CreeSo – software for creep simulation of complex alloys

Gerold Zuderstorfer*, Florian Riedlpergera and Bernhard Sondereggerb

*Institute of Materials Science, Joining and Forming, Graz University of Technology, Graz, Austria; bInstitute for Engineering Materials
Metals and Alloys, Johannes Kepler University (JKU), Linz, Austria

Summary
The MDC-model (Microstructural Dislocation Creep – model) is an advanced approach incorporating numerous creep mechanisms in complex alloys. It leads to a differential equation system that cannot be solved in a closed algebraic form. Also, some parameters of these equations cannot be measured directly. The application CreeSo (from ‘Creep Software’) helps finding numerical solutions of this model and it can also help determining missing parameters by comparing simulated with experimental creep curves. CreeSo is not limited to the MDC-model but open to any model formulated in its script language. The application is written in C++ and JavaScript and it will be available as a desktop version for MS-Windows and as an online version for web browsers.

Introduction
The modelling group at IMAT (Institute of Materials Science) at the Graz University of Technology has been involved in the modelling of material properties at the microstructural level for many years. The prediction of the rupture time due to creep stress is of particular importance, since creep experiments are expensive due to their long duration. Therefore, there has long been a desire to reduce the number of experiments and to extrapolate the rupture times for other load cases from simulations on the basis of a few experiments. Ideally only one experiment is needed.

CreeSo-MDC-model
There are essentially two approaches to the mathematical description of creep: phenomenological and physical. In the phenomenological description, the only aim is to match the measured curves as closely as possible. The parameters of the equations do not need to have a physical meaning. Physical models, on the other hand, are based on the description of processes in the material. If possible, the parameters of the equations are measurable quantities. In this work, we use a physically based dislocation creep model which we refer to as CreeSo-MDC-model.

Creep is mostly caused by migration of dislocations. The velocity of the dislocations is influenced by the microstructure of the material in question (aside from material temperature and stress).

Precondition for a predictive creep model is a physically based model for the effective dislocation velocity. In addition, we need a model for the dislocation density evolution. Then, the two quantities have to be linked to the actual plastic deformation. This can be achieved by e.g. Orowan’s law [1], which relates mobile dislocation density \( \rho_m \) and glide velocity \( v_g \) to the creep strain rate \( \dot{\epsilon} \), as can be seen in Equation (1). \( M \) stands for the Taylor factor and \( b \) is the Burgers vector.

\[
\dot{\epsilon} = \frac{b}{M} \cdot \rho_m \cdot v_g
\]

Equation (1) Orowan’s Law [1]

If the evolution of mobile dislocations and the kinematics are known, primary and secondary creep can be calculated on a physical basis. The evolution of mobile dislocations is subsequently based on self-reactions, interactions with other types of dislocations (statics or boundaries e.g.) and with a variety of populations of precipitates. The type of interaction may be generation, annihilation or locking and can therefore lead to both increase or decrease of the mobile dislocation density over time. In any case, the degradation of the material in creep environments can be accurately simulated and a numerical prediction of the lifetime is possible, depending on temperature, stress and microstructural features (which can be tuned by alloying adjustments and thermo-mechanical treatments).

The details of our model, including all microstructural interactions, are described in reference [2] and [3]. All input parameters of our model have a physical meaning, and most can be measured from the material’s ‘virgin’ starting microstructure (before creep) or they can be found in literature. However, some parameters do have

CONTACT Gerold Zuderstorfer gerold@zuderstorfer.com Institute of Materials Science, Joining and Forming, Graz University of Technology, Graz 8010, Austria
© 2022 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.
a physical background, but are not directly accessible by measurements. In these cases, they need to be fitted, although a reasonable (physically possible) range for these values can be mostly defined. This is a task where CreeSo can assist.

