Finding Transition Models using Dimensional Analysis Gene Expression Programming

A. Bleh* and G. Geiser†
German Aerospace Center (DLR), Cologne, Germany

Data-driven turbulence modeling has become an emerging field, aiming to overcome the weaknesses of classical Reynolds Averaged Navier-Stokes (RANS) models. One branch is Gene Expression Programming (GEP), which tries to find symbolic expressions for unknown functional dependencies. As an evolutionary algorithm it typically relies on many function evaluations. To reduce the computational cost, prior knowledge should be included where possible. When modeling functional dependencies in a physical context, the classical GEP is unaware of the physical dimensions of the involved quantities. Nevertheless, the validity of an expression in terms of its dimensions is a valuable hint towards its suitability and may improve the algorithms' performance. Therefore, in this work, we propose a new approach to consider physical dimensions within GEP. The new algorithm is evaluated and compared against existing approaches and applied on well-described turbomachinery test cases at transitional flow conditions.

I. Nomenclature

$C_{P,k}, C_\sigma$	=	Constants of the k-equation in the Wilcox $k-\omega$ model
D	=	Destruction term of the k-equation
g	=	Gravitational acceleration
γ	=	Transition factor
k	=	Turbulent kinetic energy
μ_m	=	Molecular viscosity
μ_t	=	Turbulent viscosity
∇u	=	Divergence of the velocity
n_{pop}	=	Number of individuals in a population
ω	=	Turbulent dissipation rate
p_{correct}	=	Probability that a generation contains the correct expression
P_k	=	Production term of the k-equation
P_{GEP}	=	Production term of the γ-equation to be modeled by GEP
$\|\Omega\|$	=	Rotation rate tensor norm
$\|S\|$	=	Strain rate tensor norm
ρ	=	Density
σ_k	=	Diffusion coefficient
τ_w	=	Wall friction
w_d	=	Distance to the closest wall

II. Introduction

Despite the increasing availability of computational power and consequently the access to scale-resolving flow simulation results, Reynolds Averaged Navier Stokes (RANS) turbulence modeling is still an essential backbone in industrial design processes. Therefore, the ability of these turbulence models to accurately model the flow of, e.g., turbomachinery designs is crucial for improving the efficiency of modern jet engines and in consequence the reduction of their emissions. The derivation of the RANS equations yields correlation terms, for which a closed-form expression

*Research scientist, German Aerospace Center, Institute of Propulsion Technology, Linder Höhe, 51147 Cologne, Germany
†Group leader, German Aerospace Center, Institute of Propulsion Technology, Linder Höhe, 51147 Cologne, Germany

could not yet be found. Consequently, solving these equations requires turbulence models, which inherently comprise empirical assumptions. Popular models are the group of linear eddy viscosity models. While being computationally efficient, these models often fail to accurately predict the flow. For example, this can be the case in the presence of separation or strong secondary flows. Data gained by experiments or scale-resolving simulations can provide information about weaknesses of such models.

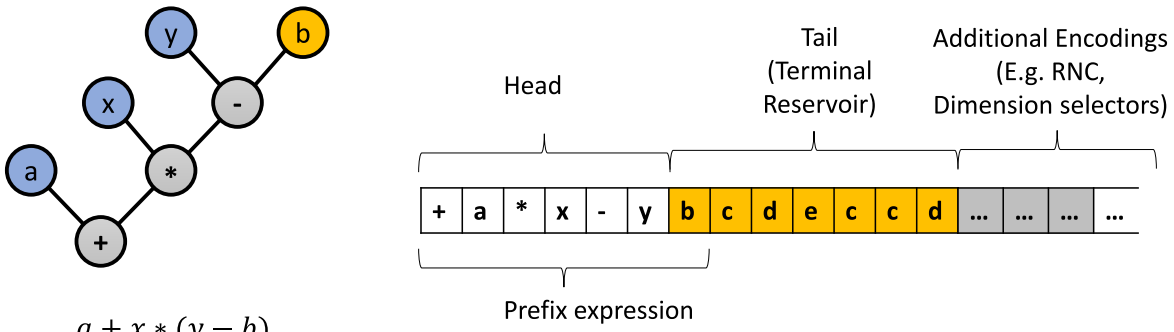
One option to exploit data by means of machine learning is the training of neural networks either in a direct fashion, as demonstrated, e.g., in [1] or by solving the inverse problem as shown in [2]. A drawback of neural networks is the difficulty to analyze and interpret their behavior. Therefore, evolutionary algorithms, which aim at finding symbolic expressions, have emerged over the recent years in the area of data driven turbulence modelling [3, 4].

The CFD-driven approach as described in [4] has grown in popularity over the recent years, as it circumvents the need for high resolution flow data and enables the use of experimental results as potential reference data. In analogy to the gradient based field inversion method [5], CFD driven evolutionary algorithms solve the inverse problem. Nevertheless, the need for CFD simulations for evaluating the fitness of each individual makes this approach computationally expensive. Therefore, approaches which aim at improving the convergence of evolutionary optimization algorithms are particularly interesting for the CFD-driven method. In this context Ma *et al.* [6] demonstrated the utilization of knowledge about the physical dimensions of the problem quantities within the evolutionary process.

In this publication, we build on this work and propose a new algorithm that is based on expanding a given expression with respect to its physical dimensions. The new method is described in section IV. To evaluate its potential, we apply it on a set of a priori known test expressions. Subsequently, we compare its performance against the approach described by Ma *et al.* [6] as well as the classical procedure which is based on defining nondimensional input features.

Finally, we validate the new approach by applying it to reference data of the turbine profiles T106C and T107A in order to obtain PDE based transition models.

III. Genetic Programming



(a) Expression tree (Phenotype)

(b) Gene containing the prefix expression (Genotype)

Fig. 1 Relation between the expression tree and its genetic representation.

Genetic Programming (GP) is an optimization technique which mimics natural evolution based on the "survival of the fittest" [7] principle. A randomly generated initial population of individuals is evaluated and single individuals are discarded based on their performance solving a given reference problem. Better performing individuals are recombined in a random fashion to form the succeeding generation.

In order to recombine two different manifestations (*phenotypes*) of individuals into a new child individual, each possible individual must be transferable into a unique linear numerical representation. This representation is denoted as the *genotype* of the individual.

A. Gene Expression Programming

Regarding mathematical expression trees, a purely random sequence of operators, variables, and constants would generally yield invalid expressions. In order to allow only valid expressions, Gene Expression Programming (GEP) puts additional constraints onto the genotype representation of an individual [8]. This is achieved by writing expressions in a linear prefix notation and providing a sufficiently large reservoir of terminal quantities in a tail part of the gene. Figure 1

demonstrates the relation between the expression tree (phenotype) and the corresponding gene (genotype). GEP is able to find global optima in discrete sampling spaces, making it a suitable tool to find symbolic expressions of unknown functional dependencies.

