Journal of Computational Physics 325 (2016) 22–37

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship

Jack Weatheritt *, Richard Sandberg

Department of Mechanical Engineering, University of Melbourne, Parkville, VIC, 3010, Australia

ARTICLE INFO

Article history:
Received 7 March 2016
Received in revised form 30 June 2016
Accepted 11 August 2016
Available online 16 August 2016

Keywords:
Tensor modeling
Evolutionary algorithms
Symbolic regression
Gene expression programming
RANS
Explicit algebraic stress modeling

ABSTRACT

This paper presents a novel and promising approach to turbulence model formulation, rather than putting forward a particular new model. Evolutionary computation has brought symbolic regression of scalar fields into the domain of algorithms and this paper describes a novel expansion of Gene Expression Programming for the purpose of tensor modeling. By utilizing high-fidelity data and uncertainty measures, mathematical models for tensors are created. The philosophy behind the framework is to give freedom to the algorithm to produce a constraint-free model; its own functional form that was not previously imposed. Turbulence modeling is the target application, specifically the improvement of separated flow prediction. Models are created by considering the anisotropy of the turbulent stress tensor and formulating non-linear constitutive stress–strain relationships. A previously unseen flow field is computed and compared to the baseline linear model and an established non-linear model of comparable complexity. The results are highly encouraging.

1. Introduction

In many areas of physics and engineering, solving a system of equations is computationally too expensive. Physics must be modeled through closure terms formulated from assumptions or *ad hoc* propositions. Modeling reduces the complexity of the problem and allows for solutions in a reasonable time, but inevitably introduces uncertainty and error. Tensor modeling is inescapable in multivariate problems. Tensors arise in physics to describe stress and metrics in differential geometry. The Reynolds-Averaged Navier–Stokes (RANS) equations are a classic example of a system which contains an unknown tensor. The turbulent or Reynolds stress τ_{ij} is an artifact of the Reynolds-averaging procedure and must be modeled.

Even with increasing computational power, modeling will remain relevant. Shortening engineering design cycles is of practical interest and therefore so are modeling frameworks [1,2]. Where possible, high fidelity data can facilitate the understanding of uncertainty and development of lower fidelity methods — extracting physics from data builds models based on actual not assumed phenomena. For the example of the Reynolds stress, high fidelity simulations/experiments of turbulence are expensive but possible [3–6] and these databases should be mined for model development.

Such a data-driven framework is proposed, specifically for training tensor models by the minimization of uncertainty measures. It is the desire to reduce the number of assumptions to produce constraint-free and tangible solutions. The framework produces a tensor model *a priori* from high fidelity data sets. The result is an algebraic equation that can be

* Corresponding author.
E-mail addresses: jack.weatheritt@unimelb.edu.au (J. Weatheritt), richard.sandberg@unimelb.edu.au (R. Sandberg).

http://dx.doi.org/10.1016/j.jcp.2016.08.015
0021-9991/

inserted into the system. In this sense the created models are 'implementation ready' and can be used as an *a posteriori* predictive tool. To achieve this, symbolic regression, that builds *model* not *data* approximations, is required. The former exists on a higher level of abstraction and allows us to probe the *how* of a process not just the *what*, yielding budgets and other quantitative information about model features. Evolutionary algorithms (EAs), that mimic nature's survival of the fittest — by iterating non-deterministically to optimize a problem, brought symbolic regression into the computer domain [7]. This paper describes such an algorithm; a novel implementation of Gene Expression Programming (GEP) [8] suitable for tensor regression.

In this proof of concept study, we use a single uncertainty measure to create candidate models by optimizing a baseline. These are then used as *a posteriori* predictive tools and compared to the baseline, plus an existing model of comparable complexity. The framework is considered a success should the training objective be improved over the baseline prediction and if this improvement is monotonic. In other words, if there is evidence that *a priori* training has a quantifiable effect in *a posteriori* testing. We will assume this statement as our null hypothesis to explore the efficacy of the new framework. This study, on a real world problem, is then used to provide evidence for this null hypothesis. Note, multi-objective optimization is possible, so models could be simultaneously trained using multiple uncertainty measures. However this study focuses on supporting our null hypothesis before considering more complex optimization. We show the results from a model of comparable complexity to highlight that improvements made over the baseline are not a formality.

The proposed framework is summarized as follows:

- Calculate the tensor of interest a_{ij} from the training data.
- Use GEP to create algebraic forms for this tensor based on scalar I^k and tensor V^k_{ij} variables, using the *a priori* form

$$a_{ij} = a_{ij}(V^1_{ij}, V^2_{ij}, \dots, I^1, I^2, \dots). \tag{1}$$

- Insert the new algebraic form into a baseline model.
- Make observations about the model by testing its predictive capability *a posteriori*.

The improvement of the Reynolds stress is chosen as the target application. Motivation of this problem domain with a summary of current uncertainty measures and utilization of high fidelity turbulence data is provided in Section 2. As the tensorial GEP method, and in particular its application to RANS model development is entirely novel, a detailed introduction to GEP is given in Section 3.1. The extension to tensor regression is described to a general audience in Section 3.2. The turbulence modeling application methodology is reported in Section 4. Firstly, the baseline model is introduced in Section 4.1, then the *a priori* optimization setup in Section 4.2 and finally the *a posteriori* test cases are described in Section 4.3. The testing of the null hypothesis is presented in Section 5. Finally, conclusions are discussed in Section 6.

2. Applied problem domain motivation and definition

Turbulence modeling is a discipline with developed techniques that utilize high fidelity data to inform about low fidelity methods. However, RANS model uncertainty remains poorly understood [9]. This is especially true for separated flows; the periodic hills test case is a good example and the subject of several workshops [10,11]. The majority of submitted results, regardless of model complexity, predicted the reverse flow poorly. In particular, models of the same class relative to the reference simulations [12–14], displayed high variability in solution quality.

