Applications in Engineering Science 9 (2022) 100080

Contents lists available at ScienceDirect

Applications in Engineering Science

journal homepage: www.elsevier.com/locate/apples

Extending a physics-based constitutive model using genetic programming

Gabriel Kronberger [a,*], Evgeniya Kabliman [b], Johannes Kronsteiner [c], Michael Kommenda [a]

[a] Josef Ressel Center for Symbolic Regression, University of Applied Sciences Upper Austria, Softwarepark 11, 4232 Hagenberg, Austria
[b] Technical University of Munich, TUM School of Engineering and Design, Department of Materials Engineering, Chair of Materials Engineering of Additive Manufacturing, Boltzmannstr. 15, 85748 Garching, Germany
[c] LKR Light Metals Technologies, Austrian Institute of Technology, Giefinggasse 2, 1210 Vienna, Austria

ARTICLE INFO

MSC:
68T05
74-04
74-05
74-10
74C99

Keywords:
Symbolic regression
Genetic programming
Material modelling
Flow stress

ABSTRACT

In material science, models are derived to predict emergent material properties (e.g. elasticity, strength, conductivity) and their relations to processing conditions. A major drawback is the calibration of model parameters that depend on processing conditions. Currently, these parameters must be optimized to fit measured data since their relations to processing conditions (e.g. deformation temperature, strain rate) are not fully understood. We present a new approach that identifies the functional dependency of calibration parameters from processing conditions based on genetic programming. We propose two (*explicit* and *implicit*) methods to identify these dependencies and generate short interpretable expressions. The approach is used to extend a physics-based constitutive model for deformation processes. This constitutive model operates with internal material variables such as a dislocation density and contains a number of parameters, among them three calibration parameters. The derived expressions extend the constitutive model and replace the calibration parameters. Thus, interpolation between various processing parameters is enabled. Our results show that the implicit method is computationally more expensive than the explicit approach but also produces significantly better results.

1. Introduction and motivation

Mathematical models are at the core of science and engineering and allow us to predict physical phenomena without direct observations. Only through modelling and simulation we are able to build extremely complex and safe physical objects (such as air planes, space vehicles, or power plants). In empirical modelling one can distinguish white-box and black-box models with a whole spectrum of grey-box models between these two extremes (Sjöberg et al., 1995; Von. Stosch et al., 2014). White-box models can be derived from physical principles and have interpretable parameters with a physical meaning (e.g. Planck's constant, Avogadro's constant). The internals of white-box models are known and can be understood. Black-box models establish a functional mapping from inputs to outputs by fitting to observations whereby the internals of the model are irrelevant or unknown. Therefore, the internal parameters of black-box models have no physical meaning (Sjöberg et al., 1995). Examples of black-box models are non-parametric statistical models (i.e. all kernel-methods including support vector machines, Gaussian processes, LOESS), neural networks, and tree ensemble methods (e.g. random forest, gradient boosted trees). Grey-box models also establish a functional mapping from inputs to outputs by fitting to observations but have only a few parameters and simple interpretable equations.

One possible approach for the identification of grey-box or potentially even white-box models is symbolic regression (SR) with genetic programming (GP). The main aim of SR is finding well fitting model equations (structure) as well as their parameters (Koza et al., 1993). SR offers the possibility of interpretability which can be achieved by including model complexity as an optimization criterion additionally to model fit. A popular approach for solving SR problems is GP, which is an evolutionary algorithm that evolves computer programs. It uses a population of solution candidates (programs) which is iteratively improved by mimicking processes observed in natural evolution, namely survival of the fittest, recombination, and mutation (Koza, 1992). Programs that are better with respect to an objective function (also called fitness function in GP) are selected with a higher probability for recombination while bad programs have a low probability to be selected.

SR models can be easily integrated into mathematical models regardless of the modelling software environment because the commonly used operators and functions are readily available in most standard libraries.

SR is particularly suited for the integration of physical principles and offers the possibility to produce interpretable models which can

* Corresponding author.
E-mail address: gabriel.kronberger@fh-hagenberg.at (G. Kronberger).

https://doi.org/10.1016/j.apples.2021.100080
Received 19 October 2021; Received in revised form 18 November 2021; Accepted 18 November 2021
Available online 20 December 2021

be integrated easily into existing software frameworks making it a particularly interesting approach for scientific machine learning (Baker et al., 2019) tasks.

In the present work, we apply SR and GP to extend a constitutive model used to describe the materials response to applied forces, which is a core of numerical metal forming simulations. The main research question thereby is whether such an extended constitutive model achieves accurate enough simulation performance when evaluated with varying process conditions.

2. Objectives

The present study is a continuation of the work published in Kabliman et al. (2021) with a goal to improve the constitutive models using GP and SR. The used constitutive model (Kabliman et al., 2019) is white-box and is based on physical laws. It describes the material flow behaviour in terms of internal state variables such as dislocation density. Among the physical values, it contains a number of calibration parameters, which depend on processing conditions (e.g. deformation temperature and strain rate) and are normally fitted to experimental data. To overcome the required regular fitting, the expressions which correlate the calibration parameters to processing parameters should be derived. This task can be addressed using GP and SR. In Kabliman et al. (2021) we have used the following approach: the calibration parameters were first optimized using a global optimizer, and then the formulas were identified for predicting the optimized values using SR. We call this approach the *explicit* method.

Recently, Asadzadeh et al. (2021) have described a hybrid modelling approach based on symbolic regression whereby the model structure is partially fixed. They have used the approach to extend a physics-based model for a sheet bending process whereby symbolic regression was used to evolve additional terms for the model. They found that partially fixing the structure leads to more consistent symbolic regression results (higher repeatability) and lower model complexity. Additionally, the hybrid modelling approach required only few data points.

In the present paper, we propose an approach similar to Asadzadeh et al. (2021), which can evolve the required formulas (extensions) to the constitutive model directly. This may improve the knowledge of the modelled system or process since the functional dependency of calibration parameter values from processing conditions is explicitly established and may be interpreted. We call this approach the *implicit* method. The difference to Asadzadeh et al. (2021) is that we use the approach for a constitutive model which requires simulation to fit stress–strain curves. In the implicit approach the fixed physics-based part is hard-coded in the simulation model and extensions are evolved using multi-tree genetic programming. Additionally, we demonstrate the approach on measured stress–strain curves instead of generated data.

To compare the results from both (explicit and implicit) approaches, we use the same formulation of the used constitutive material model and the same experimental data as in Kabliman et al. (2021). For better understanding, we also give a detailed description of the (explicit) modelling method used before.

3. Background

3.1. Data collection

To generate a data set, a series of hot compression tests was conducted for the aluminium alloy AA6082 as described in Kabliman et al. (2019) and Kabliman et al. (2021). The cylindrical samples (5 mm diameter by 10 mm) were compressed using a deformation and quenching dilatometer DIL805/A from TA Instruments. Compression tests were performed up to a strain of 0.7 at various temperatures and average strain rates as summarized in Table 1. Invalid measurements at the beginning and end of a test, when the machine resets, were removed. The specimens were induction-heated inside of the dilatometer to the prescribed temperature and afterwards compressed at an average constant strain rate and controlled temperature. For each set of deformation parameters (T and $\dot{\varphi}$), two identical tests were performed and the average values were calculated. Measurements from hot compression tests are assigned to training and testing sets as shown in Table 1. The training set is itself partitioned into four parts for four-fold cross-validation.

Table 1
Assignment of hot compression tests to training and testing sets. The training data set is split into four partitions for four-fold cross-validation. The values in each cell indicate the assignment to the partitions.