Additional arrays may be placed behind the genes tail section to encode additional information. In order to enrich the discrete evolutionary optimization algorithms with continuous random numerical constants (RNC), the RNC-GEP algorithm [8] utilizes such an additional array behind the genes tail. In this work, we also make use of an additional integer array, as described in more detail in section IV.C.

B. CFD-driven approach

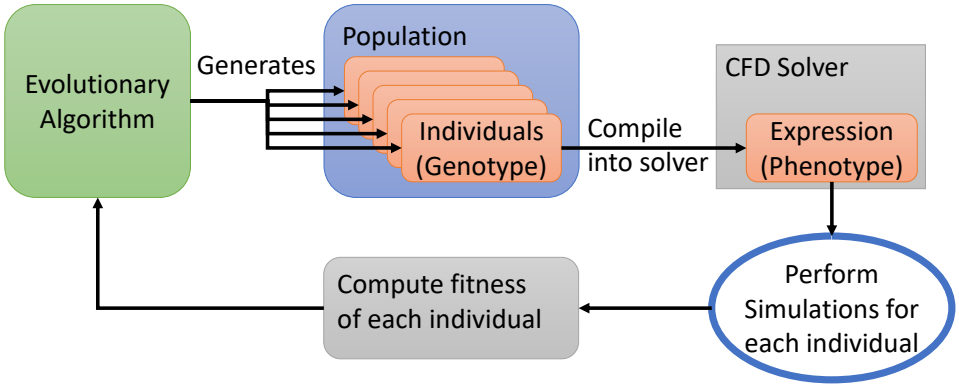


Fig. 2 Schematic visualization of the CFD-driven GEP algorithm.

To evaluate the fitness of each individual, we use the CFD-driven approach, as described in [4] and depicted in Fig. 2. This requires the execution of one or more RANS CFD simulations for each individual. After each simulation, a norm for the deviation between RANS and reference data is computed and added to the fitness. For better performance, the individual expressions are compiled at runtime using Just-In-Time (JIT) compilation.

Nevertheless, using this approach, each individual requires the execution of one or more CFD simulations. Therefore, it is of particular interest, to save the execution of CFD evaluations whenever possible or to reduce the total number of required evaluations until convergence. In this work we focus on two principles, to reduce the total number of required CFD evaluations:

- **Omitting evaluations:** This can be done, e.g., by skipping individuals that have already been evaluated. In gene expression programming, it is usually the case, that multiple different chromosomes yield an equivalent mathematical expression (e.g., x vs. $x + y/y - 1$). In this work, we use the python library *Sympy* [9] to identify equivalent expressions. Furthermore, individuals might be discarded from CFD evaluation, because their respective phenotype does not fulfill certain constraints like the occurrence of *NaN* or *inf* (e.g., $x/(1-1)$) or dimensional constraints as described in more detail in section IV.
- **Likelihood design:** Designing the optimization problem, we usually aim towards a high probability that the best or at least a good candidate is found at an early iteration. This is already inherently done, e.g., by considering only input quantities that may be relevant for the problem at hand, or by reducing the set of possible operators or functions according to a priori assumptions. Based on this, the probability of a good candidate to occur shall increase with each iteration, due to selective recombination of candidates with high fitness values. Nevertheless, in the first iteration, the generation of individuals is fully random. Hence, the probability that a certain expression occurs in the first generation is equivalent to the probability that it occurs in a randomly generated sample. Consequently, if we have knowledge or a reasonable guess about the structure of the correct or best fitting expressions, we might want to decrease the likelihood of expressions which contradict these assumptions. In contrast, if we do not have any knowledge about a subset of possible expressions, we mostly want to establish equal likelihood between all possible members of this subset.

With respect to these two mechanisms we compare different strategies to enable knowledge about physical dimensions in order to improve the output of GEP investigations.

IV. Accounting for physical dimensions

Floating point numbers in a computer are completely independent of the dimension or the physical quantity they are representing. They do not contain any information except their numerical value. In most applications dimensional significance is only formed virtually by the context in which a user or programmer decides to use these numbers.

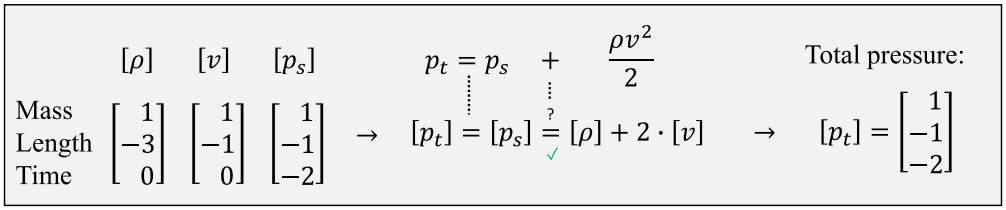


Fig. 3 Vector representation of physical units of input and output quantities for the example of the total pressure formula.

If physical dimensions shall be used or interpreted by an algorithm, they require an appropriate representation. Ma *et al.* [6] are using a prime number factorization in order to derive a boolean criterion whether a given expression is valid in terms of its physical dimensions or not.

For this work we use a vector based approach depicted in Fig. 3, which stores the physical dimensions as a fixed size integer vector of base-units. These base units may comprise SI-units, but do not have to. Each position in the vector represents one base-unit. Positive integers represent physical base-units in the numerator whereas negative integers represent base-units in the denominator. To determine the physical dimension of a product (quotient) of two inputs, we simply have to add (subtract) their respective unit-vectors. Regarding summation or subtraction in a physically valid expression, the sum or difference shall have the same physical dimension as its respective summands or minuend and subtrahend. Nevertheless, if they have different physical dimensions, the expression is considered physically invalid. Certain functions such as logarithm or trigonometric functions can yield invalid dimensions as well. In the context of this work, only whole numbers are allowed for the dimensions. This implies that, e.g., the logarithm of a term with nonzero unit-vector is considered to be invalid. Depending on the context, certain functions might also expect certain dimensions. For instance, the trigonometric functions might either expect nondimensional radians or explicitly degrees. We propose the following set of rules as suitable for the cases investigated in this paper:

1) **Multiplication** Add units: $c = a \cdot b \longleftrightarrow [c] = [a] + [b]$
2) **Division** Subtract units: $c = a/b \longleftrightarrow [c] = [a] - [b]$
3) **Addition/ Subtraction** Check equality: $c = a \pm b \longleftrightarrow$ if $[a] \neq [b]$ then invalid

In the following, we investigate three approaches to consider physical dimensions in GEP.

A. Manual approach

The standard method to account for physical dimensions in almost all publication related to data-driven modelling of physical phenomena, is to create a normalized context, which is independent of any physical dimension. This is typically achieved by providing nondimensional products of physical input features as well as a dimensionalization factor for the output quantity in case it has a physical dimension. In this work, we further denote this approach by the term *manual* nondimensionalization. Manually designing a set of nondimensional input features effectively reduces the likelihood of expressions with nonmatching physical dimensions to zero. On the other hand, the choice of input features may have an undesired influence on the probability of expressions with valid physical dimensions.