In an effort to understand model uncertainty, databases can be used as a road-map [15], that gives turbulence modelers an idea of what kind of closure is applicable *a priori*. Alternatively, Bayesian calibration has been applied [16], which provides an *a posteriori* measure of model coefficient uncertainty. In a similar vein, inverse modeling [17] is a way of quantifying uncertainty by taking high fidelity data and inferring an appropriate adjustment to a RANS model. Further, current machine learning classification algorithms have been tasked with developing markers that identify areas of inaccuracy [9].

A contrasting approach is to insert perturbations into a RANS solution and measure the propagation of identified causes of uncertainty [18,19]. Recently, the RANS equations have been solved in a mean DNS flow field [20], which is a resurfacing of an earlier technique [21], to passively assess the turbulence model. Further, budgets can be analyzed to identify specific terms that provide the greatest source of error [22].

The use of high fidelity data is not limited to assessing model validity. Simulation and experimental databases have long been used to calibrate model coefficients. More recently however, complex machine learning tools are being used to improve model prediction. Model terms have been *replaced* with an Artificial Neural Network (ANN) [23]. An ANN is the ultimate black box, it consists of an input, output and a host of highly interconnected nodes [24]. An ANN was trained on existing RANS data beforehand and then its ability to reproduce RANS solutions was tested. Gaussian Process regression has been applied to improve model prediction in a turbulent channel [25]. This method interpolates between training data to infer a functional form for the model deficiency.

EAs are another viable approach. A Genetic Algorithm was used to recalibrate the coefficients of RANS models [26], whilst GEP has been tasked with formulating hybrid RANS/LES closures [27,28]. In the latter studies, the outputs were tangible mathematical equations that could be easily incorporated into CFD codes.

The majority of RANS models rely on a linear constitutive relationship, known as the Boussinesq approximation. This relates the anisotropy of the turbulent stress,

$$a_{ij} = \tau_{ij} - \tau_{kk}\delta_{ij}/3, \tag{2}$$

to the mean strain rate S_{ij} with a proportionality variable ν_t, known as the turbulent viscosity — because of its analogy to viscous diffusion. This analogy is often violated; the alignment between the anisotropy and strain tensors in several high fidelity databases shows that, even for canonical flows, this relationship breaks down [29].

Given that the Reynolds stress is a large source of error and that the constitutive stress–strain relationship is such a fundamental part of current RANS models, the anisotropy of the Reynolds stress makes an ideal candidate for the proposed framework. The alignment [29] of a_{ij} to reference data is used as the objective. Because of the known deficiencies of RANS in separated flow and the prevalence of such flow types, this is chosen as the target area of improvement: it is a real goal.

The *a priori* training uses the DNS from a backward facing step flow [30], a classic separated flow case. The resulting model is then validated *a posteriori* using the backward facing step and the periodic hills geometries. More emphasis is on the latter because it is difficult for RANS but also because it is different to the training case. Therefore model generalization is being tested. Because the algorithm is non-deterministic, repeat runs of the optimization do not produce the same model each time. This means there is a class of models attributed to the training data and fixed parameters of the algorithm. We can then infer statistical quantities by taking a sample of this class. This sample is obtained from multiple runs of the GEP process. Then by applying the sample to the *a posteriori* cases, we can provide evidence that supports our null hypothesis.

3. Evolutionary concepts

3.1. Gene expression programming

GEP is a branch of Evolutionary computing closely related to Genetic Programming (GP) [7]. The philosophy revolves around iteratively improving a population of candidate solutions by survival of the fittest. As with biology, GEP relies on random changes to the genome and is thus a non-deterministic process. GEP is a tool with a broad application range, although here is used and introduced as a symbolic regression tool that produces constraint-free solutions. This allows the same process to be applied to varied complex problems for which one cannot *a priori* fit a functional form.

As an introduction, assume we wish to find the function $f = f(x, y)$, for which we have a set of n data points (f_k, x_k, y_k) (where $k = 1, \ldots, n$) known as the training data. Our objective is to search the space of all functions for the one that 'best' fits this data set. In order to test this, one could define the fitness of a candidate solution f^{guess} as

$$\text{Fit}(f^{\text{guess}}) = 1 - \frac{1}{n}\sum_{k=1}^{n} \frac{||f^{\text{guess}}(x_k, y_k) - f_k||}{||f_k||} \tag{3}$$

which utilizes the relative distance from each data point and defines the optimal fitness as 1.

The second major component required to search the space of all functions is a compact notation that allows for rich expressions, but also simple modifications. In GEP, a candidate solution (also known as an individual or chromosome) is encoded as a linear string consisting of a head of fixed length h and a tail of length $t = h(n_a - 1) + 1$ where n_a is the maximum number of arguments a mathematical operator takes. The head region may consist of mathematical operators $(+, -, \times, \text{sin}, \text{exp}, \ldots)$ known as the functional set and variables (x, y) and constants (random or predefined) collectively known as the terminal set. The tail may only consist of symbols from the terminal set. Functions take a predefined number of arguments. For example, $+$ and $-$ both take two, whilst sin and exp only take one. Terminal symbols do not take any arguments. For example consider the individual with $h = 6$,

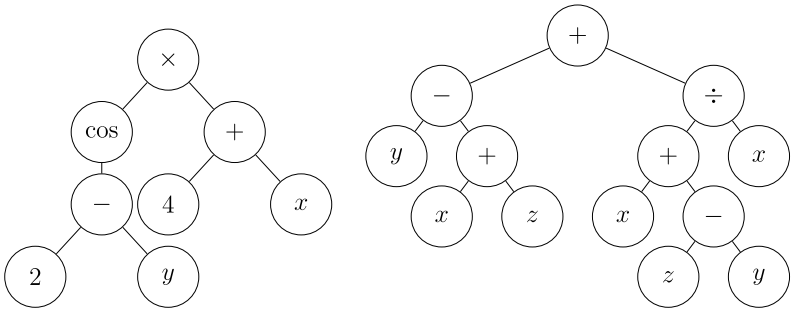
$$\texttt{*c-2y+}\,|\,\texttt{4xyyyx1}$$

where | denotes the separation between head and tail.