		$\dot{\varphi}[1/s]$				
		0.001	0.01	0.1	1	10
Temperature [°C]	350	Test	4	3	Test	2
	375	1	Test	4	3	Test
	400	2	1	Test	4	3
	425	Test	2	1	Test	4
	450	3	Test	2	1	Test
	475	4	3	Test	2	1
	500	1	4	3	Test	2

The stress and strain values were derived during the measurement using the following calculation formulas:

$$\mathrm{kf}_i = \frac{F_i}{A_i^{cs}},\ A_i^{cs} = \frac{\pi d_0^2 L_0}{4 L_i},\ \varphi_i = \ln\frac{L_0}{L_i},\ \dot{\varphi}_i = \frac{d\varphi}{dt}, \tag{1}$$

where i indexes subsequent measurements, kf_i is the stress, F_i is the measured force, A_i^{cs} is the actual cross section of the sample, φ_i is the strain, L_0 is the initial length at the start of the deformation segment, L_i is the actual sample length, $\dot{\varphi}$ is the strain rate, dt is the time difference between two measurement points, and d_0 is the initial sample diameter.

Fig. 1 shows the processed stress–strain data.

3.2. Constitutive model

Typically, a constitutive model is represented by a sum of an initial yield (threshold) stress, σ_y, and a strain-dependent part, σ_p. While the threshold stress is usually a guess value, the strain-dependent stress might be written as a power law-form or in terms of material internal state variables. The latter approach is based on physical laws and, therefore, easily interpretable. The model used in present work is based on the evolution of mean dislocation density, ρ, and is called Mean Dislocation Density Material Model (MD2M) (Kabliman et al., 2019).

$$\sigma = \sigma_y + MGb\left[\frac{\sqrt{\rho}}{2} + \frac{1}{\delta}\right] \tag{2}$$

Here M is the Taylor factor, G is the shear modulus, b is the norm of the Burgers vector and δ is a mean sub-grain size. By deformation at a temperature, T, and a strain rate, $\dot{\varphi}$, the dislocation density will change according to the following equation:

$$d\rho = \frac{M\sqrt{\rho}}{bA}\dot{\varphi}dt - 2BM\frac{d_{ann}}{b}\rho M\dot{\varphi}dt - 2CD\frac{Gb^3}{k_B T}\left[\rho^2 - \rho_{eq}^2\right]dt \tag{3}$$

The first term describes the increase of dislocations and the next two terms correspond to the dislocations reduction. The first recovery process happens when two antiparallel dislocations come to a critical distance, d_{ann}. The second recovery process is thermally-activated and controlled by a self-diffusion along the dislocations, D. When the processing conditions allow, the material can recover down to an equilibrium state described by the equilibrium dislocation density, ρ_{eq}.

The model contains several physical parameters (e.g. the Boltzmann constant, k_B), which can be found in Kabliman et al. (2021). Besides

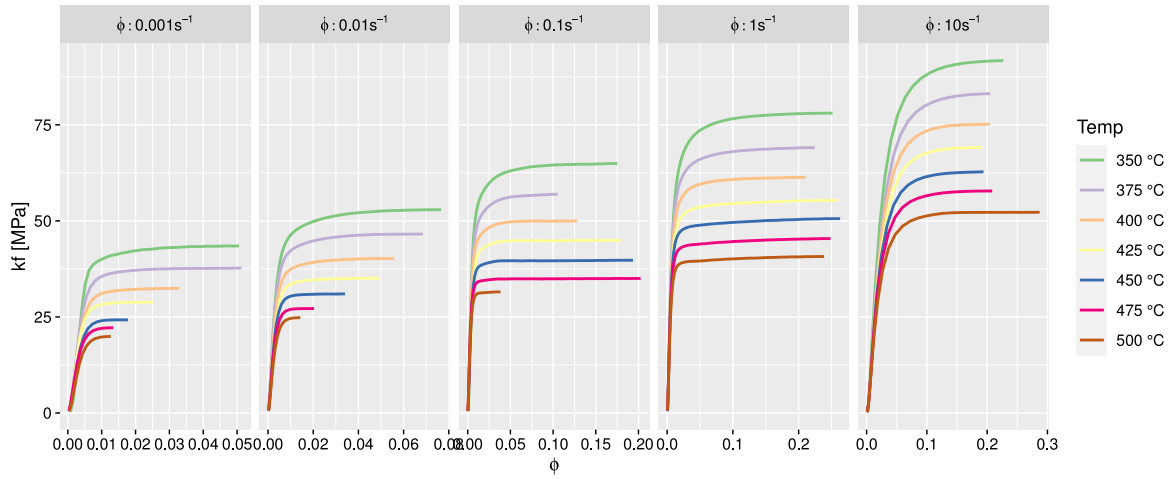

Fig. 1. Stress–strain curves acquired from the hot compression tests. Only measurements up to the maximum stress value are used for modelling.

them, there are three calibration parameters A, B and C. Their values depend on the material, its state and deformation conditions, but these dependencies are not fully understood. Therefore, these parameters are normally tuned using the experimental stress–strain curves.

From inspection of the white-box model, it can be determined that A, B, and C must be positive, because, otherwise, Eq. (3) has no physical meaning. Furthermore, B and C depend on the scale of A, since the relative contributions of the terms must be of similar size. To simplify the optimization and prediction of the parameters, we use an alternative parameterization u, v and w with

$$A^{-1} = \exp(u),\ B = \exp(v)A^{-1},\ C = \exp(w)A^{-1}, \tag{4}$$

and limit the search space to $u \in [-15, 0]$, $v \in [-15, 15]\, w \in [-15, 0]$. The transformation stretches the search space non-linearly, whereby the space becomes exponentially larger as A, B or C approach zero. The value for A cannot reach zero. This guarantees a physically feasible solution. The domain for u, v and w is set based on the range of values, which are plausible for the computation material scientists.

4. Methods

4.1. Quantification of the model accuracy

To measure the accuracy of developed model extensions, we use the sum of mean of squared errors (SMSE) over all tests for the training and testing sets separately. The measurement frequency is the same for tests with different strain rates, which implies that the data set has a variable number of measurements for each test. A simulation run produces outputs $\hat{\text{kf}}(\varphi)$ with a much higher resolution for φ. From these values we keep only the points with matching measurements $\text{kf}(\varphi)$ and sum up the squared errors. The metric for model accuracy is the sum over all tests of the mean of squared errors (SMSE). This puts equal weight on each test even when the number of measurements differs over the tests.

$$\text{MSE}(\hat{\text{kf}}, \text{kf}_t, \varphi_t) = \frac{1}{n_t} \sum_{i=1}^{n_t} \left(\hat{\text{kf}}(\varphi_{t,i}) - \text{kf}_t(\varphi_{t,i}) \right)^2 \tag{5}$$

$$\text{SMSE} = \sum_{t \in \text{tests}} \text{MSE}(\hat{\text{kf}}, \text{kf}_t, \varphi_t) \tag{6}$$

$\hat{\text{kf}}(\varphi)$ are the filtered points from the simulation and φ_t and kf_t the measurements from test t. All vectors have n_t elements. We do not normalize the MSE values for the tests because the target values are all on the same scale (see Fig. 1) and we aim to reduce absolute not relative errors of predictions.

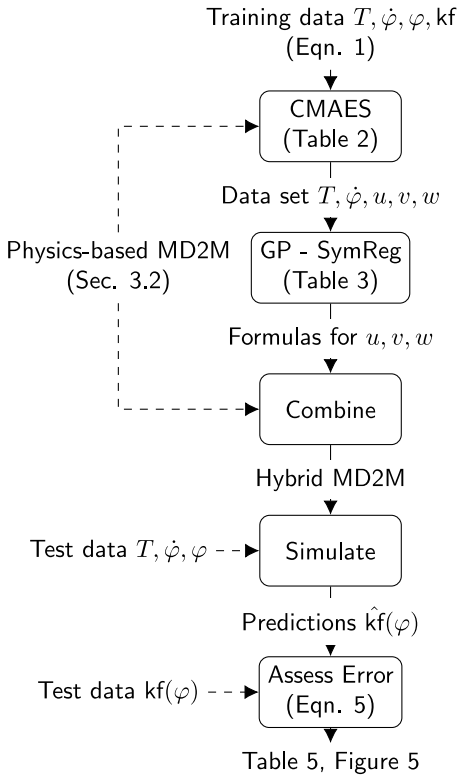

Fig. 2. Workflow of the explicit method for extending the constitutive model (MD2M). CMAES is used to optimize parameters u, v and w for each test set from Table 1 and GP is used to find the expressions for optimized u, v and w depending on T and $\dot{\varphi}$.