This effect is shown in Fig. 4. Consider the nondimensional input feature $\Pi_1 = b/a$, with a and b being dimensional quantities of the same dimension. The symbol T denotes an arbitrary terminal. In this example all tokens would terminate the expressions if chosen for T. The likelihood of the expression $b/a = \Pi_1$ (Genome/prefix notation $|\Pi_1|T|$) to appear in a random sample is higher than the occurrence of the expression $b/a = 1/\Pi_1$ (Genome/prefix notation $|/|1|\Pi_1|T|$). This is due to the more complex genome representation of the latter expression. Therefore, unfavorably

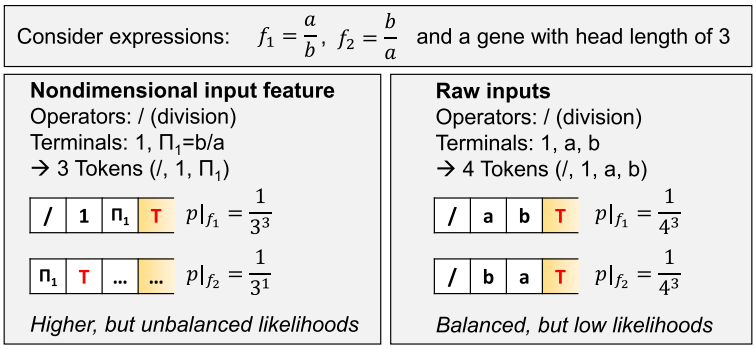


Fig. 4 Demonstration how the selection of nondimensional input features causes unbalanced likelihoods for expression of equal complexity.

selected nondimensional input features may unbalance the random expression probabilities in an undesired way.

Consequently, we would like to have an algorithm, which yields balanced likelihoods, yet without reducing the probability of valid expressions in terms of physical dimensions.

B. Discarding approach

As mentioned before, in case there are no assumptions or prior knowledge about an adequate choice of nondimensional products as input features, it would be desirable to not harm a balanced likelihood between two possible valid expressions.

A method which allows the direct use of the dimensional input units, hence preserving a balanced likelihood, is proposed by Ma *et al.* [6], who introduce a dimensional homogeneity constraint. Using the raw input quantities, their approach essentially corresponds to the principle shown in Fig. 4 on the right. Since the generation of invalid expressions in terms of physical dimensions is allowed, the probability of all valid expressions is reduced in return. The authors counteract this lower likelihood per individual by increasing the population size. To account for the increased problem complexity, they are using a prime number factorization to determine a boolean criterion whether a given expression has matching physical dimensions or not. In case of mismatching dimensions, the expression is removed from the competition by assigning a fixed large fitness value, hence saving the need for an expensive CFD evaluation. This way, invalid individuals can be quickly discarded, which allows to save considerable computational resources by reducing the number of required CFD simulations. Thus, this method follows the principle of omitting evaluations described in section III.B.

The fitness of discarded individuals needs to be set to a very high fixed value. Consequently, there is no competition between invalid members, although they are still part of the selection process. This property may harm convergence, since randomly generated sample expressions using dimensional quantities will yield invalid expressions with a high probability, ultimately counteracting the evolutionary development. In section V we will investigate whether the performance gain due to discarded individuals might compensate the effect of reduced competition.

C. Dynamic Dimension Normalization

The discarding approach by Ma *et al.* [6] described in section IV.B heavily relies on the principle of discarding individuals.

We propose a new method, which does not require discarding individuals based on their dimensional homogeneity, yet which also uses the raw dimensional input quantities directly. The method is intended to combine the advantages of the manual approach and the discarding approach. First, it shall preserve a balanced likelihood as is the case for the discarding approach. Second, the new approach is expected to yield better convergence, since competition is preserved due to all individuals being evaluated.

This is achieved by rendering the expression from the genome using a rule, which always produces matching physical dimensions. We do this by fixing or extending the raw expression we get from the original GEP algorithm. This is achieved by adding an appropriate normalization factor for each subtree of a given expression, where dimensions do not match. Therefore, we further denote the new approach as *Dynamic Dimension Normalization* (DDN).

In general, there is an infinite number of possible ways to fix an expression in order to achieve matching dimensions. Since evolutionary algorithms produce random expressions, there is no ultimate way to do this. Nevertheless, we seek

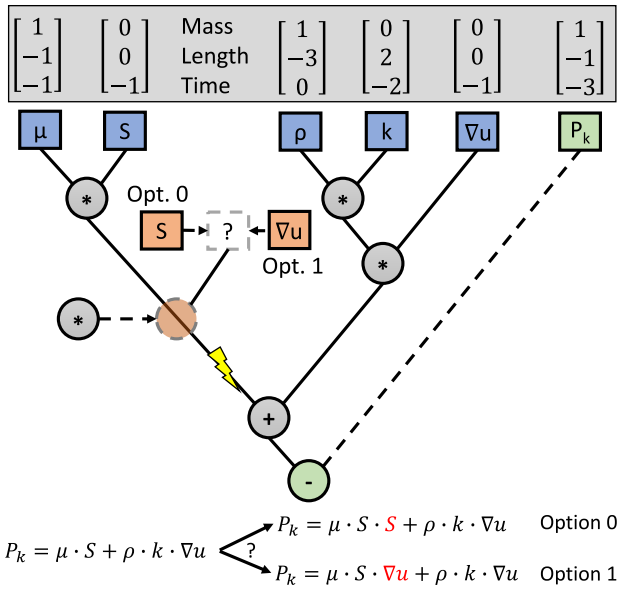


Fig. 5 Modification of expressions in the DDN approach.

for an algorithm using the following premises:

1) **Deterministic** While the mutation and crossing of individuals is stochastic, the process to convert the genotype into the phenotype shall be deterministic.
2) **Simplicity** The algorithm shall fix the expression with the least or close to the least amount of added terminals or operators.

First, the individual is converted from the chromosome (genotype) into an expression (phenotype), as described in the original paper on GEP by Ferreira *et al.* [8]. Next, the expression is parsed into a tree as shown in Fig. 5. Afterward, the unit-vectors of the input features are propagated through the expression tree, using the rules defined in section IV. It may (and generally will) happen that the dimension of the two summands of an addition do not match. If this is the case, we assume that the second summand (or the subtrahend) yields the correct dimension. This implies that in order to fix the expression, the first summand (or minuend) must be normalized using the given set of input quantities. Figure 5 demonstrates this procedure. An additional multiplication is introduced in the affected branch of the expression tree and the required quantity to create matching dimensions is multiplied to the left summand (or minuend).