By reading recursively, left to right, one can write what is called the individual's Expression Tree (ET) shown in Fig. 1(a). Note that c translates to cos. The equation,

$$f^{\text{guess}}(x, y) = \cos(2 - y)(x + 4) \tag{4}$$

can then be read from the ET. Terminals exist only at the extremities of this tree, where as functions cannot be leaf nodes. This implies a minimum number of terminals for a given number of functions. Therefore, the inclusion of the tail in GEP individuals guarantees mathematical syntax and is the primary difference from GP. This reduces significantly the overheads of discarding invalid solutions. Any mathematical equation can be expressed, provided the mathematical operators are included in the allowed set. The simplicity of representation (linear string, called the genotype) against the complexity of expression (ET, called the phenotype) allows for simple modifications to the functional form of f^{guess} and does not limit the type of function obtainable by GEP, thus producing constraint-free solutions.



(a) Example ET corresponding to Eq. 4.

(b) Example multigenic ET.

Fig. 1. Example ETs.

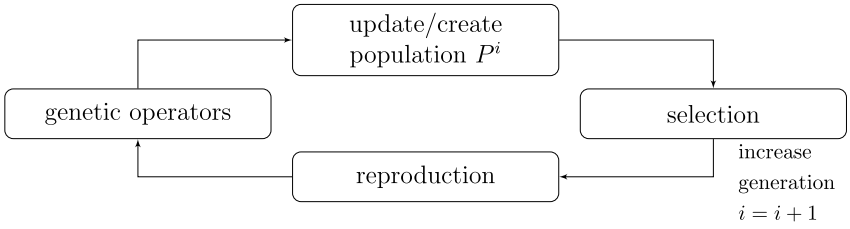


Fig. 2. Flow chart of GEP algorithm.

GEP chromosomes usually consist of multiple genes of equal length. For example the individual

```
-y+x|zxxyx <+> /+x-|zyxyy
```

which consists of two genes and linked by the + operation is expressed in Fig. 1(b). The equation deduced from Fig. 1(b) is,

$$(y-(x+z))+\frac{x+(z-y)}{x} \tag{5}$$

Multigenics allows for many reading points in the same individual, thus promoting diversity of expression.

Constants can be predefined if there is an intuitive reason, for example π can be included for fitting periodic functions. In general however, the user cannot specify the perfect set of constants available to GEP individuals. Therefore Random Numerical Constants (RNCs) can be introduced into the chromosome. This involves another section to the chromosome of length t which are indices that point to an array of RNCs defined between two preset numbers $r_{\min}, r_{\max}$. The `?` symbol is included in the terminal set and when one is expressed it accesses the next unused RNC. For example, say the RNC array is predefined as $(0.01, 10.4, -5.2, 0.99)$, then the individual,

```
*-x?+|?1?y42|323213
```

would be expressed as

$$f^{\text{guess}}(x,y) = (x-(-5.2))(10.4+1) \tag{6}$$

because the 3rd and 2nd indices were required.

In order to efficiently traverse the space of all functions, an evolutionary analogy directs the search. The GEP algorithm is schematically shown in Fig. 2.

A population P^i of N individuals (candidate solutions) is randomly created at generation $i = 0$ from the allowed list of mathematical symbols. Selection, or survival of the fittest, acts as a filter on P^i by weighting the chance of survival in favor of more fit solutions, directing the search away from poorer functional forms. This is achieved via tournament selection. This involves playing $N-1$ tournaments, each for one of $N-1$ spaces in generation $i+1$. A tournament consists of s individuals, chosen with replacement. The individual with the highest fitness (Eq. (3)) wins the tournament and fills the space for generation $i+1$. The final space in the next generation is always taken by the overall fittest individual, thus ensuring the best solution does not get removed if by chance it is not selected for a tournament.

The genetic operator step provides the search aspect of the algorithm. By modifying members of the population, variations on existing solutions are explored. Poor variants are then filtered out in the selection step. By varying s and the probability of genetic modification to an individual, a balance between variation and convergence of the population can be achieved. With not enough variation, locally optimal solutions propagate through the population in early generations and cause convergence (halting the incremental rise in the fitness), whilst too much variation does not allow evolution to ever start and the algorithm reduces to a random search. In general, the specification of s is problem and population size specific.

Such operations can be varied in effect and as long as the tail does not contain function symbols, these can be defined freely. The most common are borrowed from GP, namely mutation and recombination. The recombination operator occurs in the reproduction step and requires two individuals and a random integer $0 \leq c \leq h + t$. c acts as a cutting point in individuals, after which all information is swapped. The resulting new individuals are called the offspring and replace their parents in P^{i+1}. For example, the two individuals with $c = 3$

```
-y+ex|xy1yxy
+x/ys|xxx1yy
```

produce the offspring,

```
-y+ys|xxx1yy
+x/ex|xy1yxy
```

Mutation requires only one individual and is a 'copy with error' operator. If chosen for mutation, one symbol in an individual is randomly changed into another (ensuring that the tail cannot contain functions).

Other operators exist exclusively to the GEP framework that: invert, insert or translate a packet of symbols in an individual. These packets are subsets of genes but special versions of each operator exist that act on an entire gene. Genetic operators act on the linear string so aredeleted duplicated 'are' (Minor point 5.) simple to implement, always guarantee syntax and have large impacts on the functional form. As these operators provide bigger jumps in the search space, they should only occur very rarely to avoid too much variation between generations. Typically, each operation occurs with a probability defined in the range $(1/N, 1/h)$ with the exception that recombination occurs much more frequently ($p \sim 0.7$). A full description of these operators can be found in the original description of the algorithm [8].

