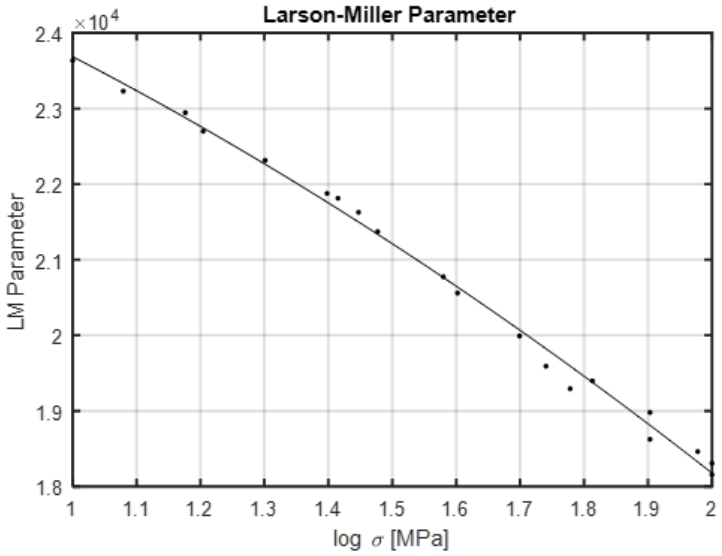


Figure 2.8: LM parameter

Time to rupture is than obtained as second degree polynom, dependent on stress and temperature

$$logt_R = -C + \frac{1}{T}(a_0 + a_1 log\sigma + a_2 log^2\sigma), \tag{2.15}$$

where $T[K]$ is temperature and $\sigma[MPa]$ is stress.

Table 2.4: Obtained polynom coeficients from least square method

| C | $a_0$ | $a_1$ | $a_2$ |
|---|---|---|---|
| 20 | 26960,21 | -2155,62 | -1117,56 |

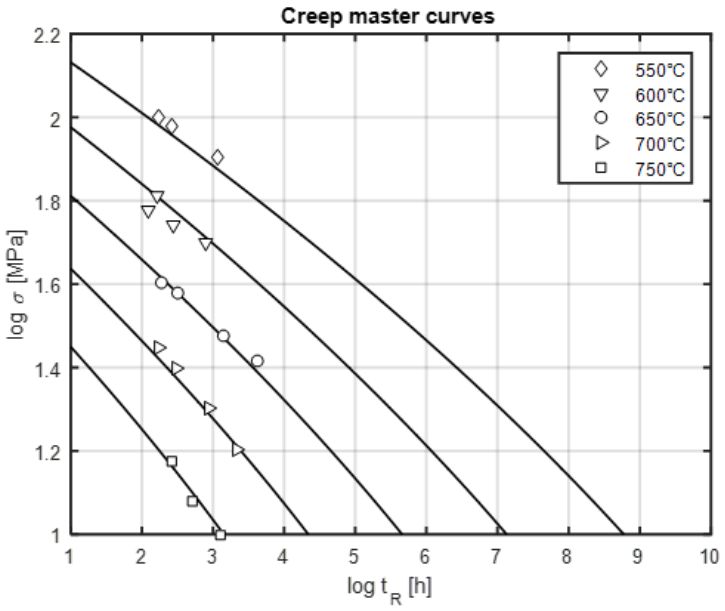


Figure 2.9: Creep Master Curves

Creep can be

- tensile only
- compressive-tensile
- or even compressive healing can occur.

How creep affects life, specifically of SiMo 4.06, needs futher more study. To be safe, compresive-tensile creep should be assumed.

## 2.3 Total damage calculation

Total damage at $t_i$ is defined as

$$D(t_i) = D_f(t_i) + D_c(t_i). \tag{2.16}$$

For better corespondence with reality, damage calculation should be done for first and for second loading cycle coresponding to FEA, as first cycle is more damaging. Second cycle is characterized by stabilization of stress-strain trajectory, $D(t_i)$ also tends to stabilize. Number of cycles to failure is than

$$N_f = \frac{1 - D_1 + D_2}{D_2}$$

or

$$N_f = 1, if D_1 > 1.$$

Damage for first and second cycle is defined as

$$\begin{aligned} D_1 &= D_{1f} + D_{1c} \\ D_2 &= D_{2f} + D_{2c} \end{aligned} \tag{2.17}$$

Further more study needs to be done on how plastic works affects life together with creep.