4.2. Explicit method: Parameter optimization and symbolic regression

Fig. 2 shows the workflow of the explicit method. A single simulation run produces a stress–strain curve $\hat{\text{kf}}(\varphi)$, which is returned as a table.

First, we use covariance matrix adaptation evolution strategy (CMAES) (Hansen et al., 2003) to optimize the calibration parameters for each test in the training set. The parameters may depend on each other, and we cannot assume that the optimization problem is convex. This makes parameter fitting difficult (Domkin, 2005). Thus, a derivative-free global optimization method such as CMAES or differential evolution can be an appropriate choice. Still, the optimizer may

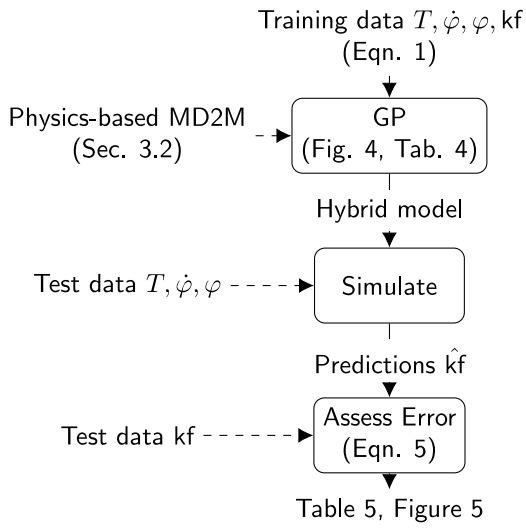

Fig. 3. Workflow of the implicit method for extending the constitutive model (MD2M). The extensions are encoded as multi-tree GP individuals and fitness evaluation uses simulations of the hybridized model.

converge to different solutions. Thus, multiple repetitions for the same data set are required. Our goal is to find a function mapping values of the known (processing) parameters to values of the calibration parameters. Therefore, it is important that the global optimizer converges reliably to the same or similar solutions. Otherwise, we will fail to find the solution in the subsequent step. At the end of the optimization, the found solutions are collected into a data set with best values for (u, v, w) for each test.

Next, we use the data set of the optimized (u, v, w) values for SR with GP to produce three formulas for the calculation of u, v, and w from the known parameters $(T, \dot{\varphi})$. At this stage, we recommend to use cross-validation to tune GP parameters (see Table 1). It is essential to find a GP parameterization that reliably produces a good solution, because we must select only a single formula for each of u, v, and w. After grid-search for good GP parameters, we execute a GP run with the whole training set for each of the three calibration parameters and combine the three formulas with the considered constitutive model (MD2M) to produce the *hybrid* model. The hybrid MD2M is then used to produce simulation results $\hat{\text{kf}}(\varphi)$ for the tests in the testing set.

4.3. Implicit method: Evolutionary extension

Fig. 3 shows the workflow for the implicit method. In contrast to the explicit method we do not fit the model to training data by parameter optimization, but instead use tree-based GP to directly evolve the three expression trees representing the formulas for the calculation of (u, v, w). It is important to note that the output of the MD2M model depends on all three calibration parameters. The quality of the extended MD2M model can be quantified via the SMSE between simulated and the measured stress–strain curves. However, we cannot attribute changes in the SMSE (Eq. (6)) to each of the formulas for (u, v, w). Therefore, we use a multi-tree GP for the implicit method.

Each GP individual consists of three separate trees for the three formulas. For fitness evaluation the three formulas are used to calculate the (u, v, w) values for each test. The SMSE between simulation and measurement over all tests is used to determine fitness, whereby individuals with smaller error are assigned a higher fitness value. Therefore, individuals with three formulas that work well in combination have a higher fitness and are more likely to be selected.

To highlight how recombination and mutation operations act independently on the components of the model, we give a pseudo-code for GP algorithm in Fig. 4. Crossover between two parent individuals acts on the three components independently. To produce a child we need to compose three trees from the parents. For each of the three components, we first choose randomly whether a crossover operation should be performed using the crossover probability parameter. The choice is made independently for each component. Without crossover we simply select one of the two trees for this component from the parents randomly. Otherwise a new tree is created using a sub-tree crossover. This crossover scheme does not allow exchange of genetic material between separate components. We recommend a crossover rate smaller than 100% to allow combination of already well working formulas. Mutation acts on the three components independently using the mutation probability parameter.

The individual with highest fitness for the tests in the training set is selected as the solution. The solution is then used to produce simulation results $\hat{\text{kf}}(\varphi)$ for the tests in the test set.

Table 2
CMAES parameters.

Parameter	Value
Search space	$u \in [-10..0] \times v \in [-15..15] \times w \in [-15, 0]$
Generations	500
Pop. size	100
Initialization	Uniform
Fitness	SMSE (Eq. (6)) for a simulated stress–strain curve
Recombination	Log-weighted

4.4. Algorithm configuration

4.4.1. Explicit approach

For the explicit approach, we execute 30 independent CMAES runs with the parameter settings given in Table 2. We choose the best values for each of the three parameters (u, v, w), which are then back-transformed to produce values for (A, B, C). The back-transformed variables are used as target for SR.[1] The resulting data set has 35 rows, two input variables, and three target variables. We use two subsets of the data for training and testing of models, and assign data from one test completely either to the training set or to the testing set using a systematic partitioning scheme from Schützeneder (2020) (see Table 1).

The maximum length of symbolic expression trees is selected using 30 independent repetitions of cross-validation (CV) with four folds on 24 training samples using the partitions shown in Table 1. The setting with the smallest median cross-validated root mean squared error (CV-RMSE) is used to train models on the full training set. The maximum limit for the number of tree nodes is chosen from the set $\{5, 7, 10, 15, 20, 25, 30, 35, 40\}$. The settings with best CV-RMSE are 25 nodes for A, 35 nodes for B, and 20 nodes for C. These are relatively tight limits for GP, but the grid-search showed that GP started to overfit with larger models. This can be explained by the small number of data points for training. Models with 20 to 35 nodes (before simplification) are relatively easy to interpret.[2]

For each of the three targets, we execute 30 GP runs and return the individual with highest fitness as the solution. The outputs of all GP models are clamped using target-specific limits as shown in Tables 3 and 4. This ensures that the calibration parameter values produced by GP models are physically plausible. For instance, negative parameter values are physically impossible and the maximum values depend on the material. We generate 30 hybrid MD2M by combining the three models from the ith SR run for each target and calculate the SMSE for training and testing sets.

[1] We also ran SR experiments to instead predict the transformed parameters u, v, w but found that the results were significantly worse.