3.2. Multidimensional gene expression programming

The standard GEP algorithm, outlined in Section 3.1, is an excellent scalar field regression tool. However, for problems of tensor regression (here we only consider Cartesian tensors) it easily produces 'dimensionally invalid' solutions. In this section, the aim is to fit a functional form $T_{ij} = T_{ij}(A_{ij}, B_{ij}, x, y) \in \mathbb{R}^{3\times 3}$ to the data set $(T_{ij,k}, A_{ij,k}, B_{ij,k}, x_k, y_k)$ where $k = 1, \ldots, n$. The chromosome (the tail is currently dropped without loss of argument),

```
++Ax*By
```

which is expressed as,

$$T_{ij}^{\text{guess}} = xA_{ij} + yB_{ij} \tag{7}$$

could undergo a mutation $\texttt{B} \rightarrow \texttt{y}$. This would violate the dimensionality of the candidate solution T_{ij}^{guess} and an overhead would be incurred to fix the issue. This problem becomes the norm and not the exception when one increases the size of the individual and includes more complex genetic operators.

There are many possible implementations of tensor regression. For example, one may write down a basis set V_{ij}^n and optimize each coefficient α_n,

$$T_{ij}^{\text{guess}} = \sum_n \alpha_n V_{ij}^n. \tag{8}$$

This is a quick solution, however unsatisfactory for a number of reasons. Should the basis set V_{ij}^n be reasonably large, then the resulting expressions for T_{ij}^{guess} would be clunky and trickier to evolve due to the high probability of expressing detrimental code. Such a basis set may also not be known. Further, consider that an EA is a natural feature extractor. By fixing the basis and optimizing parameters individually, one loses the algorithm's ability to join expressions together that cross bases. For example, assume that $f(x, y, z, \ldots)(V_{ij}^1 + V_{ij}^2)$ is a useful expression. For this to propagate through the population of a coefficient optimization, f must propagate twice (once for the V_{ij}^1 base and once for the V_{ij}^2). Instead a

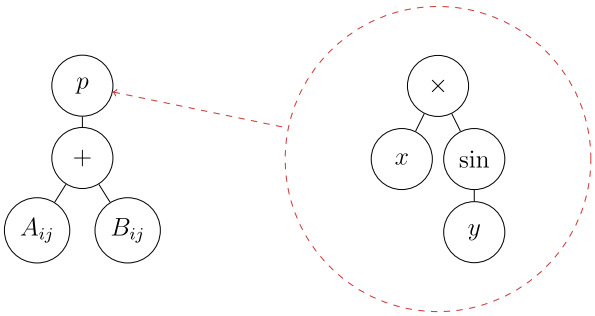


Fig. 3. Host chromosome and plasmid, p is the latching point, in this case at the root node.

more general implementation of tensor regression is now proposed; one that keeps EA modularity, ensures complex feature extraction and does not enforce constraints on the model form.

In order to alleviate this, the concept of a plasmid is introduced into the algorithm. A plasmid is a biological entity that exists within a cell, but can replicate independently of its host [31]. There is often a symbiotic relationship between host and plasmid, which increases the host fitness. The current algorithm involving plasmids is denoted Multidimensional Gene Expression Programming (M-GEP) for distinction and is a subtle yet powerful modification.

The 'host population' of individuals consist only of tensor symbols. Keeping with the example above, the function set could consist of the inner product, addition and subtraction and the terminal set could consist of A_{ij}, B_{ij}, δ_{ij}. The host population acts just as the population in the standard GEP framework and provides valid solutions to the regression problem. However they lack the ability to express the scalars x and y.

The plasmids are a separate population consisting of symbols that limits their phenotype to a scalar field. On their own they do not make sense, as the desired functional form is a tensor. They latch onto a host via the inclusion of a special `p` symbol in the host function set. This symbol acts as a request for a plasmid and multiplies itself by the argument of the `p` symbol. For example, consider the host and plasmid,

```
p+AB|AIBBI
*xsy1|22xy1y
```

which combine to give the ET in Fig. 3. They can then be translated into a single mathematical expression,

$$T_{ij}^{\text{guess}} = p(x, y) \times (A_{ij} + B_{ij}) = x \sin(y)(A_{ij} + B_{ij}). \tag{9}$$

Plasmids are set to have the same fitness as their hosts, as a scalar field cannot be measured against $T_{ij,k}$. This makes sense; if a host is considered fit, then any contained plasmids are contributing to this fitness. Therefore they deserve a fitness comparable to the host.

The fitness of a host T_{ij}^{guess} is calculated as (with Einstein notation over all indices except k)

$$\text{Fit}(T_{ij}^{\text{guess}}) = \sum_{k=1}^{n} \frac{T_{ij,k} T_{ij,k}^{\text{guess}}}{T_{mn,k} T_{nm,k} T_{pq,k}^{\text{guess}} T_{qp,k}^{\text{guess}}} \tag{10}$$

This checks the alignment between the training data and the candidate solution. The fitness function is bound between -1, complete misalignment, and 1, complete alignment, with 0 indicating complete orthogonality.

For each expressed `p` symbol in the host population, a new plasmid is created. Equally, when a host dies or stops expressing a `p` symbol, the given plasmid is also destroyed. A host individual is free to express multiple plasmids, with each `p` calling a distinct plasmid. When a host is copied so are the corresponding plasmids, thus creating multiple instances of the same scalar field.

The overall flow of the algorithm is very similar to Fig. 2, but with a few tweaks to allow co-evolution. Essentially the flow chart is traversed twice per generation — once for the hosts and once for the plasmids. First the host population undergoes selection, reproduction and mutation. This will cause plasmids to die and new plasmids to be created. Next, the plasmid population undergoes the same process outlined in Fig. 2. Each host individual keeps a reference to its plasmids to ensure that it receives its own, once the genetic operators have been applied to the plasmid population. This returned plasmid may be: unchanged, modified by mutation or be an entirely new plasmid obtained through selection. Note that during the plasmid tournament selection, should the selection of a new plasmid cause a drop in host fitness by 50%, then the tournament is undone and the original plasmid is retained. This helps to stop too much variation caused by rapidly changing plasmids.