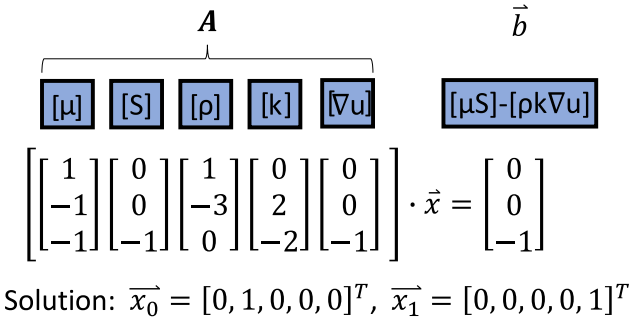


Fig. 6 Solution of the diophantine equation system based on the example shown in Fig. 5.

The possible solutions to normalize the first summand in order to match the second summand can be determined by solving the diophantine equations [10]. All dimension vectors of the normalization input features are written into an integer left hand side matrix A. The difference between the dimensions of the first and second summand yield the right hand side integer vector b. Figure 6 demonstrates the solution of the diophantine equation system based on the example shown in Fig. 5. For this publication, we solve the resulting equation system using the publicly available python library *Diophantine* [11]. When solving the diophantine equation system there are three cases to be distinguished:

1) **Exactly one solution (generally rare)** In this case we can simply take the given solution without any further care.
2) **No solution (problem in setup)** This case usually hints at a problem in the setup of the input features, i.e., the dimension of one of the inputs cannot be expressed by any product of the normalization features. Nevertheless, this might also hint at the existence of a missing physical constant.
3) **Multiple solutions (common case)** Diophantine equation systems usually feature an infinite number of nontrivial solutions. The LLL algorithm described in [10] aims at finding small integer solutions. This means that e.g. the solution $[1, 1]$ would be discarded, if the solution $[1, 0]$ exists. For the scope of this work, this restriction is tolerated. Nevertheless, in order to make the normalization more flexible, allowing solutions up to a certain norm might be beneficial. Yet, even if we restrict solutions to the smallest norm, there is generally a finite set of different smallest solutions. This is, e.g., the case if multiple different normalization features with the same physical dimensions are provided. Since we do not know which solution is most suitable to normalize the given term, the choice is encoded within the genome. Therefore, we add an array of integers to the genome, which shall represent the index of the solution to be selected in case a term needs to be normalized. In the worst case, an equation system has to be solved for all operators in an expression as well as for nonmatching output dimension. Hence, the index array has to provide at least as many numbers as there are entries in the gene's head, plus one additional entry for the output dimension. Thereby, each position in the normalization solution index array corresponds to the respective position in the genes head, except for the last which is used to fit a potential mismatch of the output unit. We do not know a priori the number of possible solutions, yet we want an approximately equal probability for each solution. This is achieved by using a sufficiently large range of possible integer values and taking the modulus with respect to the number of solutions to compute the actual solution index. For the mutation and crossover operations to modify the normalization index array, we use the same operators that already exist for the additional index array required by the RNC-GEP algorithm [8].

V. Compare performance of dimensional aware GEP approaches

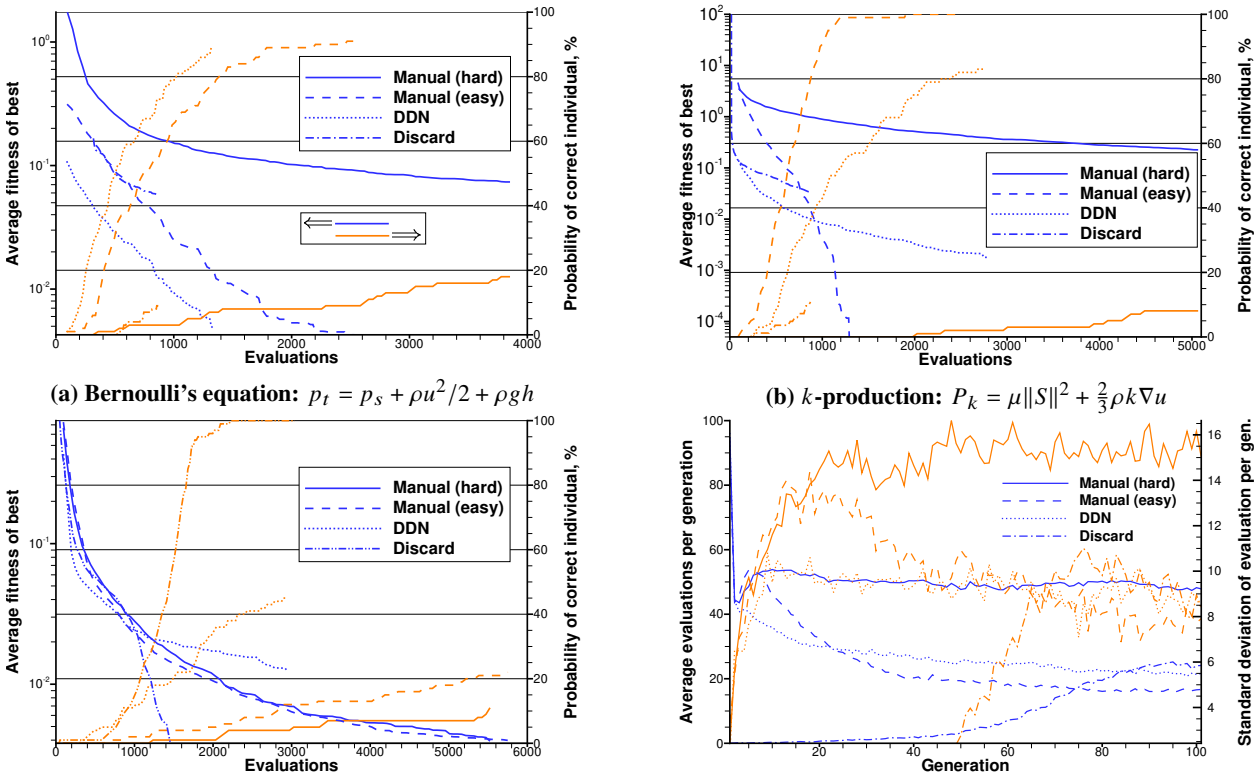
In section IV we described different methods based on the consideration of physical dimensions. In order to evaluate the performance of the different approaches, they are applied on sample expressions multiple times in order to achieve a certain statistical significance. Afterward, the averaged convergence history is being compared. All investigations are conducted using the python library *geppy* [12], which builds on the evolutionary algorithm framework *DEAP* [13].

Since the approach of [6] heavily relies on discarding members, hence saving computational cost of related CFD simulations, it would be disproportionately disadvantageous, if we only consider the convergence over the raw number of individuals. Therefore, we normalize the convergence history with the number of individuals that actually would have to be evaluated in a CFD-driven approach. Besides individuals which are invalid with respect to their physical dimensions, expressions which have already been evaluated once are discarded as well. That is, if a different individual yields an equivalent expression, it will be counted as already evaluated. This is important since different approaches might yield a different probability of producing equivalent expressions.