[2] In this context it is important to point out that the variable nodes in the leaves of trees implicitly contain a scaling coefficient $(c_i * x_i)$. This is counted as only one node in this work while it would be counted as three nodes in most other GP systems.

```
Inputs: Model: MD2M(x, theta)  // x: vector of inputs
                               // theta: parameter vector of length dim
        Matrix of inputs: X = (x_t) // t = 1..tests
        List of targets:  y = (y_t) // t = 1..tests
                               // each y_t is a time series of measurements
Output: Hybridized model g(x) = MD2M(x, theta: f(x))

P = Init(popSize)              // each individual is a vector
                               // of expressions encoded as trees
for g = 1 .. maxGenerations
    fitness = [ Evaluate(MD2M, y, X, individual))
                for each individual in P]
    P = order P by descending fitness
    P_next[1] = P[1];      // copy best individual
    for k = 2 .. popSize
        p1 = Select(P)     // select two parents with tournament selection
        p2 = Select(P)
        child = new empty vector of dim expressions
        for treeIndex = 1 .. dim
            if rand() < crossoverProbability
                child[treeIndex] = Crossover(p1[treeIndex], p2[treeIndex])
            else
                child[treeIndex] = rand() < 0.5 ? p1[treeIndex] : p2[treeIndex]

            if rand() < mutationProbability
                Mutate(child[treeIndex])
        end                    // for all components
        P_next[k] = child
    end                        // for population
    P = P_next
end
return P[1];                   // return individual with best fitness
```

Fig. 4. Pseudo-code for the multi-tree GP algorithm to evolve the extensions for the MD2M model. Individuals contain multiple expression trees, one for each element of the parameter vector θ. The crossover and mutation act independently on the components of individuals.

Table 3
GP parameters for SR as part of the explicit approach.

Parameter	Value
Pop. size	300
Generations	250
Max. length	$A: 25, B: 35, C: 20$
Max. depth	8
Initialization	PTC2 (Luke, 2000)
Selection	Tournament (size 3)
Recombination	Sub-tree crossover (90% internal nodes)
Mutation	Probability 15%
	Select randomly:
	For a random parameter: $x \leftarrow x + N(0,1)$
	For all parameters: $x \leftarrow x + N(0,1)$
	Change the symbol of a random node
	Change a random variable node
Clamp predictions	$A: [0..150], B: [0..150], C: [0..1]$
Fitness	Sum of squared errors (for A, B, C predictions)
Replacement	Generational with one elite.
Function set	$\{+, -, \times, \div, \exp(x), \log(x)\}$
Terminals	50% variables, 50% numeric parameters
	Variables: {temp, $\dot{\varphi}$, $log_{10}(\dot{\varphi})$}
	Numeric parameters $\sim U(-20, 20)$

Table 4
GP parameter settings for the implicit approach.

Parameter	Value
Pop. size	5000
Generations	250
Max. length	25
Max. depth	10
Initialization	PTC2
Selection	Tournament (size 7)
Recombination	Probability 30% for each tree of an individual
	Sub-tree crossover (90% internal nodes)
Mutation	Probability 15% for each tree of an individual
	Select randomly:
	For a random parameter: $x \leftarrow x + N(0,1)$
	For all parameters: $x \leftarrow x + N(0,1)$
	Change the symbol of a random node
	Change a random variable node
Clamp predictions	$u: [-15..0], v: [-15..15], w: [-15..0]$
Fitness	SMSE (Eq. (6))
Replacement	Generational with one elite.
Function set	$\{+, -, \times, \div, \exp(x), \log(x)\}$
Terminals	50% variables, 50% numeric parameters
	Variables: {temp, $\dot{\varphi}, log_{10}(\dot{\varphi})$}
	Numeric parameters $\sim U(-20, 20)$

4.4.2. Implicit approach

We use tree-based GP with generational replacement with elitism. As described above, each individual contains three trees (u, v, w). Trees are limited to 25 nodes and a maximum depth of 10 for each component. Our GP system initializes trees randomly using PTC2, whereby for each leaf it first randomly determines the leaf type: variable or parameter. All variable nodes always include a scaling factor sampled randomly from $N(0,1)$.[3] The parameters are sampled randomly from $U(-20, 20)$. The GP parameter settings used for the implicit approach are given in Table 4.

5. Results

Fig. 5 visualizes the simulation outputs when A, B, C are calculated via interpolation, the implicit, and the explicit model. Only the 11 hot compression tests from the testing set are shown. For the implicit and the explicit model, the model with best training SMSE from the 30 runs is used. A systematic deviation for small kf values is apparent for all methods, which is due to a systematic bias of the MD2M. It is not

[3] This is the default of the GP system used. This increases the number of parameters in the SR models and can be helpful or detrimental for fitting. We have not analysed the effects of removing scaling parameters for variables.

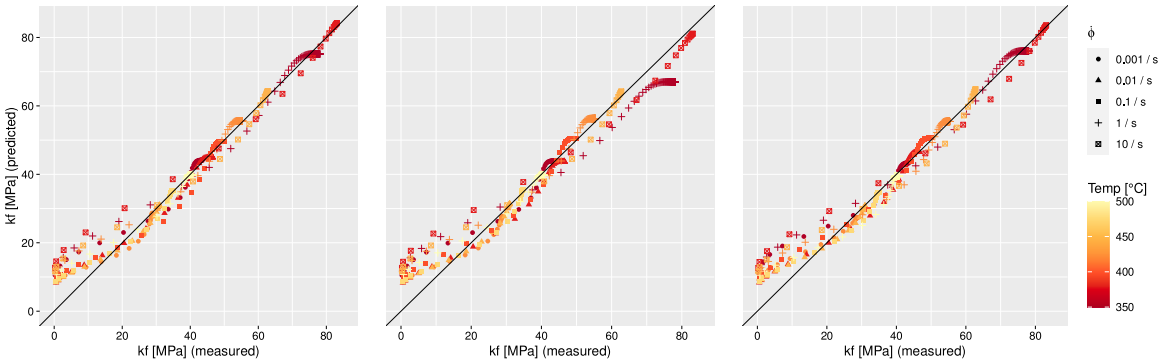

Fig. 5. Scatter plots for predictions over measurements for interpolation (left), the explicit (middle), and the implicit (right) approach. The implicit approach produces a better fit for high stress values.

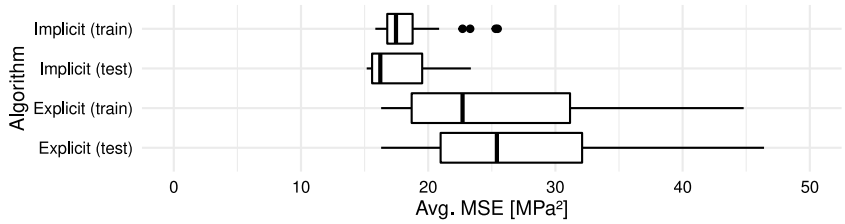

Fig. 6. Box-plots for the average MSE over 30 repetitions for training and testing sets. Not shown: twelve outliers for the explicit method (training: 2, test: 10).

possible to calibrate the MD2M to improve the simulation results for small kf values even with CMAES.

Fig. 6 illustrates box-plots for the average training and test MSE values achieved by both methods over 30 repetitions. The implicit method has smaller MSE values on average on the training and testing sets. Additionally, the variance for the implicit method is much smaller than for the explicit method. The non-parametric Brown-Mood median test (for samples with different variances) indicates to reject the null-hypothesis of equal medians for the training set (p-value = 0.0001792) and for the testing set (p-value = $1.183 \cdot 10^{-07}$). For the optimized parameter values (CMAES) the average test MSE is 13.58 MPa^2 (RMSE=3.69 MPa), for the interpolated values it is 14.35 MPa^2 (RMSE=3.79 MPa).

The median MSE values for each test for the hybrid models produced by 30 independent runs of the explicit and implicit method are collected in Table 5. The implicit method has smaller MSE values than the explicit method for the training and the testing sets. For comparison, the median MSE values achieved with optimized parameters (CMAES) and with interpolation are given as a reference. The CMAES runs reliably converged to the same solutions and give an indication of the best possible MSE values that can be achieved when using the MD2M.

Instead of learning a model we can also interpolate A,B,C from the optimized parameters for the compression tests in the training set. We found the best results with linear interpolation along the temperature dimension. For this we calculate the parameters for each test from the values found for lower and higher temperatures. The results of linear interpolation are shown in the interpolation column in Table 5.