4. Methodology

4.1. Baseline model

The RANS model chosen for modification is a slight variant of the SST [32]. The SST is known to over-predict the size of recirculation regions making it an ideal target for improvement. The model is very robust and easy to use; the result is ubiquity throughout CFD codes, such that modifications presented in this paper have a reasonable chance of being implementation ready. The baseline[1] model transport equations for turbulent kinetic energy k and specific dissipation rate ω are,

$$\partial_t k + U_j \partial_{x_j} k = P_k - \varepsilon + \partial_{x_j}\left[(\nu + \sigma_k \nu_t)\partial_{x_j} k\right] \tag{11}$$

$$\partial_t \omega + U_j \partial_{x_j} \omega = \gamma \frac{\omega}{k} P_k - \beta\omega^2 + \partial_{x_j}\left[(\nu + \sigma_\omega \nu_t)\partial_{x_j} \omega\right] + \sigma_d CD^+_{k\omega}, \tag{12}$$

where U_i is the Reynolds-averaged velocity. The Reynolds stress τ_{ij} is defined,

$$\tau_{ij} = \frac{2}{3}\delta_{ij}k - 2\nu_t S_{ij} + a^x_{ij}, \tag{13}$$

where a^x_{ij}, the extra anisotropy with respect to the linear Boussinesq approximation, is set to zero for the SST. This is the subject of modification via optimization. Auxiliary relations to close the transport equations are,

$$\begin{aligned} &CD_{k\omega} = \partial_{x_j} k \partial_{x_j} \omega, \quad CD^+_{k\omega} = \max\left[CD_{k\omega}, 0\right], \\ &\varepsilon = \beta^* k\omega, \qquad\qquad P_k = -\tau_{ij}\partial_{x_j} U_i, \\ &\nu_t = \frac{k}{\omega}. \end{aligned} \tag{14}$$

Note the eddy viscosity ν_t or production P_k are not limited as suggested by [33], because the goal is to assess the performance of M-GEP as a tool for model improvement. As a^x_{ij} is strongly linked to both, it is the opinion of the authors not to obfuscate findings by attempting to give a helping hand.

Finally the coefficients ψ of the model are blended using the function F_1, such that ψ_1 is active near the wall and ψ_2 in the far field,

$$\begin{aligned} \psi &= \psi_2 + F_1(\psi_1 - \psi_2), \qquad\qquad F_1 = \tanh\left(1.5\Gamma^4\right) \\ \Gamma &= \min\left[\max\left[\frac{k^{\frac{1}{2}}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right], \varphi\right], \quad \varphi = \frac{20k}{\max\left[y^2 CD_{k\omega}, 200k_\infty\right]}. \end{aligned} \tag{15}$$

The function F_1 is a modified version [34] of the original. The full set of coefficients are,

$$\begin{aligned} &\gamma_1 = 0.518, \quad \gamma_2 = 0.44, \quad \beta_1 = 0.0747, \quad \beta_2 = 0.0828, \\ &\sigma_k = 1.1, \quad \sigma_{\omega 1} = 0.53, \quad \sigma_{\omega 2} = 1.00, \quad \beta^* = 0.09, \\ &\sigma_{d1} = 1.0, \quad \sigma_{d2} = 0.4. \end{aligned} \tag{16}$$

4.2. A priori setup

Mean statistics from DNS wall-normal profiles are used to calculate the training data. The profiles are of τ_{ij}, $\partial_{x_j} U_i$, k and ε at the point of reattachment behind a backward facing step [35]. The wall-normal stress is prominent and a successful regression should pick up on the curvature effects. The inflow for this training case was laminar and therefore so was the subsequent separation. In order to train turbulence models, a fully turbulent profile is required because of the underlying assumptions of RANS closures. Therefore, in order to capture separation and maximize turbulence intensity, we use the location of reattachment. This results in a total of 100 data points. Reynolds number based on step height h is $\mathrm{Re}_h = 3000$. The case is ideal for DNS-RANS comparisons because ε – and by association ω – is well predicted by RANS, implying treatment of the data to make it consistent with RANS predictions is unnecessary [21].

The full anisotropy, Eq. (2), serves as the target variable. Resulting expressions then replace a^x_{ij} in Eq. (13) without coefficient calibration. Therefore a given a^x_{ij} model must survive by its own virtue and is not protected via manual modifications. This is the harshest way to test M-GEP and allows for complete automation of the process.

Fitting a^x_{ij} with a_{ij} is seen as the most practical method. S_{ij} is physically not aligned with a_{ij}, resulting in regression being overly sensitized to the strain and not the anisotropy of turbulence. Using S_{ij} in Eq. (13) is an artifact of two-equation models assuming a linear approximation in their formulation and therefore must be included in the modified Reynolds

[1] Baseline with respect to modifications to a^x_{ij}, it does not refer to the BSL-SST model.

stress. Failure to do so strongly increases the likelihood of divergence in *a posteriori* testing. However, in preliminary tests to confirm this statement, M-GEP often found models $a_{ij} = -C\nu_t S_{ij} + \cdots$ where C is constant.

Independent variables are defined assuming,

$$a_{ij} = a_{ij}(k, \varepsilon, S_{ij}, \Omega_{ij}), \tag{17}$$

where $2\Omega_{ij} = \partial_{x_j} U_i - \partial_{x_i} U_j$. This follows from classical analysis [36] that is a starting point for the derivation of Explicit Algebraic Stress Models (EASMs), of which many flavors are available [34,37–40].