Table 1 Sample expression for comparison of the dimensional aware approaches.

Fitness	Expression	Manual setup (easy)	Manual setup (hard)
1. Bernoulli	$p_t = p_s + \rho u^2/2 + \rho g h$	$p_s \cdot f\left(\frac{\rho u^2}{p_s}, \frac{\rho g h}{p_s}\right)$	$p_s \cdot f\left(\frac{p_s}{\rho u^2}, \frac{p_s}{\rho g h}, \frac{\rho u^2}{\rho g h}\right)$
2. Production k	$P_k = \mu\|\|S\|\|^2 + \frac{2}{3}\rho k \nabla u$	$\mu\|\|S\|\|^2 \cdot \left(\frac{\rho k \nabla u}{\mu\|\|S\|\|^2}\right)$	$\mu(\nabla u)^2 \cdot f\left(\frac{\nabla u}{\|\|S\|\|}, \frac{\rho k}{\mu\|\|S\|\|}\right)$
3. Fictional	$(\mu_t - \mu_m)/(\|\|\Omega\|\| + \|\|S\|\|)$	$\frac{\mu_t}{\|\|\Omega\|\|} \cdot f\left(\frac{\mu_m}{\mu_t}, \frac{\|\|S\|\|}{\|\|\Omega\|\|}\right)$	$\frac{\mu_t}{\|\|\Omega\|\|} \cdot f\left(\frac{\mu_m}{\mu_t}, \frac{\|\|\Omega\|\|}{\|\|S\|\|}\right)$

The sample expressions that are used to evaluate the approaches are listed in Table 1. Expression 1 is the Bernoulli equation for total pressure and expression 2 is the production term of the turbulent kinetic energy in the Wilcox $k - \omega$



(a) Bernoulli's equation: $p_t = p_s + \rho u^2/2 + \rho g h$

(b) k-production: $P_k = \mu \|S\|^2 + \frac{2}{3}\rho k \nabla u$

(c) Fictional expression: $(\mu_t - \mu_m)/(\|\Omega\| + \|S\|)$

(d) Evaluated cases in each generation for Bernoulli

Fig. 7 Averaged convergence of 100 repeated GEP evaluations for each test case.

model [14]. Hence, these expressions are actual formulas taken from the CFD context and the result of a physically motivated derivation. Expression 3 is fictional, yet might be the result of an empirical regression. It features the sum and difference of different input quantities with equal physical dimensions. It is expected that this structure poses a challenge particularly for the manual nondimensionalization.

A. Comparison with the manual approach

When using nondimensional products to account for physical dimensions, the convergence of the GEP optimization may depend on the design and choice of input features. Usually, the optimal choice is not known a priori. Hence, we evaluate two different sets of nondimensional input features for each expression. The first set is chosen in an optimal way, hence denoted by the term *easy*. The attribute optimal in this case means that the expression required to yield the original formula is found early with a high probability during the GEP optimization. This is roughly the case when the expression is very short. The number of equivalent expressions due to commutation and distribution laws is relevant as well, nevertheless we neglect this effect. The second set is denoted by the term *hard*, since it requires a more complex expression to yield the original formula. E.g., the first set for Bernoulli's equation would require the expression $1 + 2/3\Pi_1$ (prefix: $+1\cdot\Pi_1$ / 2 3), whereas the second set would at least require the expression $\Pi_1^{-1}\left(\Pi_1^{-1} + 2/3\Pi_2\right)$ (prefix: $\cdot$ / $1\Pi_1$+/1 $\Pi_1\cdot$ / 2 3 Π_2). When comparing the prefix expressions, the different complexity of the two expressions becomes even clearer.

B. Performance of the different algorithms

Figure 7(a-c) shows the averaged results of 100 GEP optimizations for each test expression. The left y-axes represent the averaged fitness of the best individual in each generation. The right y-axes represent the probability p_{correct} that the best individual in a given generation of size n_{pop} is equivalent to the correct expression:

$$p_{\text{correct}}(g) = \frac{n_{\text{correct}}(g)}{n_{\text{pop}}}$$

The x-axes denote the average number of virtual CFD evaluations n_{CFD} that would have to be conducted in case of a CFD-driven investigation:

$$\overline{n_{\text{eval}}(g)} = \frac{1}{n_{\text{GEP}}} \sum_{i=0}^{g} \sum_{j=0}^{n_{\text{GEP}}} n_{\text{CFD}}$$

The symbol n_{GEP} denotes the total number of repeated gene expression investigations that have been performed. For each case we conducted 100 GEP investigation over 100 generations. Each generation comprised 100 individuals except for the discarding approach. Since most of the individuals yield invalid dimensions and thus are discarded, this method requires a much larger population to be effective. We scaled the number of individuals to 20 000 for each case. Different configurations might yield different results for either case and method. Nevertheless, a more thorough parameter study is beyond the scope of this work. To provide insight in the different discarding characteristics of each method, Fig. 7d shows the average number of virtual CFD evaluations in each generation. The discarding approach requires a certain start-up phase until a sufficiently large stock of valid expressions is available. Nevertheless, since invalid expressions are mostly negligible in terms of performance, this does not necessarily harm the overall convergence.

As expected the manual approach converges faster for the easy configuration compared to the harder configuration for all cases. Consequently, the average fitness of the best individual is lower as well. The DDN approach is converging an order of magnitude faster than the hard manual configuration for all cases. For expressions 1 and 2 the optimal manual and the DDN approach yield similar performances, with the latter being better for expression 1 and inferior for expression 2. It should be noted that the DDN approach achieved this performance without any a priori knowledge about a suitable factorization.

For the fictional expression 3 the dimension aware method yield a significantly higher probability to yield the original formula. This supports the assumption formulated above that sums of dimensional constants in a target expression may be particularly difficult to be found by the manual approach. Interestingly, the discarding approach is also better than the DDN approach in this case. One reason might be that the dynamic normalization would not normalize a sum with another sum, thus decreasing the likelihood of such expression compared to the discarding method. E.g., in the case of expression 3 an occurrence of $(\mu_t - \mu_m)$ would always be normalized with $\|\Omega\|$ or $\|S\|$, but not with $(\|\Omega\| + \|S\|)$. For expressions 1 and 2 the discarding approach is between the DDN and manual approach. Nevertheless, its convergence seems strongly dependent on the number of individuals per generation. A more detailed investigation is beyond the scope of this paper. Nevertheless, we can summarize that the methods which use input quantities directly and enable knowledge about their dimensions are mostly superior or yield convergence close to an optimal manual configuration.

VI. Application on transitional turbomachinery test cases

A. Test case selection

In order to validate the proposed method of dynamic dimension normalization, we apply it to two turbomachinery test cases featuring a transition induced separation. The turbine profiles T106C and T107A at Reynolds numbers of 80 000 as well as 700 000 are used for training. The profile T107A at a Reynolds number of 300 000 is used for validation.