# Chapter 3

# Script for material model constants evaluation.

The script serves to evaluate parameters of Chaboche's phenomenological material model of plastic response described in chapter 1.1 with equations 1.1–1.5. It works on the basis of minimization of difference between experimentally determined and calculated *cyclic, stress-plastic strain curves.*

## 3.1 The function of script

To calculate cyclic stress/strain curves equation 1.7

$$\sigma_a = \sigma_Y + \sum_{k=1}^{N} \begin{cases} \frac{C_k}{\gamma_k} \tanh\left(\gamma_k . \varepsilon_{a\,pl}\right) & \text{for} \quad \gamma_k > 0 \\ C_k \varepsilon_{a\,pl} & \text{for} \quad \gamma_k = 0 \end{cases},$$

is used, at which parameters $C_k$ and $\gamma_k$ are expressed as temperature dependent according to equation 1.8

$$C_k\left(T\right), resp. \gamma_k\left(T\right) = \frac{a_1 - a_2}{1 + exp\frac{T - a_3}{a_4}} + a_2 \cdot$$

The model has $N$ backstresses and the last, $(N{+}1)^{\text{th}}$ backstress assumes $\gamma_{N+1} = 0$. There are $2 \times 4 \times N + 4$ parameters $\{P_j\}_{j=1}^{2\times4\times N+4}$. Let us arrange the parameters as algebraic vector

$$\mathbf{P} = [P_1,\, P_2. \ldots,\, P_{2\times4\times N+4}] \cdot \tag{3.1}$$

Then we can express

$$C_k\left(T\right) = \frac{P_{8\times(k-1)+1} - P_{8\times(k-1)+2}}{1 + exp\frac{T - P_{8\times(k-1)+3}}{P_{8\times(k-1)+4}}} + P_{8\times(k-1)+2}\,, \qquad \text{for} k = 1,\, 2,\, \ldots,\, N+1\,, \tag{3.2}$$

and

$$\gamma_k\left(T\right) = \frac{P_{8\times(k-1)+5} - P_{8\times(k-1)+6}}{1 + exp\frac{T - P_{8\times(k-1)+7}}{P_{8\times(k-1)+8}}} + P_{8\times(k-1)+6}\,, \qquad \text{for} k = 1,\, 2,\, \ldots,\, N \cdot \tag{3.3}$$

Now, stress amplitude of stabilized cycle under strain-controlled cyclic loading with constant plastic strain amplitude $\varepsilon_{a\,pl}$ (mean value $\varepsilon_m = 0$) at given temperature $T_i$, can be expressed via substituing equations 3.2, and 3.3 into 1.7 as function of parameters $\{P_j\}_{j=1}^{2\times4\times N+4}$.

$$\sigma_a = \sigma_a \left( \{P_j\}_{j=1}^{2\times4\times N+4}, T_i, \varepsilon_{a\,pl} \right) . \tag{3.4}$$

We have $Q$ cyclic, stress-plastic strain curves providing us with depences of measured stress amplitude on controlled strain amplitude at temperature $T_i$, $i = 1, 2, \ldots, Q$, each approximated on the basis of $M_i$ measured stabilized cycles represented as a sequence of points

$$\{[\tilde{\sigma}_{a\,T_i\,j}\,;\,\tilde{\varepsilon}_{a\,pl\,j}]\}_{j=1}^{M_i} .$$

Number of measured points at all temperatures, $M$, is

$$M = \sum_{i=1}^{Q} M_i \,,$$

so, we can express $M$ differences between calculated and measured stress amplitude at these points as a residua vector

$$\boldsymbol{\Delta\sigma}_a = \left[\Delta\sigma_{a\,T_1\,1},\, \ldots,\, \Delta\sigma_{a\,T_1\,M_1},\ldots,\, Delta\sigma_{a\,T_Q\,1},\, \ldots,\, \Delta\sigma_{a\,T_Q\,M_Q}\right] , \tag{3.5}$$

or

$$\boldsymbol{\Delta\sigma}_a = [\Delta\sigma_{a\,1},\, \Delta\sigma_{a\,2}\,, \ldots,\, \Delta\sigma_{a\,M}] \cdot \tag{3.6}$$