Summarizing the results, we found a better model (with on average a smaller SMSE) using the implicit method, than with the explicit method. With increasing strain rate, the fit achieved with the explicit method gets better. For the highest strain rate, the explicit method produces a better fit for 5 of 7 tests. Linear interpolation works well and produces even better results than the implicit method. However, interpolation does not give us a simple formula to calculate A,B,C.

Eqs. (7)–(9) show the resulting set of expressions for u, v and w produced using the implicit method and having the best performance on the test set after algebraic simplification. To determine A, B and C, the back-transformation as shown in Eqs. (10)–(12) is required, whereby $\text{clamp}(x, l, u)$ returns $\min(u, \max(l, x))$. The models identified by GP are non-linear in the input variables $(T, \dot{\varphi})$ and very short. Most of the parameters are linear which facilitates interpretation of the formula. The formulas identified by the explicit approach found in Kabliman et al. (2019) had a similar complexity, because the tree size restrictions for both algorithms were similar.

$$u(T, \dot{\varphi}) = -\log(0.11\,T + 3.734\,\dot{\varphi} - 17.34) - 0.069\,\log(\dot{\varphi}) \tag{7}$$

$$v(T) = -4.35\,\log(T) + 4.938 \tag{8}$$

$$w(T, \dot{\varphi}) = -13.708\,\log(\dot{\varphi})\,T^{-1} - 10205\,\log(\dot{\varphi})\,T^{-2} + 13675\,T^{-2} + 0.777\,\log(\dot{\varphi}) - 0.8657 \tag{9}$$

$$A(T, \dot{\varphi}) = \exp\left(-\text{clamp}(u(T, \dot{\varphi}), -15, 0)\right) \tag{10}$$

$$B(T, \dot{\varphi}) = \exp\left(\text{clamp}(v(T), -15, 15)\right) A(T, \dot{\varphi})^{-1} \tag{11}$$

$$C(T, \dot{\varphi}) = \exp\left(\text{clamp}(w(T, \dot{\varphi}), -15, 0)\right) A(T, \dot{\varphi})^{-1} \tag{12}$$

Finally, the predicted (A, B, C) values for the tested range of temperatures and strain rates are shown in Fig. 7 to visualize the correlation with the processing parameters.

6. Discussion

The goal of the present study was to extend the physics-based constitutive model using GP and SR for predicting of unknown relations of the calibration parameters to the processing conditions. We showed that it is possible to evolve short formulas using two (explicit and implicit) methods. However, there are several caveats that we want to discuss below and plan to address in future work.

6.1. Comparison to classical interpolation methods

Instead of finding a model for the calculation of the calibration parameters it is possible to use linear or cubic spline interpolation. In fact, linear interpolation along the temperature dimension produced better fitting results than the model. However, the equations produced by GP are more attractive compared to interpolation, since they have only a few parameters and are easily interpretable. For example the

Table 5
Median MSE values over 30 runs for the explicit and implicit method for the training and testing sets. The MSE values for CMAES and interpolation are shown as a reference value as they are indicative of the best achievable value for the MD2M.

	Temp [° C]	$\dot{\varphi}$ [s^{-1}]	CMAES MSE [MPa^2]	Interp. MSE [MPa^2]	Explicit MSE [MPa^2]	Implicit MSE [MPa^2]
Training	375	0.001	9.44	9.71	22.23	**11.48**
	400	0.001	12.70	12.95	21.11	**14.63**
	450	0.001	14.99	15.81	28.55	**18.31**
	475	0.001	14.04	14.46	33.87	**17.19**
	500	0.001	14.94	16.69	26.34	**21.54**
	350	0.01	7.76	11.79	**10.44**	10.58
	400	0.01	6.23	6.20	10.70	**8.33**
	425	0.01	8.38	8.66	12.02	**9.64**
	475	0.01	10.71	10.96	13.92	**12.22**
	500	0.01	13.37	13.87	23.63	**16.20**
	350	0.1	7.88	8.13	10.41	**9.79**
	375	0.1	9.61	17.50	14.73	**11.56**
	425	0.1	6.31	8.02	8.65	**8.03**
	450	0.1	3.85	3.95	5.66	**4.79**
	500	0.1	7.88	8.74	**9.63**	11.09
	375	1	7.82	8.19	11.27	**10.04**
	400	1	6.54	7.01	**7.88**	7.94
	450	1	4.89	4.95	**5.25**	7.29
	475	1	4.45	4.57	**4.76**	6.57
	350	10	43.58	45.70	50.92	**50.81**
	400	10	38.07	38.45	**38.54**	38.57
	425	10	38.86	39.11	**39.14**	40.08
	475	10	31.05	31.24	**31.41**	33.53
	500	10	18.59	19.15	**20.49**	22.60
	Avg		14.25	15.24	19.23	**16.78**
Test	350	0.001	17.07	18.39	41.24	**21.59**
	425	0.001	14.43	15.24	21.81	**16.76**
	375	0.01	5.92	6.73	9.38	**9.21**
	450	0.01	8.99	9.15	11.44	**10.50**
	400	0.1	5.07	5.94	**5.83**	6.43
	450	0.1	2.38	2.51	**2.86**	3.53
	350	1	9.94	12.42	81.95	**15.11**
	425	1	6.12	7.00	8.52	**7.66**
	500	1	3.78	4.00	**4.68**	6.22
	375	10	42.98	43.50	45.62	**44.47**
	450	10	32.77	32.93	**32.95**	34.65
	Avg		13.59	14.35	24.21	**16.01**

interpolation map for parameter A has 24 coefficients while the best model (Eq. (7)) has only four coefficients. The expressions for the three calibration parameters can be combined easily with the expressions of the M2DM.

6.2. Runtime and convergence

For the explicit method, the runtime for one CMAES run was approximately 6 hours (10 min for each of the 35 tests) and the runtime for SR was just a few minutes, as there were only 24 training points. Linear interpolation is computationally very cheap therefore much faster than SR, but it also requires the same optimization runs with CMAES. The implicit method is computationally much more expensive because each GP fitness evaluation requires a separate simulation run. The runs took approximately 45 days on average single-threaded. Concerning runtime the explicit method is therefore much better than the implicit approach. The implicit method however provides more options for the evaluation of the fit and led to significantly more accurate models. The runtime can be reduced through parallelization of the simulation runs. We cannot definitively state whether more runtime for the explicit approach would improve the results, because longer runs were not tested. However, we analysed the convergence of CMAES runs and observed that it reliably converged to the same points with the parameters that we have chosen. Therefore, we believe it is unlikely that much larger population sizes or more generations would have improved the results for CMAES. The runtime of CMAES is the main contributor to the total runtime for the explicit approach so increasing the number of generations for SR would increase total runtime only by a small percentage.

6.3. Combination of models

The implicit approach directly evolves a set of models that produces a good prediction in combination, while the explicit approach evolves each model independently. To create 30 hybrid models for evaluation, we used the three models from the first run for the first combined model, the three models from the second run for the second combined model and so on. The finally selected combined model is the combination with the best performance (SNMSE) on the training set. However, from the 3*30 individual models, we could also create all 90 combinations with minor overhead. This could potentially increase the chance to build and select a good combination of models, but might also increase the chance for overfitting.