The original formulation of the $k - \omega$ model from Wilcox [14] does not predict the flow separation, which can be observed when looking at the LES data for T106C published in [15]. It further fails to predict the separations for T107A observed in the experiments by Hoeger [16]. This profile is an interesting testcase in a sense that the mechanism of the transition related separation shifts from a separation induced transition at lower Reynolds numbers to a mere bypass transition at higher Reynolds numbers. This effect is thoroughly investigated in [17]. Therefore, this selection of testcases features a variation in geometry as well as a nontrivial variation in different effects due to Reynolds number variation.

B. Model setup

A typical approach to augment turbulence models to better predict transitional flows is to apply a correction factor on the turbulent kinetic energy production term. This correction has been done similarly for other turbulence models in

several publications [2, 5, 18, 19]. Nevertheless, in order to consider the nonlocal nature of transition, the production term is not corrected directly using an algebraic expression. Instead, the correction factor is modelled by an additional convection-diffusion equation, of which we model the production term using GEP generated expressions. Additionally, the values of the resulting correction term are capped to a range between a value of close to zero and two, yielding the following model adaption:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_j}\left(\rho u_j k\right) = \underbrace{\gamma^*}_{\text{correction}} \overbrace{\tau_{ij}\frac{\partial u_i}{\partial x_j}}^{\text{production}} - C_{P,k}\rho\omega k + \frac{\partial}{\partial x_j}\left[\left(\mu + C_\sigma \mu_t\right)\frac{\partial k}{\partial x_j}\right]$$

$$\gamma^* = \max(\min(\gamma, 2), 10^{-5})$$

$$\frac{\partial\left(\rho\gamma\right)}{\partial t} + \frac{\partial\left(\rho u_j \gamma\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_\gamma \mu_t\right)\frac{\partial \gamma}{\partial x_j}\right] + P_{\text{GEP}}$$

Note that the constants and capping boundaries are assumed fixed, based on a priori experience and are not subject to optimization in the scope of this work. This approach can be interpreted in two different ways. First, it can be considered as finding a new source term for a slightly modified version of the γ transition models described in [20]. On the other hand, modelling source terms of partial differential equations instead of directly using algebraic expression can be considered as a simple and mesh independent way of including nonlocal effects into machine learning models.

C. Fitness evaluation

The turbine profile T106C at a Reynolds number of 80 000 as well as the profile T107A at Reynolds numbers 100 000 and 300 000 are used as training data. The turbine profile T107A at a Reynolds number of 700 000 is used as validation data.

First, the fitness of the investigated individual is evaluated by executing one CFD simulation with the resulting expression for each of the three training cases. Afterward, the deviation of the CFD results from the reference data is calculated. The deviation from reference data is normalized in a way that it yields a value of 1 for the default $k-\omega$ model for each training case. Hence, all three cases are prioritized equally. For T106C the pressure distribution of the whole blade is used as reference data. Regarding the experimental data of T107A, we only use the tail area of the blade. Due to the way in which the experimental data was extracted, there might be inaccuracies when mapping the nondimensional x-positions back onto the upstream parts of the blade surface.

All CFD computations have been conducted using the CFD code TRACE developed at the German Aerospace Center [21].

VII. Results for the CFD-driven GEP evaluation

A. Convergence

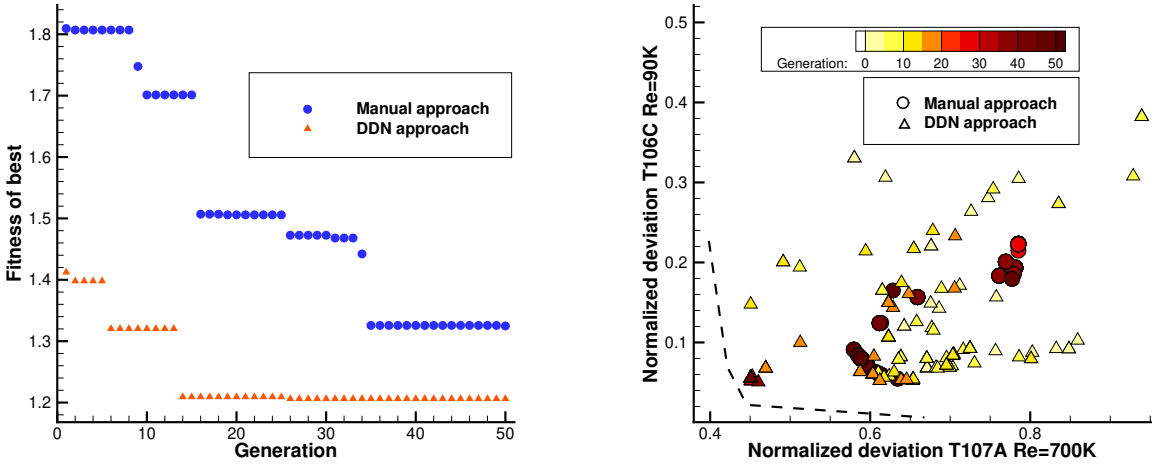
We applied the manual approach as well as the DDN approach on two CFD-driven GEP optimizations, in order to find potential transition models focused on turbomachinery cases. Since we do not have any a priori knowledge about the structure of the best possible model, the choice of nondimensional input features for the manual approach will likely not be optimal. For the DDN approach, we provide the following quantities to model the production term:

$$P_\gamma = f(\gamma, \|S\|, \|\Omega\|, \rho, w_d, \mu_t, \mu_m, k, \omega, \tau_w)$$

For the manual approach, these inputs are combined into a set of nondimensional input features:

$$P_\gamma = f(\gamma, \frac{\|S\|}{\|\Omega\|}, \frac{k\rho}{\tau_w}, \frac{\omega w_d}{\sqrt{k}}, \frac{\mu_t}{\mu_m})$$

Furthermore, we enable the addition of random numerical constants (RNC) [8]. This algorithm adds further slots to the genes as described in section III.A. Nevertheless, it is fully compatible with the newly proposed approach.



(a) Fitness of the best individual for each generation **(b) Pareto front for T106C and T107A (Re=700K)**

Fig. 8 Convergence of the GEP investigation for the manual and the DDN approach.