In previous simulation applying our model [2], the software MatLab has been used, which also has some disadvantages. For this reason we have been developing our own software with the project name 'CreeSo' (from 'Creep Software') within the past 3 years. The software is programmed in C++ and HTML5 and was originally planned as a desktop version only. Due to the ongoing Covid situation, we decided to change our plan. The fact that face-to-face meetings are problematic and that most employees therefore work from home led to the decision to introduce the online version in addition to the desktop variant.

**Application**

CreeSo software aims at numerically solving nonlinear differential equation systems. However, CreeSo is not limited to creep simulations. In principle, a variety of tasks in science and technology can be modelled as well, e.g. chemical plants, biological growth kinetics, trajectory calculations and many more. You can get more information via the link in Figure 1.

The software is divided into a modelling and a calculation part. The variables and equations are defined in the modelling part and numerical calculations are carried out using this model in the calculation part. However, the majority of users will fall back on models that have already been made available in the form of modules and will probably not create their own models.

**Modelling part**

The first step is to define the constants and variables of the system of equations. In addition to the variables and constants which are actually part of the physical model, following variables must be included:

- The ‘loop counter’, usually designated with the variable name ‘i’.
- The ‘control variable’, in most cases this is the time.
- The ‘control increment’, in most cases this is a variable containing the time steps per iteration (accordingly named ‘TIME_STEP’ or the like).
- If variable time steps are to be used, a variable for a ‘general change rate’ must also be specified.

Furthermore, if there are derivatives of variables used, they too have to be defined. For example: The strain rate is the first derivative of the strain over the time.

In addition to the name, symbols for a mathematical representation, a unit of measurement and a description can be specified for constants and variables (Figure 2).

In the next step the equations of the system have to be defined (Figure 3). The actual program code for the simulation is then later automatically deduced from this equation system. Within the software, the equations can also be displayed in mathematical notation. One can optionally also add a description or annotation to them. These options enable a full documentation of the model, and help other users to understand the physical background.

In the last step of the modelling process, after the model is prepared, it can be compiled to C++ or JavaScript and form a module, which can then be run by the simulation software. This way, individual models can be easily exchanged between different users of the software.

**Calculation part**

In the calculation part one first has to select an existing model as a module. Then the calculation can be started and the course of individual variables can be plotted (Figure 4 and 5). The result can also be exported as a table for further processing. The values for preselected variables and constants can be set before the calculation and further data can be imported in the form of tables. This is important for the MDC creep model, for which data from a thermo kinetic calculation previously made with the software MatCalc [4] can be imported.

There are also functions for automatic parameter finding and sensitivity analysis, but at the time of this publication these are still in development and without a GUI. However, the automatic parameter determination has already been used successfully on the MDC creep model.
Design of the software

When planning the software, attention was paid to the durability of the components and high effective computing speed. Furthermore, the software architecture was designed to implement both a desktop and an online version without major changes to the code. Although the desktop version was planned for the Windows operating system from the start, a high level of platform independence was also desired.

C++ was therefore chosen as the programming language for the computationally intensive parts in the back-end, as the programming language guarantees maximum execution speed and very good platform independence. Because of the desire for an online version, the decision was made to use HTML5 for the GUI (Graphical User Interface). Figure 6 now shows how these components are connected.

For the online version, a web server with a 'Common Gateway Interface' (CGI-BIN) is used, which connects the front-end (browser with HTML5) with the back-end (compiled C++ code).

The desktop version also uses a web browser for the GUI. Therefore, the Windows application itself has a small web server (http server) integrated. However, the users don’t see much difference to the online version, because this server is started with the application and also ended with it. The users do not have to worry about the configuration of the server either, as it is set up exclusively for use with CreeSo. Access from outside the computer is not possible. This feature is vital especially for users operating with sensitive data with respect to intellectual property.

Both variants, desktop and online, have their individual advantages and disadvantages.

Advantages of the online version:

- Easier entry because there is no need to install a software.
- Data and models can be shared more easily with others.

Advantages of the desktop version:

Figure 2. Variables of a model.
No computational load on the server. Calculations can therefore run for several hours on the local computer without affecting the server.