It is in evidence, that we can construct function mapping parameters $\mathbf{P}$ onto residua $\boldsymbol{\Delta\sigma}_a$, using measured cyclic, stress-plastic strain curves

$$\boldsymbol{\Delta\sigma}_a = \underbrace{\boldsymbol{\Delta\sigma}_a}_{\text{residua\_CDKT}} \left( \underbrace{\mathbf{P}}_{\text{paramsT}} , \underbrace{\left\{ \{[\tilde{\sigma}_{a\,T_i\,j}\,;\,\tilde{\varepsilon}_{a\,pl\,j}]\}_{j=1}^{M_i} \right\}_{i=1}^{Q}}_{\text{T\_eap\_sigma}}, \underbrace{N}_{\text{nback}} , \underbrace{m}_{\text{TYPE}} \right) . \tag{3.7}$$

To search parameters $\mathbf{P}$ we use non-linear least square method *leastsq* implemented in Python module *scipy.optimize*. It uses function taking array of optimized parameters and returning array of residua, and minimizes the sum of quadrats of residua, so,

$$\mathbf{P} = \arg\min_{\mathbf{P}} \left( \boldsymbol{\Delta\sigma}_a \cdot \boldsymbol{\Delta\sigma}_a \right) \tag{3.8}$$

## 3.2 Usage of script

The script is written in language Python, and uses modules *scipy.optimize*, *numpy*, *matplotlib.pyplot*, and *xlswriter*.

### 3.2.1 Inputs

should be writen in pre-scribed places of script denoted as remark with **USER EDITABLE AREA**. First, basic definitions should be done

```
#============================================================================
#  USER EDITABLE AREA
#  BASIC DEFINITIONS
#============================================================================

backStressNo = 3   # Number of backstresses: if backStressNo = i, than we have
#                     C1, Gamma1, C2, Gamma2, ... Ci, Gammai, C(i+1)
#
MATERIALNAME = 'SiMo CZ'
TASKNAME='CZ_2'
SHEETNAME = 'AA'
```

At this place number of backstresses is set and strings describing material and task. **MATERIALNAME** will appear at plots, **TASKNAME** and **SHEETNAME** constitute names of output figures and excel file with results. Next, measured data input as following structure

```
#============================================================================
#  USER EDITABLE AREA
#  DATA T_eap_sigma
#============================================================================
# Data represent N cyclic deformation curves (CDC) measured under N temperatures
#
# [[Temperature1, StrainAmplitudes1, StressAmplitudes1], ...,
#  [Temperaturei, StrainAmplitudesi, StressAmplitudesi], ...
#  [TemperatureN, StrainAmplitudesN, StressAmplitudesN]]
#
#  Temperaturei ...    number ... temperature of mesuring i-th CDC
#  StrainAmplitudesi   array  of Mi numbers
#  StressAmplitudesi   array  of Mi numbers
#
#  (StrainAmplitudesi[j], StressAmplitudesi[j]) j-th point of i-th CDC
#============================================================================
T_eap_sigma = [
[
20.,
[0.,    0.000166406,    0.000439653,    0.00057489,     0.000874943,      0.002748034],
[298.,  330.,           427.,        464.,           495.,             580.]
],
[
400.,
[0.,    0.000862059,    0.001140157,    0.002409895,    0.003438169],
[204.,  260.,           290.,           450.,           477.]
],
[
550.,
[0.,    0.001895626,    0.002417357,    0.004242806,    0.005293459],
[119.,  160.,           220.,           255.,           302.]
],
[
650.,
[0.,    0.001659135,    0.003624521,    0.006501653,    0.008173044],
[70.,   130.,           135.,           150.,           172.]
],
[
750.,
[0.,    0.002274266,    0.003562612,    0.004719857,    0.008468261],
[37.,   38.,            60.,            80.,            88.]
]
]
#============================================================================
```