6.4. Choice of fitness function for SR

One potential reason for the worse performance of the explicit method is that the chosen fitness function for SR could be inappropriate. The implicit method uses the fit for the simulated stress–strain curve as a fitness criterion and therefore directly optimizes the error that we measure for the test set. With the explicit method, we build formulas to minimize the error to the (u, v, w) values produced by CMAES. In our experiments, we used the sum of squared errors as the fitness function for this SR step based on the assumption that a formula with smaller squared error for (u, v, w) also leads to a better fit for the simulated stress–strain curve $\hat{kf}(\varphi)$. Based on our results, we believe that this assumption does not hold. For example, it may be necessary to predict a value more accurately as it approaches zero, while for

log10(ϕ)	350 °C	375 °C	400 °C	425 °C	450 °C	475 °C	500 °C
-3	13.16	14.87	16.58	18.29	20.00	21.71	23.42
-2.5	14.24	16.09	17.95	19.80	21.65	23.50	25.35
-2	15.43	17.44	19.44	21.44	23.45	25.45	27.45
-1.5	16.76	18.93	21.10	23.27	25.44	27.60	29.77
-1	18.36	20.71	23.05	25.40	27.75	30.09	32.44
-0.5	20.62	23.16	25.70	28.24	30.78	33.32	35.86
0	24.89	27.64	30.38	33.13	35.88	38.63	41.37
0.5	35.74	38.71	41.69	44.66	47.63	50.61	53.58
1	68.82	72.03	75.25	78.47	81.69	84.90	88.12

log10(ϕ)	350 °C	375 °C	400 °C	425 °C	450 °C	475 °C	500 °C
-3	47.94	45.45	43.48	41.87	40.55	39.43	38.47
-2.5	44.28	41.99	40.17	38.69	37.46	36.42	35.54
-2	40.87	38.76	37.08	35.72	34.59	33.63	32.82
-1.5	37.62	35.69	34.16	32.91	31.88	31.01	30.26
-1	34.35	32.63	31.27	30.15	29.23	28.45	27.78
-0.5	30.59	29.18	28.05	27.12	26.35	25.69	25.13
0	25.34	24.45	23.72	23.12	22.60	22.16	21.78
0.5	17.65	17.46	17.29	17.15	17.02	16.91	16.82
1	9.17	9.38	9.58	9.76	9.93	10.08	10.22

log10(ϕ)	350 °C	375 °C	400 °C	425 °C	450 °C	475 °C	500 °C
-3	0.00039	0.00031	0.00025	0.00021	0.00018	0.00016	0.00014
-2.5	0.00076	0.00061	0.00051	0.00043	0.00038	0.00033	0.00030
-2	0.00149	0.00122	0.00103	0.00089	0.00078	0.00069	0.00062
-1.5	0.00292	0.00243	0.00208	0.00181	0.00160	0.00143	0.00129
-1	0.00567	0.00480	0.00416	0.00366	0.00326	0.00294	0.00268
-0.5	0.01073	0.00927	0.00816	0.00727	0.00656	0.00597	0.00548
0	0.01890	0.01678	0.01508	0.01370	0.01255	0.01157	0.01074
0.5	0.02798	0.02583	0.02399	0.02239	0.02099	0.01976	0.01866
1	0.01453	0.01388	0.01329	0.01274	0.01224	0.01178	0.01135

Fig. 7. Predicted values for the calibration parameters A, B, and C.

larger target values we may also allow larger errors. In this case, it could be better to minimize the relative error. This can be considered an advantage of the implicit approach, because it frees us from the burden to select an appropriate fitness function for the intermediate SR step. It would be worthwhile to study different fitness functions in future work to gain a better understanding about the effect of this choice on the overall performance of the approach.

6.5. Implementation effort

The explicit method is easier to implement than the implicit method, because the separate steps in the workflow only require to call readily available and well-tuned software components (CMAES, SR tools). For the implicit method, it is necessary to use a multi-tree GP system or to adapt a GP system accordingly. Additionally, for the fitness evaluation, the simulation model has to be implemented or connected to the GP system. Most GP systems, however, allow this kind of extension.

7. Related studies

The well-maintained GP bibliography (Langdon, 2020) contains many references to prior work, in which GP has been used for constitutive modelling in particular for predicting stress for various materials. GP for multi-scale material modelling is extensively discussed in Sastry (2007). Two early works which describe the application of GP for the identification of constitutive models are (Schoenauer et al., 1996; Sebag et al., 1997). The approach described in Schoenauer et al. (1996) is especially notable as it uses specific operators to ensure that the resulting models have physical interpretation as elastic, plastic and viscous components.

Since then GP has been used extensively for constitutive modelling such as for modelling flow stress for various metallic materials (Brezocnik et al., 2000, 2001, 2002) including aluminium alloys (Sastry et al., 2004), for modelling stress distribution in cold-formed copper alloys (Brezocnik and Gusel, 2004) and X6Cr13 steel (Brezocnik et al., 2005), for predicting impact toughness (Gusel and Brezocnik, 2006), for the identification of visco-elastic models for rocks (Feng et al., 2006), and stress–strain behaviour of sands under cyclic loading (Shahnazari et al., 2010, 2015), for predicting material properties under hot deformation, in particular for carbon silicon steel (Kovacic et al., 2005), and a nickel-based alloy (Lin et al., 2017), for predicting the presence of cracks in hot rolled steel (Kovacic et al., 2011), for modelling tensile strength, electrical conductivity of cold-drawn copper alloys (Gusel and Brezocnik, 2011), for prediction of shear strength of reinforced concrete beams (Gandomi et al., 2013), for formulating the stress–strain relationship of materials in Gandomi and Yun (2015), and for predicting fatigue for 2024 T3 aluminium alloys under load (Mohanty et al., 2015). GP has further been used for predicting non-linear stress–strain curves (e.g. for aluminium and stainless steel alloys) (Kabliman et al., 2019; Cevik and Guzelbey, 2007; Schützeneder, 2020), predicting elastic distortional buckling stress of cold-formed steel C-sections (Pala, 2008), predicting residual stress in plasma-nitrided tool steel (Podgornik et al., 2011), and modelling mechanical strength of austenitic stainless steel alloys (SS304) as a function of temperature, strain and strain rate (Vijayaraghavan et al., 2017).

An evolutionary method for polynomial regression and the combination with finite element analysis has been used for constitutive modelling and applied for instance to predict the behaviour of soils under drained and undrained load conditions in Rezania (2008) and Ahangarasr (2012). Furthermore, evolutionary algorithms have been used for the optimization of calibration parameters of constitutive models in Lin et al. (2015) and optimizing alloy composition using Gaussian process surrogate models and constitutive models for simulation (Tancret and thermodynamics, 2013). In Mulyadi et al. (2006), different methods for parameter optimization of a constitutive model for hot deformation of a titanium alloy have been tested and compared with the predictions made by artificial neural networks.

All of the papers discussed above describe a from of regression modelling. In those papers, GP is used for supervised learning to establish a free-form constitutive model using SR (e.g. Sastry, 2007; Schoenauer et al., 1996) or alternatively the model structure is fixed and parameters are optimized using evolutionary algorithms. We are, however, mainly interested in combining or extending physics-based models with machine learning models and found only a few papers with a similar focus in the material science domain.

A hybrid modelling approach using a physics-based model and neuro-fuzzy evolution has been described and applied for modelling thermo-mechanical processing of aluminium alloys in Abbod et al. (2002). The same authors later sketch a similar GP-based approach in Abbod et al. (2006). This work is similar to the present study, but there are several important differences. Abbod et al. (2006) used a simpler physics-based model for flow stress and predicted only three relevant points in the stress–strain curves (steady-state flow stress, the relaxation stress, and the relaxation strain) instead of the full curve. The authors first fit neuro-fuzzy models to the data and only later used GP to find short equations that predict the output of those models. In contrast to our approach proposed in the present work, the resulting GP models are not directly linked to the physics-based constitutive model. Instead, the pre-calculated features and sub-expressions were derived from the physics-based model within GP to produce similar expressions.