Momentarily ignoring k and ε in Eq. (17), a linearly independent basis can be derived for a_{ij} through the Cayley–Hamilton theorem. For this study, the first four basis functions are used along with the two scalar invariants,

$$\begin{aligned} V^1_{ij} &= S_{ij} & V^2_{ij} &= S_{ik}\Omega_{kj} - \Omega_{ik}S_{kj} \\ V^3_{ij} &= S_{ik}S_{kj} - \tfrac{1}{3}\delta_{ij}S_{mn}S_{nm} & V^4_{ij} &= \Omega_{ik}\Omega_{kj} - \tfrac{1}{3}\delta_{ij}\Omega_{mn}\Omega_{nm} \\ I_1 &= S_{mn}S_{nm} & I_2 &= \Omega_{mn}\Omega_{nm}. \end{aligned} \tag{18}$$

To preserve the dimension of models for a^x_{ij}, all variables must be non-dimensionalized before being passed into M-GEP. The velocity gradient is normalized by $2\beta^* k/\varepsilon$. This coefficient is chosen because the optimization is solely based on Eq. (10). When re-multiplying through by k, the eddy viscosity falls out and M-GEP is at least given a chance of producing terms with the right magnitude.

V^n_{ij} and I_m are added to the host and plasmid terminal sets respectively. This tasks the M-GEP algorithm with producing EASM variants of the SST.

The specifics of the M-GEP algorithm parameters, such as h and N, can be found in Appendix A. The M-GEP algorithm is run 50 times with these parameters frozen. The best models from each optimization form a sample of the class of models associated with the described training data, fitness function and algorithm parameters.

4.3. A posteriori test cases

To validate the process and test the framework, the created models are applied to the backward facing step (BFS) and periodic hills (PH) flows. The first is the same geometry as the training data, but the second represents a previously unseen flow field that tests the predictive capability.

The BFS Reynolds number based on h and free stream velocity is $\mathrm{Re}_h = 5100$, which is a 40% increase over the training case. The aspect ratio of the inflow to outflow height is 5/6, with $x_{\mathrm{in}} = -10h$, $x_{\mathrm{out}} = 20h$ and the step itself placed at $x = 0$. The incoming boundary layer of $\mathrm{Re}_\theta = 670$ was taken from a precursor simulation using the SST. A total of 39000 grid points were used: 60 and 210 points before and after the step in the streamwise direction and in the wall normal direction, 68 points were placed above the step and 100 points in the expansion behind it. The DNS of Le, Moin and Kim [30] acts as reference data which we denote as DNS-LMK.

The PH case, is a flow over a series of constriction or 'hills' in a two dimensional channel [12–14]. The Reynolds number based on average streamwise velocity U_b from hill crest to upper surface is $\mathrm{Re}_h = 10595$. Streamwise periodic boundary conditions are imposed, with a pressure gradient to ensure the correct mass flow rate. The grid consists of 120×130 cells. The LES of Temmerman and Leschziner [12], denoted here as LES-TL, serves as reference data.

The models are implemented into OpenFOAM [41], an open-source finite volume code. The equations are discretized with linear upwinding for the divergence terms and 2nd order central differencing for the turbulent diffusion. For both cases, grid convergence and 'time' convergence were checked using an M-GEP non-linear model. The flow field predicted by the SST acted as initial conditions for the M-GEP models.

5. Results

5.1. Evidence of evolution

Fifty runs of the M-GEP algorithm produced fifty distinct equations for a^x_{ij} that have been tested on the PH and BFS. Of these 100 simulations only one diverged. The best model, based on metrics discussed in the Section 5.2, is

$$a^x_{ij} = 2\nu_t V^1_{ij}\Big[0.02 - 0.0648\tau_l^2\Big(3I_1 + I_2 + 0.0162\tau_l^2 I_1^2\Big)\Big] - 0.18\nu_t\tau_l\Big[V^2_{ij} + V^3_{ij} + 2V^4_{ij}\Big] \tag{19}$$

where τ_l is the turbulent time scale. Equation (19) has been simplified from its original phenotypic expression, but indicates the clear tangibility of results from the algorithm. Another expression resulting from the algorithm is given as,

$$a^x_{ij} = 2\nu_t V^1_{ij}\Big[0.01 - 0.0324\tau_l^2(3I_1 - I_2)\Big] - 0.18\nu_t\tau_l\Big[V^2_{ij} + V^3_{ij} + 2V^4_{ij}\Big]. \tag{20}$$

Again this equation has been manually simplified but one can immediately see the similarity to Eq. (19). Both of these functions have come from two separate runs of the algorithm, meaning distinct initial populations and random constants

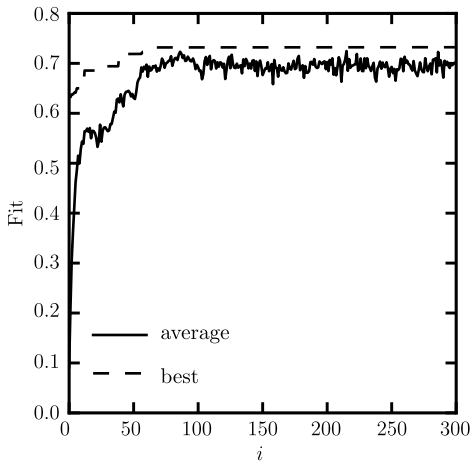


Fig. 4. Example of average and best fitness evolution within a single population.

have yielded two syntactically similar functions. This is surprising and pleasing as the non-deterministic algorithm has converged to functionally similar expressions. Both runs of the algorithm have found gains through modifying an expression attached to V_{ij}^1. The particle $-0.18\nu_t\tau_I\left[V_{ij}^2+V_{ij}^3+2V_{ij}^4\right]$ has been discovered in both runs of the algorithm and its expression in both Eq. (19) and Eq. (20) makes them close in the search space. This functional similarity is transferred into comparable predictive performance, which is addressed in Section 5.2.