The last user input is accomodation of $C_k(T)$, resp. $\gamma_k(T)$ form. Provided, Boltzmann form was found as appropriate, but user can change parameters **CENTERT** - temperature of inflexe of the dependence should be somewhere at the center of temperature ranges, **DELTACENTERT** - represents range at which inflexe point does not result into penalty, **DELTAT** - controls the steepnes of dependence. The smaller $a_3$, the steeper dependence. So, if $a_3 <$ DELTAT, penalty is generated.

```
#============================================================================
#   USER EDITABLE AREA
#   Function chaboche_par_T of chaboche parameters Ci(T), Gammai(T) temperature
#   dependence.
#============================================================================
# The dependence is expressed on the base of chaboche_T_npar temperature
```

```
# independent parameters for each chaboche parameter.
# The function, according to chosen form, may require some external definition.
#
# def chaboche_par_T(Parameters, T):
#    Defines Chaboche parameter Ci, or Gammai as function of temperature T.
#    It depends on parameters in list Parameters (number of parameters must be
#    stored in chaboche_T_npar)
#    The Parameters are variable. It changes outside to fit required response
#    (as CDC or hysteresis loop]
#    There may be external fixed parameters to modify the model, that are defined
#    together with (in front of) function
#    Function returns the value of chaboche parameter Ci, or Gammai (number) for
#    given parameters and temperature T, and penalty (number), that expresses
#    measure of violation of conditions. If penalty equals 1., no modification of
#    target function
#
#    In descriptions P0 = Parameter[0] ... Pi = Parameter[i]
#=============================================================================
# BOLTZMANN1
# (P0 - P1)/(1+e^((T - P2)/(P3))) + P1
#
# P2 ... temperature at inflex point
# P3 ... width of transfer interval
# CENTERT  ... expected temperature at inflex point
# DELTAT ..... expected width of transfer interval
#
# We require decreasing positive function <=> P1 > 0 and P0 > P1
#=============================================================================
chaboche_T_npar = 4
CENTERT = 400.
DELTACENTERT = 30.
DELTAT = 30.
#rellim = 0.
#centerlim   = 0.1
#klim = 100.

#=============================================================================
def chaboche_par_T(Parameters, T):
#=============================================================================
    return (Parameters[0]-Parameters[1])/(1.+np.exp((T-Parameters[2])/
    Parameters[3])) + Parameters[1]

#=============================================================================
def chaboche_par_pen(Parameters):
#=============================================================================
    Penalty = []
    if Parameters[0] > Parameters[1]:
        Penalty.append(0.)
    else:
        Penalty.append(Parameters[1]-Parameters[0])
    if Parameters[1] >= 0.:
        Penalty.append(0.)
    else:
        Penalty.append(-Parameters[1])
    if Parameters[2] < CENTERT-DELTACENTERT:
        parpen = CENTERT-DELTACENTERT-Parameters[2]
    elif Parameters[2] > CENTERT+DELTACENTERT:
        parpen = Parameters[2] - CENTERT-DELTACENTERT
    else:
        parpen = 0.
    Penalty.append(parpen)
    if Parameters[3] < DELTAT:
        Penalty.append(DELTAT-Parameters[3])
    else:
        Penalty.append(0.)
    return Penalty
```

### 3.2.2 Outputs

is provided in form of pictures with plots of parameters and stabilized cyclic plastic strain-stress curves, or as an excel sheet with plasticity data.

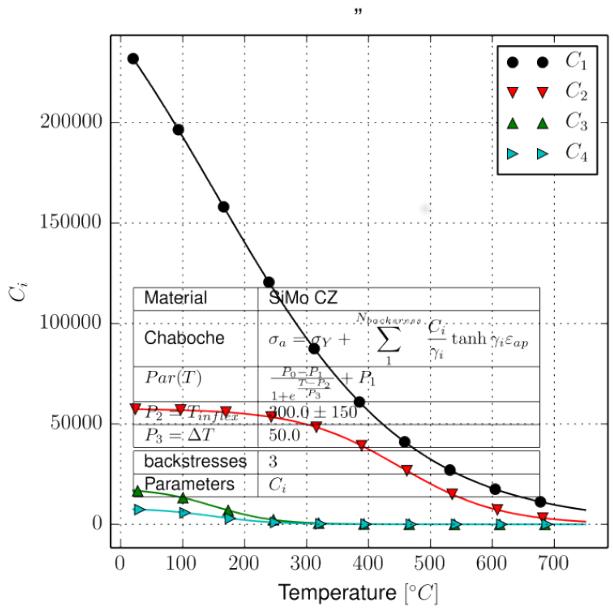


Figure 3.1: Example calibration, parameters $C_k\,(T)$.