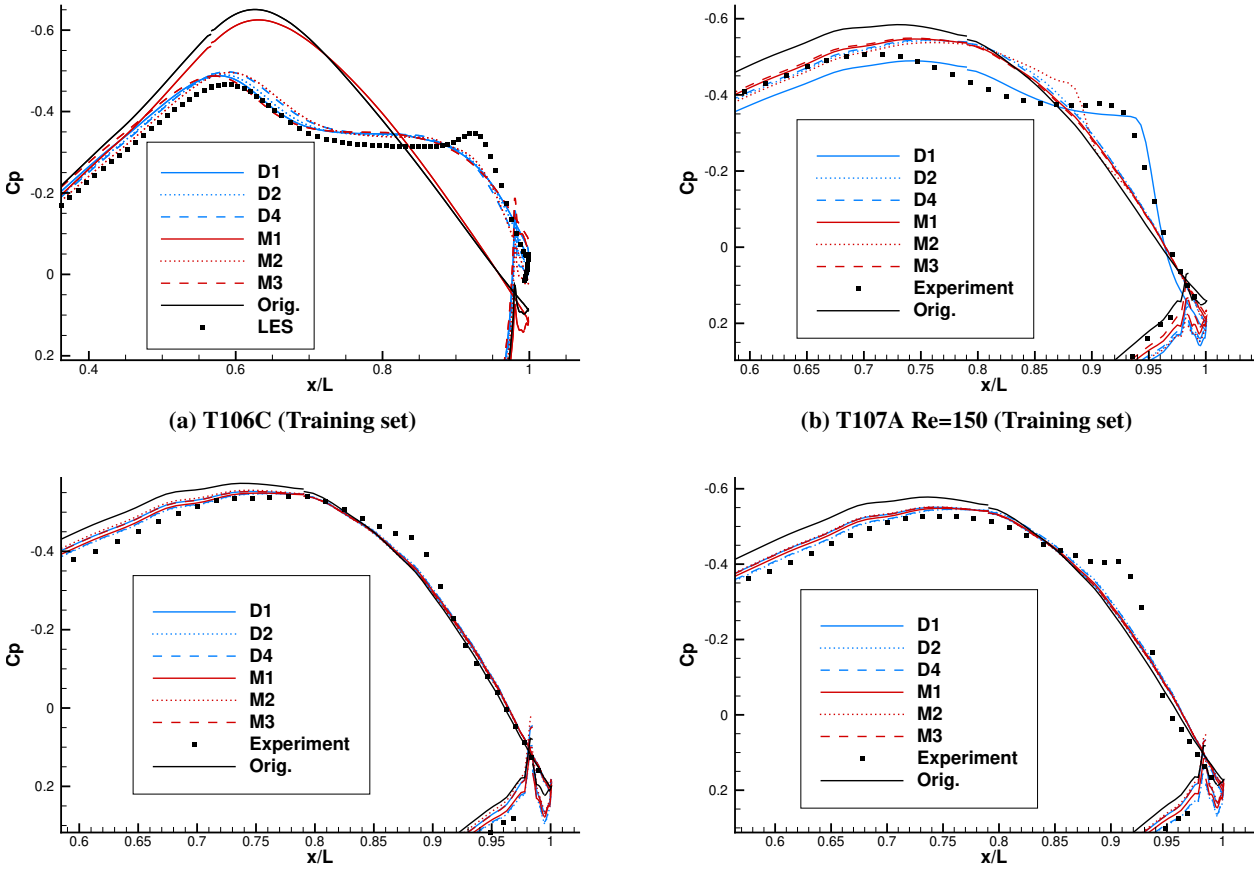
Figure 8a shows the overall fitness value of the best individual per generation. Since elitism is enabled, the values are decreasing monotonously. Given these two executions, the newly proposed approach converges faster and to a lower minimum. The pareto front depicted in Fig. 8b further indicates that the DDN approach produces more individuals with a high fitness compared to the manual approach in this case. This behaviour is generally in accordance with the observations demonstrated in section V. It shows that the new approach has the potential to improve the performance of a GEP investigation. Nevertheless, since evolutionary algorithms are highly stochastic, the execution of only one optimization per test case is not sufficient to derive a general statement. Furthermore, the selection of another set of nondimensionalized input features may yield different results. Due to the cost of a CFD-driven GEP evaluation this is beyond the scope of this paper.

B. CFD results

Table 2 lists a selection of the best and well performing individuals for both approaches. Floating point numbers generated by the RNC algorithm are rounded. Expressions from the manual approach are indicated by the letter *M*, whereas *D* denotes the DDN approach. Expression D4 and M3 have the best overall fitness. Expressions M1, M2, D1, D2 and D5 yield a minimal deviation for a certain case (marked with a box). Expression D3 has been chosen due to its low complexity. The remaining deviations from the reference data are an order of magnitude lower for test case T106C when compared to test cases using the profile T107A. This also holds for the similar Reynolds number of 150 000, yet less severe. Note that T106C is using an LES solution as reference data, whereas T107A is compared against experimental data. This may be one reason for the remaining deviations. Nevertheless, for higher Reynolds numbers, the transition mechanism of turbine profile T107A changes as mentioned in section VI.A and described in more detail in [17]. Therefore, it is expected that the transition at higher Reynolds numbers is more difficult to be captured. The validation case T107A at a Reynolds number of 300 000 yields deviations at the order of magnitude of the lower Reynolds numbers for the same blade geometry.

Figure 9 provides a more detailed insight into the performance of a subset of the augmented models. It shows the pressure distribution along the trailing part of the suction sides of the corresponding blades. For the lower Reynolds numbers of testcase T107A and T106C sown in Fig. 9a and 9b the models D1 and M2 are able to capture the separation characteristics. While the shape of the separation for T107A at Reynolds number 150 000 in Fig. 9b is captured better by model D1, the overall pressure level upstream the separation is worse compared to the other models. For higher Reynolds numbers as shown in Fig. 9d and 9c, all models are able to improve the overall pressure level. On the other hand, they fail to predict the characteristic transition related separation.

In summary, the manual approach as well as the DDN approach were able to find transition models which reduce the



(a) T106C (Training set)

(b) T107A Re=150 (Training set)

(c) T107A Re=700 (Training set)

(d) T107A Re=300 (Validation set)

Fig. 9 Performance of a selection of the best performing models.

Table 2 Best and well performing expression found by the manual and DDN approach

Id	Expression	Fitness	T107A Re=150k	T107A Re=700k	T106C Re=80k	T107A Re=300k
				Training		Validation
M1	$0.275 - \frac{\nu_m}{\nu_t} + 1.11\frac{\tau_w}{\rho k}$	2.056	0.474	0.664	0.918	0.519
M2	$\gamma + \frac{\omega w_d}{\sqrt{k}} + \frac{\nu_t}{\nu_m} - \frac{\nu_m}{\nu_t}$	1.613	0.689	0.882	0.042	0.625
M3	$2.998 + \frac{\omega w_d}{\sqrt{k}} - \frac{\nu_m}{\nu_t}\|\|$	1.322	0.580	0.652	0.091	0.593
D1	$\|\Omega\|\left(\frac{2}{\omega} + 3\frac{\nu_m}{\rho k} + \frac{2.103}{\|S\|}\right) - \frac{\nu_t}{\nu_m}$	1.209	0.587	0.558	0.063	0.595
D2	$\frac{(\|S\|+2\omega)(\|\Omega\|(\rho k-\nu_t\omega)+\rho k\omega)}{\|S\|\rho k\omega}$	1.223	0.448	0.729	0.0465	0.469
D3	$\omega\left(\frac{2}{\|S\|} - \frac{\nu_t}{\rho k}\right)$	1.214	0.450	0.708	0.056	0.478
D4	$\omega\frac{\|S\|\rho k-\|\Omega\|(\|S\|\nu_t-\nu_m\omega)}{\|S\|^2\rho k}$	1.206	0.451	0.70	0.056	0.470
D5	$\omega\frac{\|S\|(\gamma\nu_t+2\nu_m-2\nu_t)+\rho k}{\|\Omega\|\rho k}$	1.487	0.446	0.758	0.282	0.459

deviation to the given reference data significantly. Additionally, some models are able to predict the separation for lower Reynolds numbers. For the given cases, the DDN approach found more well performing models in a shorter period of time.