To demonstrate the evolutionary analogy, Fig. 4 is a plot of average and best fitness in the population as a function of generation. This training run, for this small test problem, is on the order of core minutes. The data is taken from the first run of the algorithm. We can observe an early incremental rise in the population fitness, with high variation in the average as the algorithm explores the search space.

Finally, to further demonstrate the use of evolutionary search, the same problem was also attempted at random. Random search was only able to find as high as the median of the M-GEP sample, however it took three orders of magnitude more comparisons. Note, training using M-GEP is on the order of minutes, whilst the random search was left for several days to record anything greater than the minimum in the M-GEP sample.

5.2. Evaluation of performance metrics — on design

The 50 evolved algebraic expressions for a_{ij}^x can be tested in many ways. Most simply, they can be validated *a posteriori* by comparing profiles of flow quantities. They can also be tested *a priori* by using the fitness function, $\text{Fit}(a_{ij}^x)$, defined in Eq. (10). The natural extension of this is to define the comparative *a posteriori* measure, where ref corresponds to the relevant reference data, either LES-TL or DNS-LMK,

$$\rho_a = \rho_a(y) = \frac{a_{ij}a_{ji}^{\text{ref}}}{a_{mn}a_{nm}a_{pq}^{\text{ref}}a_{qp}^{\text{ref}}}, \tag{21}$$

at streamwise locations containing the point of reattachment for the BFS and PH cases. Such profiles, similar to the training data, are $x/h = 4.0$ and $x/h = 6.0$ for the PH and BFS respectively. These can then be integrated to give a single point measure for each model and each case $\rho_{a,\text{case}}$,

$$\rho_{a,\text{case}} = \int_y \rho_a(y)dy, \tag{22}$$

that can be compared directly with the *a priori* alignment calculated in Eq. (10). These metrics of performance, $\text{Fit}(a_{ij}^x)$ and $\rho_{a,\text{case}}$ give a direct relationship between how well the model was trained and how this training corresponds into predictive performance in 'on design' conditions, i.e. the most straightforward test of our null hypothesis. Classical validation by considering profiles of flow quantities is also discussed below, however these are in a sense 'surrogate' outcomes, as we were optimizing directly for anisotropy alignment and not the quality of these profiles. The focus is 'on design' conditions — the training should produce models that are fit for scenarios represented by the training data. That said, we briefly explore the 'off design' conditions of the PH geometry in Section 5.3.

Other metrics of model performance measure the ratio of the normal stresses,

$$\tau_r = \tau_{22}/\tau_{11} \tag{23}$$

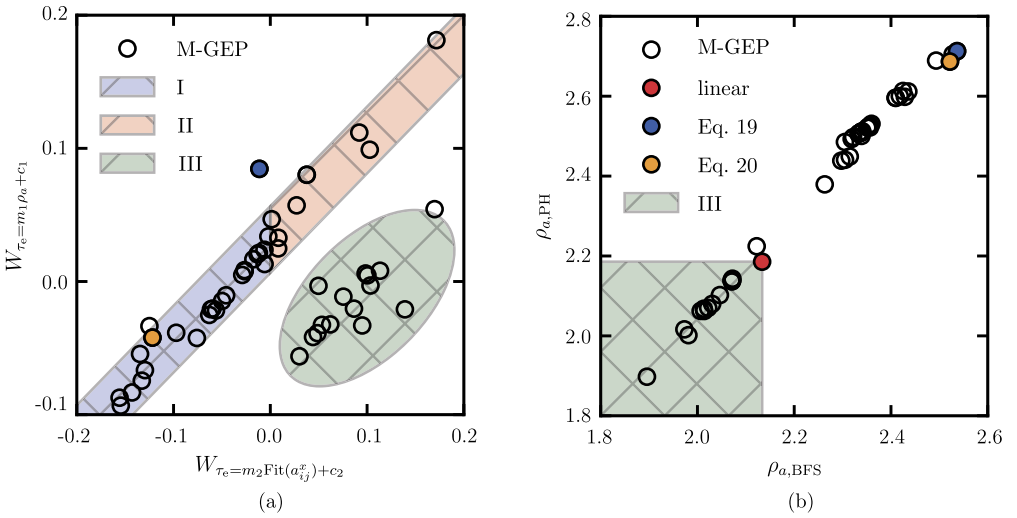


Fig. 5. Left: Residual–residual plot of the two linear regressions fitting magnitude error as a function of *a priori* fitness and *a posteriori* alignment. Right: Relationship between PH and BFS case alignment.

and the error in Reynolds stress magnitude in a wall normal profile,

$$\tau_e = \int_y \frac{(\tau_{ij} - \tau_{ij}^{\mathrm{ref}})(\tau_{ij} - \tau_{ij}^{\mathrm{ref}})}{|\tau_{kk}^{\mathrm{ref}}|} dy \tag{24}$$

τ_e is a measure of the relative error, which was not controlled in the M-GEP optimization. Therefore it is possible, in the effort to maximize $\mathrm{Fit}(a_{ij}^x)$, that the algorithm produced models such that $a_{ij}^x \approx C \cdot a_{ij}$ where C is a very large or very small constant. τ_e is a confounding variable and must be considered in the analysis. To calculate the partial correlation between $\mathrm{Fit}(a_{ij}^x)$ and $\rho_{a,\mathrm{PH}}$, first the following linear regressions are performed,

$$\tau_e = m_1 \cdot \rho_{a,\mathrm{PH}} + c_1 \tag{25}$$

$$\tau_e = m_2 \cdot \mathrm{Fit}(a_{ij}^x) + c_2. \tag{26}$$

The corresponding residuals,

$$W_{\tau_e=m_1\rho_{a,\mathrm{PH}}+c_1} = \widehat{\tau_e}\Big|_{\rho_{a,\mathrm{PH}}} - \tau_e \tag{27}$$