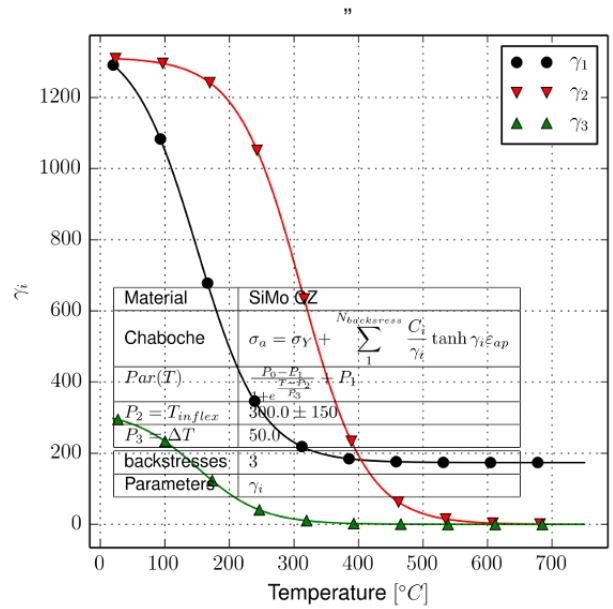


Figure 3.2: Example calibration, parameters $\gamma_k\,(T)$.

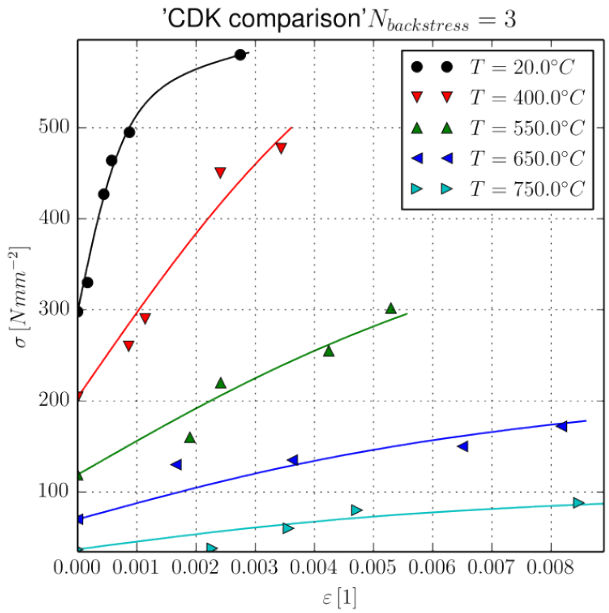


Figure 3.3: Example calibration, resulting cyclic stress-plastic strain curves

# Chapter 4

# Script for thermo-mechanical fatigue predictions

The script evaluates both low-cycle thermo-mechanical fatigue and creep damage during step representing, typically stabilized, operational cycle (fatigue step) in FEM model under software Abaqus by Dassault Systemes.

The evaluation follows chapter 2, so there is no need of another comments about theory.

## 4.1 Usage of script

The script is based on Python and Abaqus application interface API. It is dedicated to run on fatigue step containing time history of temperature (in Abaqus output variable TEMP), stress (in abaqus output variable S), and plastic strain (in abaqus output variable PE). The user is responsible for writing these variables into Abaqus output database (odb) in fatigue step, as well as for appropriate computational time sampling within the fatigue step. Fatigue step represents typically, but not necessarilly stabilized cycle.