Not all users trust online versions. With a desktop version the users’ data never get to the Internet, but always remain on their local computer.

Modularity

As already mentioned above, the models can be compiled in their own modules. These are then linked to the software before the calculations are carried out – see Figure 7.

Our initial approach provides that the models are translated into C++ code and compiled into a DLL (‘Dynamic Link Library’) under Windows or into a SO (‘Shared Object’) under Linux. However, a benchmark test we recently carried out showed that calculations performed directly in the browser using the JavaScript programming language also provide sufficiently good computing speed for most purposes. As an alternative, we are therefore testing the possibility of compiling modules to JavaScript code and making them available in the browser for calculations – as depicted in Figure 8.

This combines the advantage of the desktop version, that there is still no load on the server due to long calculations, with the advantages of the online version. With this design, we can also use CreeSo in exercises for a larger number of students or make the software available to the audience directly during a presentation.

The modularity of CreeSo makes extending the software rather easy. It is planned to add further creep models and models of other material properties. Also an expansion to other scientific fields aside from material science makes sense.

Test cases and future goals

The MDC-model was tested in a prototype of the software on martensitic 9–12% chromium steels and showed very good agreement to the test data.
Figure 4 and 5. Selection of a model and plot results of the calculation (CreeSo reproduces the results presented in [2]).

Figure 6. Desktop vs. online version of CreeSo (Additional images [5] and [6]).

Figure 7. Connecting modules on the server side.
One successful application of the model was demonstrated for P91 at 650°C, where a modelled master creep curve at 70 MPa from [2] is shown in Figure 9. A paper on P92 has recently been published [3].

At the moment the group is improving the automatic parameter optimisation of the software, which allows a fast adaption of the model to other materials. The group is also working on an adaption of the MDC-model for nickel-based alloys and hopes to publish the results soon.

For the long-term future, we try to expand the boundaries of the software further to describe hot deformation and other changes in material properties under different loading conditions.

Conclusions

CreeSo, both desktop and online version, reproduces the results of our MatLab version. However, CreeSo features significant advantages of creep simulation and modelling compared to the previously used simulation code in MatLab:

- Ease of use through a GUI (Graphical User Interface).
- Use of different models due to modular structure.
- No external licences are required for the product.
- Increased computing speed by a factor of 10 to 20 even in single-thread mode.
- Further increase in computing speed in multi-thread mode for calculations that can be processed in parallel.
• High flexibility in the software architecture: Support for a desktop version and an online version with calculation on the server or in the browser possible.
• The online version can also be used on platforms other than MS Windows.
• The modularity of the software allows for testing a variety of models on one set of actual creep data.
• The software modules (containing individual creep models) can be easily shared with people outside the workgroup.

Acknowledgments

We gratefully acknowledge funding from Austrian Science Fund (FWF) within project “Software Development on Dislocation Creep in Alloys” (P-31374). We also like to thank Graz University of Technology and Johannes Kepler University Linz for providing the IT infrastructure.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the Austrian Science Fund.

References

[1] Orowan E. Problems of plastic gliding. Proc Phys Soc. 1940;52:8–22.
[2] Riedlspurger F, Krenmayr B, Zuderstorfer G, et al. Application of an advanced mean-field dislocation creep model to P91 for calculation of creep curves and time-to-rupture diagrams. Materialia. 2020;12:100760.
[3] Riedlspurger F, Zuderstorfer G, Krenmayr B, et al. Application of a physically-based dislocation creep model to P92 for constructing TTR diagrams. Mater High Temp. 2022;39:161–166.
[4] Matcalc - solid state and kinetics precipitation. Austria; [cited 2021 May 30]. Available from: https://www.matcalc.at/
[5] Image “Desktop computer”. Austria; [cited 2021 May 30]. Available from: https://pixabay.com/de/vectors/computer-hardware-monitor-wohnung-24882/
[6] Image “Earth”. Austria; [cited 2021 May 30]. Available from: https://pixabay.com/de/illustrations/erde-globus-planeten-karte-3087437/