VIII. Conclusion

We proposed a new method to enhance the convergence of GEP optimizations by utilizing knowledge about the physical dimensions of the relevant input quantities. It extends the genes with encoded information to fix invalid physical dimensions of given expressions. Using a validation set of a priori known reference formula, it was shown that the new approach may increase the probability of finding the correct function representation during an early generation. This is particularly the case, if no knowledge about an optimal set of nondimensional input features is available a priori. In future works, it should be evaluated more thoroughly how the observed advantage depends on the structure of the target formula, the type of required operators, the number of input parameters and the number of relevant physical dimensions.

Furthermore, we applied the new method to find potential transition models using a CFD-driven GEP optimization. Several expressions could be found that are able to predict the flow separation of two turbine profiles at low Reynolds numbers. Nevertheless, a more general transition model will require the use of additional reference testcases. Also the use of additional operators such as *min* or *max* should be considered.

References

[1] Ling, J., Kurzawski, A., and Templeton, J., "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, Vol. 807, 2016, pp. 155–166. https://doi.org/10.1017/jfm.2016.615.

[2] Parish, E. J., and Duraisamy, K., "A paradigm for data-driven predictive modeling using field inversion and machine learning," *Journal of Computational Physics*, Vol. 305, 2016, pp. 758–774. https://doi.org/10.1016/j.jcp.2015.11.012.

[3] Weatheritt, J., and Sandberg, R., "A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship," *Journal of Computational Physics*, Vol. 325, 2016, pp. 22–37. https://doi.org/10.1016/j.jcp.2016.08.015.

[4] Zhao, Y., Akolekar, H. D., Weatheritt, J., Michelassi, V., and Sandberg, R. D., "RANS turbulence model development using

CFD-driven machine learning," *Journal of Computational Physics*, Vol. 411, 2020, p. 109413. https://doi.org/10.1016/j.jcp.2020.109413.

[5] Holland, J., Baeder, J., and Duraisamy, K., "Field Inversion and Machine Learning With Embedded Neural Networks: Physics-Consistent Neural Network Training," 2019. https://doi.org/10.2514/6.2019-3200.

[6] Ma, W., Zhang, J., Feng, K., Xing, H., and Wen, D., "Dimensional homogeneity constrained gene expression programming for discovering governing equations from noisy and scarce data," *arXiv*, 2022. https://doi.org/10.48550/arXiv.2211.09679.

[7] Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*, Bradford Books, 1993, p. 18.

[8] Ferreira, C., "Gene Expression Programming: A New Adaptive Algorithm for Solving Problems," *Complex Systems*, Vol. 13, No. 2, 2001. URL https://www.complex-systems.com/abstracts/v13_i02_a01/.

[9] Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., Curry, M. J., Terrel, A. R., Roučka, Š., Saboo, A., Fernando, I., Kulal, S., Cimrman, R., and Scopatz, A., "SymPy: symbolic computing in Python," *PeerJ Computer Science*, Vol. 3, 2017, p. e103. https://doi.org/10.7717/peerj-cs.103.

[10] Havas, G., Majewski, B. S., and Matthews, K. R., "Extended GCD and Hermite Normal Form Algorithms via Lattice Basis Reduction," *Experimental Mathematics*, Vol. 7, No. 2, 1998, pp. 125–136. https://doi.org/10.1080/10586458.1998.10504362.

[11] Close, T. G., "Diophantine," GitHub Repository, 2017. URL https://github.com/tclose/Diophantine.

[12] Shuhua Gao, Morgan, A., and M, T., "ShuhuaGao/geppy: Release 0.1 of geppy," , 2020. https://doi.org/10.5281/zenodo.3946297.

[13] Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., and Gagné, C., "DEAP: Evolutionary Algorithms Made Easy," *Journal of Machine Learning Research*, Vol. 13, 2012, pp. 2171–2175. URL https://github.com/DEAP/deap.

[14] Wilcox, D. C., "Reassessment of the scale-determining equation for advanced turbulence models," *AIAA Journal*, Vol. 26, No. 11, 1988, pp. 1299–1310. https://doi.org/10.2514/3.10041.

[15] Morsbach, C., and Bergmann, M., "Critical Analysis of the Numerical Setup for the Large-Eddy Simulation of the Low-Pressure Turbine Profile T106C," *Direct and Large Eddy Simulation XII*, Springer International Publishing, Cham, 2020, pp. 343–348. https://doi.org/10.1007/978-3-030-42822-8_45.

[16] Hoeger, "Experimentelle Untersuchungen am Turbinengitter T107. Teil 2: Grenzschichtmessungen mit Dünnfilmen und Pitotsonde," Tech. Rep. IB 129-84/39, DFVLR, 1984.

[17] Marciniak, V., "Phenomenological transition modelling for turbomachinery flows," Doctoral thesis, Ruhr-Universität Bochum, 2016. URL https://nbn-resolving.org/urn:nbn:de:hbz:294-47825.

[18] Ferrero, A., Iollo, A., and Larocca, F., "Field inversion for data-augmented RANS modelling in turbomachinery flows," *Computers and Fluids*, Vol. 201, 2020, p. 104474. https://doi.org/https://doi.org/10.1016/j.compfluid.2020.104474.

[19] Jäckel, F., "A Closed-form Correction for the Spalart-Allmaras Turbulence model for Separated Flows," *AIAA SCITECH 2022 Forum*, 2022. https://doi.org/10.2514/6.2022-0462.

[20] Menter, F. R., Smirnov, P. E., Liu, T., and Avancha, R., "A One-Equation Local Correlation-Based Transition Model," *Flow, Turbulence and Combustion*, 2015, pp. 583–619. https://doi.org/10.1007/s10494-015-9622-4.

[21] Geiser, G., Wellner, J., Kügeler, E., Weber, A., and Moors, A., "On the Simulation and Spectral Analysis of Unsteady Turbulence and Transition Effects in a Multistage Low Pressure Turbine," *Journal of Turbomachinery*, Vol. 141, No. 5, 2019. https://doi.org/10.1115/1.4041820.