### 4.1.1 Inputs

User must provide name of file with output database **resultodbname** (without extension), name of fatigue step **fatigue_step_name**, and for purposes of creep, the temperature and time units.

```
#==========================================================================
# MKP model
#==========================================================================
resultodbname = 'spec-center-direct-0-CR'
fatigue_step_name  = 'cycle-4'
# fatigue_domain = ''
TempQuant = 'C'   # C pro teploty ve st. Celsia, K pro teploty v Kelvinech
TimeQuant = 's'   # s pro sekundy, h pro hodiny
#==========================================================================
```

Further, material data for LCTMF in form of coefficients $k_1$, resp. $k_2$ given in arrays **k1**, resp. **k2** corresponding with temperatures of single cyclic stress-plastic strain curves, at which fatigue life was determined - **LCF_temperatures**, that

describe relation between plastic strain amplitude $\varepsilon_{a\,pl}$ and density of energy dissipated per cycle via plastic strain

$$w = k_1 * \varepsilon_{a\,pl}^2 + k_2 * \varepsilon_{a\,pl}$$

It is not in correspondence with equation 2.3, nevertherless quadraratic dependence can be calibrated as almost identical, and provides higher performance of script. Parameters $c_1$, resp. $c_2$ given in arrays **c1**, resp. **c2** corresponding with temperatures **LCF_temperatures** corresponds with $c_1$, resp. $c_2$ in equation 2.2.

```
#==========================================================================
#Material data by experiments
#==========================================================================
# Low cycle fatigue (LCF)
# Calibration temperatures
LCF_temperatures = [20.,400.,550.,650.,750.]

#Dependence between amplitude of plastic strain and dissipated energy
#                 k1*e_a_pl^2+k2*e_a_pl=w_dis_merna,

# coefficients k1 and k2 at calibration temperatures
k1=[135913, 98889, 45636, 17829, 14018 ]
k2=[1396.2, 1071.4, 558.02, 389.51, 252.88]

# LCF dependence on dissipated energy
#              w_diss_merna=c1*Nfatigue^c2
# konstanty c1 a c2 z unavovych krivek pro jednotlive teploty
c1=[10.1, 31.689, 174.14, 168.75, 186.91]
c2=[-0.406, -0.531, -0.773, -0.788, -0.929]
```

At least, material data for creep in form of creep experimental temeratures array **CRP_temperatures**, elastic limit of material as array **kT** corresponding with **CRP_temperatures**, parameters **a0**, **a1**,**a2**, and **C** approximating Larson-Miller parameter and providing time to failure under creep. It corresponds with $a_0$, $a_1$, $a_2$, and $C$ in equations 2.2.1, and 2.15. Parameter **TCRP** represents so, called creep temperature.

```
#==========================================================================
# Creep
# Kalibracni teploty
CRP_temperatures = [550., 600., 650., 700., 750.]
kT = [ 50., 40., 12., 8., 3.]
a0 = 26960.21092
a1 = -2155.615956
a2 = -1117.558112
C  = 20.
TCRP=450.

#==========================================================================
```

Now, the script can be run in Abaqus environment using *run script* command in *file* menu.

### 4.1.2 Outputs

are the fields of LCTMF damage, creep damage, and total damage per fatigue step as field outputs in session step.

# Bibliography


[1] Španiel, M. - Bartošák, M.: Experimental determination of Manson-Coffin curves of Turbochargers Case Material. [Výzkumná zpráva]. Praha: ČVUT v Praze, FS, 2014. 12105/2014/27. 24 s. (in Czech).

[2] ANTHES,R.J. Modified rainflow counting keeping the load sequence. International Journal of Fatigue, Volume 19, Issue 7, October 1997, Pages 529-535, ISSN 0142-1123. http://dx.doi.org/10.1016/S0142-1123(97)00078-9.

[3] NAGODE, M., HACK, M., FAJDIGA, M. (2010). Low cycle thermo-mechanical fatigue: damage operator approach. Fatigue and Fracture Of Engineering Materials and Structures, 33(3), 149-160. doi:10.1111/j.1460-2695.2009.01424.x.

[4] Ĺ ERUGA, Domen, NAGODE, Marko. Unification of the most commonly used time-temperature creep parameters. Mater. sci. eng., A Struct. mater.: prop. microstruct. process.. 2011, vol. 528, no. 6, pp. 2804-2811, doi: 10.1016/j.msea.2010.12.034.

[5] NAGODE, Marko. Continuous damage parameter calculation under thermo-mechanical random loading, MethodsX, Volume 1, 2014, Pages 81-89, ISSN 2215-0161, http://dx.doi.org/10.1016/j.mex.2014.07.004.

[6] ABAQUS/CAE 6.14. User's Manual. Online Documentation Help: Dassault Systemes.