Versino et al. (2017) have described different methods for physics-informed SR for modelling of the stress–strain curves. The methods include addition of artificial data points to improve extrapolation, constraints (e.g. to force models to be non-negative), seeding of a GP population with initial solutions based on the physics-based models, and user-defined features using building blocks derived from the physics-based models (e.g. non-linear transformations of input variables). Seeding GP with the physics-based model is similar to our approach but does not guarantee that the evolved model has the same structure as the physics-based model. In the conclusions Versino et al.

(2017) state: *[...] model development can [again] be expertly guided by choosing appropriate building blocks, avoiding functions that might introduce excessive numerical issues. At the same time [...] symbolic regression presents clear limits. When no experimental data or expert knowledge is available, the behaviour of obtained models is highly unpredictable, and unlikely to be rooted in solid physics. [...] Moreover, symbolic regression will probably return completely different models for different materials, limiting the re-usability of a result. Additionally, as EAs are stochastic in nature, there is no guarantee that two runs of the algorithm on the same dataset will provide exactly the same results, introducing reproducibility problems.* We try to partially alleviate these issues by extending the physics-based model with GP. This ensures that at least the core of the model remains unchanged.

Sedighiani et al. (2020) have described a computationally efficient method for the identification of constitutive parameters from stress–strain curves and demonstrated the method for a number of models including a dislocation-density-based crystal plasticity model for steel. They use a genetic algorithm to fit the parameters and response surface models as surrogate models to save simulations and reduce runtime. In contrast to the method proposed in this paper, the method does not produce formulas which relate the constitutive parameters to the load parameters. Instead it produces the parameter values directly. The method could be used to speed-up the first step (parameter identification) within the explicit approach described below.

8. Conclusions

We used machine learning for extending a physics-based constitutive model, which describes the materials behaviour in terms of internal state variables. Using GP and SR, we could derive three short expressions for the unknown dependencies of the model calibration parameters to known impact variables such as the processing conditions. As a result, hybrid physics-based and data-driven constitutive models were formulated. We compared two, explicit and implicit, methods for derivation of required formulas.

The critical step in the explicit method is the combination of the SR models for the calibration parameters with the physics-based model. Even though the individual SR models were able to predict the calibration parameters accurately for training and testing partitions, they did not perform well in combination. The implicit method instead directly optimizes the fit of the hybrid material model, and is able to evolve a combination of short formulas. Thus, the results have shown that the hybrid material models produced by the implicit method have significantly better predictive accuracy on average than the models produced by the explicit method. Additionally, the implicit method has a smaller variance. However, the results achieved with traditional linear interpolation are better than the results obtained by the two model-based approaches.

We recommend the implicit approach over the explicit approach for finding formulas. It allows the end-to-end fitting of the simulation model and does not have multiple intermediate steps of model selection, where an appropriate fitness function has to be chosen. However, the implicit approach comes at a significantly greater computational expense, because more evaluations of the simulation model are required. In particular, the computational demand is much higher compared to traditional interpolation methods. To improve the runtime, future work could try to either improve the explicit approach or try to incorporate surrogate models instead of the simulation model. However, even with those improvements it will not be possible to reach similar runtimes as traditional interpolation methods.

The use of the formulas instead of the calibration parameters might help to overcome the required regular fitting and reduce the amount of necessary measurements. Moreover, the main advantage of the proposed approach is that parameters A, B, and C directly depend on the processing conditions which gives a higher resolution of the calibration parameter values. The accuracy of the calculated local flow stress is therefore higher compared to fixed calibration values. The proposed approach is of a general purpose and can be applied in other areas, when the relation of model parameters to impact factors is not well understood.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors gratefully acknowledge support by the Christian Doppler Research Association, Austria and the Federal Ministry of Digital and Economic Affairs within the Josef Ressel Center for Symbolic Regression. This work was supported by the Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology (in German, BMK) in a framework of the project LiMFo (Light Metal Forming).

References

Abbod, M., Linkens, D., Zhu, Q., Mahfouf, M., 2002. Physically based and neuro-fuzzy hybrid modelling of thermomechanical processing of aluminium alloys. Mater. Sci. Eng. A 333, 397–408. http://dx.doi.org/10.1016/S0921-5093(01)01873-1.

Abbod, M.F., Mahfouf, M., Linkens, D.A., Sellars, C.M., 2006. Evolutionary computing for metals properties modelling. In: THERMEC 2006, Volume 539 of Materials Science Forum. Trans Tech Publications, Vancouver, pp. 2449–2454, 10.4028/www.scientific.net/MSF.539-543.2449.

Ahangarasr, A., 2012. Application of an Evolutionary Data Mining Technique for Constitutive Modelling of Geomaterials. (Ph.D. thesis). University of Exeter, UK.

Asadzadeh, M.Z., Gänser, H.-P., Mücke, M., 2021. Symbolic regression based hybrid semiparametric modelling of processes: An example case of a bending process. Appl. Eng. Sci. 6, 100049. http://dx.doi.org/10.1016/j.apples.2021.100049, https://www.sciencedirect.com/science/article/pii/S2666496821000157.

Baker, N., Alexander, F., Bremer, T., Hagberg, A., Kevrekidis, Y., Najm, H., Parashar, M., Patra, A., Sethian, J., Wild, S., et al., 2019. Workshop Report on Basic Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence. Technical Report, USDOE Office of Science (SC), Washington, DC (United States).

Brezocnik, M., Balic, J., Gusel, L., 2000. Artificial intelligence approach to determination of flow curve. J. Technol. Plast. 25, 1–7.

Brezocnik, M., Balic, J., Kampus, Z., 2001. Modeling of forming efficiency using genetic programming. J. Mater Process. Technol. 109, 20–29. http://dx.doi.org/10.1016/S0924-0136(00)00783-4.

Brezocnik, M., Balic, J., Kuzman, K., 2002. Genetic programming approach to determining of metal materials properties. J. Intell. Manuf. 13, 5–17. http://dx.doi.org/10.1023/A:1013693828052.

Brezocnik, M., Gusel, L., 2004. Predicting stress distribution in cold-formed material with genetic programming. Int. J. Adv. Manuf. Technol. 23, 467–474. http://dx.doi.org/10.1007/s00170-003-1649-3.

Brezocnik, M., Kovacic, M., Gusel, L., 2005. Comparison between genetic algorithm and genetic programming approach for modeling the stress distribution. Mater. Manuf. Process. 20, 497–508. http://dx.doi.org/10.1081/AMP-200053541.

Cevik, A., Guzelbey, I.H., 2007. A soft computing based approach for the prediction of ultimate strength of metal plates in compression. Eng. Struct. 29, 383–394. http://dx.doi.org/10.1016/j.engstruct.2006.05.005.

Domkin, K., 2005. Constitutive Models Based on Dislocation Density : Formulation and Implementation Into Finite Element Codes. (Ph.D. thesis). Luleå Tekniska Universitet.

Feng, X.-T., Chen, B.-R., Yang, C., Zhou, H., Ding, X., 2006. Identification of viscoelastic models for rocks using genetic programming coupled with the modified particle swarm optimization algorithm. Int. J. Rock Mech. Min. Sci. 43, 789–801. http://dx.doi.org/10.1016/j.ijrmms.2005.12.010.

Gandomi, A.H., Yun, G.J., 2015. Coupled SelfSim and genetic programming for nonlinear material constitutive modelling. Inverse Probl. Sci. Eng. 23, 1101–1119. http://dx.doi.org/10.1080/17415977.2014.968149.

Gandomi, A.H., Yun, G.J., Alavi, A.H., 2013. An evolutionary approach for modeling of shear strength of RC deep beams. Mater. Struct. 46, 2109–2119. http://dx.doi.org/10.1617/s11527-013-0039-z.

Gusel, L., Brezocnik, M., 2006. Modeling of impact toughness of cold formed material by genetic programming. Comput. Mater. Sci. 37, 476–482. http://dx.doi.org/10.1016/j.commatsci.2005.11.007.

Gusel, L., Brezocnik, M., 2011. Application of genetic programming for modelling of material characteristics. Expert Syst. Appl. 38, 15014–15019. http://dx.doi.org/10.1016/j.eswa.2011.05.045.

Hansen, N., Müller, S.D., Koumoutsakos, P., 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary Computation 11, 1–18.

Kabliman, E., Kolody, A.H., Kommenda, M., Kronberger, G., 2019. Prediction of stress-strain curves for aluminium alloys using symbolic regression. AIP Conf. Proc. 2113, 180009. http://dx.doi.org/10.1063/1.5112747.

Kabliman, E., Kolody, A.H., Kronsteiner, J., Kommenda, M., Kronberger, G., 2021. Application of symbolic regression for constitutive modeling of plastic deformation. Appl. Eng. Sci. 6, 100052. http://dx.doi.org/10.1016/j.apples.2021.100052.

Kovacic, M., Brezocnik, M., Turk, R., 2005. Modeling of hot yield stress curves for carbon silicon steel by genetic programming. Mater. Manuf. Process. 20, 543–551. http://dx.doi.org/10.1081/AMP-200053572.

Kovacic, M., Mackosek, K., Mihevc, A., Marolt, T., 2011. Modeling of cracks presence in steel after hot rolling. In: Brdarevic, S. (Ed.), 7th Research/Expert Conference with International Participations, QUALITY 2011, Faculty of Mechanical Engineering. University of Zenica, Neum, Bosnia and Herzegovina, pp. 569–572.

Koza, J.R., 1992. Genetic Programming: On the Programming of Computers By Means of Natural Selection, Vol. 1. MIT Press.

Koza, J.R., Keane, M.A., Rice, J.P., 1993. Performance improvement of machine learning via automatic discovery of facilitating functions as applied to a problem of symbolic system identification. In: 1993 IEEE International Conference on Neural Networks, Vol. I. IEEE, San Francisco, USA, pp. 191–198.

Langdon, W.B., 2020. Genetic programming and evolvable machines at 20. Genet. Program. Evol. Mach. 21, 205–217. http://dx.doi.org/10.1007/s10710-019-09344-6.

Lin, J., Cao, J., Balint, D., 2015. Development and determination of unified viscoplastic constitutive equations for predicting microstructure evolution in hot forming processes. Int. J. Mechatron. Manuf. Syst. 4, 387–401.

Lin, Y.C., Nong, F.-Q., Chen, X.-M., Chen, D.-D., Chen, M.-S., 2017. Microstructural evolution and constitutive models to predict hot deformation behaviors of a nickel-based superalloy. Vacuum 137, 104–114. http://dx.doi.org/10.1016/j.vacuum.2016.12.022.

Luke, S., 2000. Two fast tree-creation algorithms for genetic programming. IEEE Trans. Evol. Comput. 4, 274–283. http://dx.doi.org/10.1109/4235.873237.

Mohanty, J.R., Mahanta, T.K., Mohanty, A., Thatoi, D.N., 2015. Prediction of constant amplitude fatigue crack growth life of 2024 T3 Al alloy with R-ratio effect by GP. Appl. Soft Comput. 26, 428–434. http://dx.doi.org/10.1016/j.asoc.2014.10.024.

Mulyadi, M., Rist, M.A., Edwards, L., Brooks, J.W., 2006. Parameter optimisation in constitutive equations for hot forging. J. Mater Process. Technol. 177, 311–314.

Pala, M., 2008. Genetic programming-based formulation for distortional buckling stress of cold-formed steel members. J. Construct. Steel Res. 64, 1495–1504. http://dx.doi.org/10.1016/j.jcsr.2008.01.018.

Podgornik, B., Leskovsek, V., Kovacic, M., Vizintin, J., 2011. Residual stress field analysis and prediction in nitrided tool steel. Mater. Manuf. Process. 26, 1097–1103. http://dx.doi.org/10.1080/10426914.2010.525573.

Rezania, M., 2008. Evolutionary Polynomial Regression Based Constitutive Modelling and Incorporation in Finite Element Analysis. (Ph.d. in Geotechnical Engineering). School of Engineering, Computing and Mathematics, University of Exeter, UK.

Sastry, K.N., 2007. Genetic Algorithms and Genetic Programming for Multiscale Modeling: Applications in Materials Science and Chemistry and Advances in Scalability. (Ph.D. thesis). University of Illinois at Urbana-Champaign, Urbana IL. USA.

Sastry, K., Johnson, D.D., Goldberg, D.E., Bellon, P., 2004. Genetic programming for multiscale modeling. Int. J. Multiscale Comput. Eng. 2, 239–256. http://dx.doi.org/10.1615/IntJMultCompEng.v2.i2.50.

Schoenauer, M., Sebag, M., Jouve, F., Lamy, B., Maitournam, H., 1996. Evolutionary identification of macro-mechanical models. In: Angeline, P.J., Kinnear, Jr., K.E. (Eds.), Advances in Genetic Programming, Vol. 2. MIT Press, Cambridge, MA, USA, pp. 467–488. http://dx.doi.org/10.7551/mitpress/1109.003.0030.

Schützeneder, G., 2020. Die Modellierung Von Spannungs-Dehnungs-Kurven für Die Aluminiumlegierung 6082 Mit Methoden Des Maschinellen Lernens (Master's thesis). University of Applied Sciences Upper Austria, School of Informatics, Communication, and Media, Hagenberg.

Sebag, M., Schoenauer, M., Maitournam, H., 1997. Parametric and non-parametric identification of macro-mechanical models. In: Quagliarella, D., Periaux, J., Poloni, C., Winter, G. (Eds.), Genetic Algorithms and Evolution Strategies in Engineering and Computer Sciences. John Wiley, pp. 327–340.

Sedighiani, K., Diehl, M., Traka, K., Roters, F., Sietsma, J., Raabe, D., 2020. An efficient and robust approach to determine material parameters of crystal plasticity constitutive laws from macro-scale stressâ€ strain curves. Int. J. Plast. 134, 102779. http://dx.doi.org/10.1016/j.ijplas.2020.102779, https://www.sciencedirect.com/science/article/pii/S0749641919308769.

Shahnazari, H., Dehnavi, Y., Alavi, A.H., 2010. Numerical modeling of stress-strain behavior of sand under cyclic loading. Eng. Geol. 116, 53–72. http://dx.doi.org/10.1016/j.enggeo.2010.07.007.

Shahnazari, H., Dehnavi, Y., Alavi, A.H., 2015. The next-generation constitutive correlations for simulation of cyclic stress-strain behavior of sand. J. Civil Eng. Manag. 21, 31–44. http://dx.doi.org/10.3846/13923730.2013.802726.

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P.-Y., Hjalmarsson, H., Juditsky, A., 1995. Nonlinear black-box modeling in system identification: a unified overview. Automatica 31, 1691–1724.

Tancret, F., thermodynamics, Computational., 2013. Gaussian Processes and genetic algorithms: combined tools to design new alloys. Modelling Simulation Mater. Sci. Eng. 21, 045013. http://dx.doi.org/10.1088/0965-0393/21/4/045013.

Versino, D., Tonda, A., Bronkhorst, C.A., 2017. Data driven modeling of plastic deformation. Comput. Methods Appl. Mech. Engrg. 318, 981–1004.

Vijayaraghavan, V., Garg, A., Tai, K., Gao, L., 2017. Thermo-mechanical modeling of metallic alloys for nuclear engineering applications. Measurement 97, 242–250. http://dx.doi.org/10.1016/j.measurement.2016.11.003.

Von. Stosch, M., Oliveira, R., Peres, J., d. Azevedo, S.F., 2014. Hybrid semi-parametric modeling in process systems engineering: Past, present and future. Comput. Chem. Eng. 60, 86–101.