and

$$W_{\tau_e=m_2\mathrm{Fit}(a_{ij}^x)+c_2} = \widehat{\tau_e}\Big|_{\mathrm{Fit}(a_{ij}^x)} - \tau_e \tag{28}$$

are plotted in Fig. 5(a) and the linear regression between those provides the partial correlation r between *a priori* and *a posteriori* environments. In Fig. 5(b), there are three identified regions. The first, in which $r = 0.93$, provides very strong evidence that *a priori* training directly relates to *a posteriori* performance. This relationship is also continued in region II. In region III however, we see a large departure from the desired trend. When $W_{\tau_e=m_2\mathrm{Fit}(a_{ij}^x)+c_2} > 0$, the magnitude error is bigger than expected and the single-objective nature of the study has been detrimental to performance. Region II therefore consists of models that have luckily retained a higher than expected *a posteriori* alignment. In practical applications of the methodology, it is clear that multi-objective optimization is required.

In this analysis, $\rho_{a,\mathrm{PH}}$ was used as the *a posteriori* measure of anisotropy alignment. However, in Fig. 5(b), a plot of Eq. (22) for each model shows that performance transfers between cases with very low uncertainty. The red data point is $\rho_{a,\mathrm{case}}$ evaluated using the linear SST. The models contained within the bounding square are those that performed worse than the baseline and are precisely those contained inside region III of Fig. 5(a). Fig. 5(b) shows that Eq. (19) and Eq. (20) have a very similar performance level. Considering Fig. 5(a), Eq. (19) is revealed to be an outlier — it is much better *a posteriori* than its *a priori* fitness suggests. This is another effect of the confounding variable τ_e.

From these plots, the conclusion is that if the magnitude has been sacrificed during the optimization process, then the model has performed worse than the baseline. If this hasn't been the case, then the models have been successfully sensitized to the training data and crucially the more effectively this happened, the better the *a posteriori* prediction.

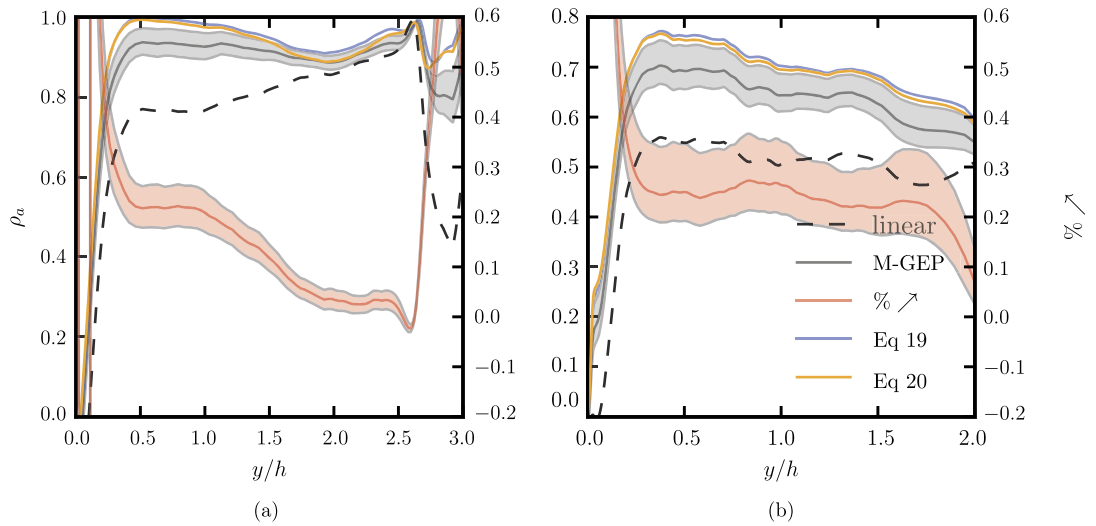


Fig. 6. *A posteriori* alignment as a function of y for each test location. M-GEP is represented as the 99.9% confidence interval for the class mean. Red is 99% confidence for the rate of increase over linear. Left: PH, Right: BFS. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

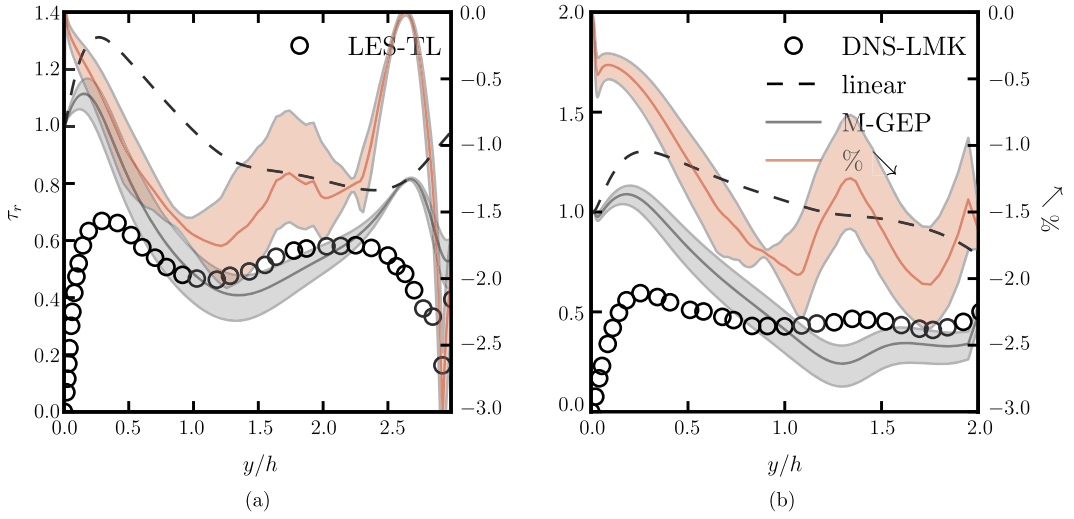
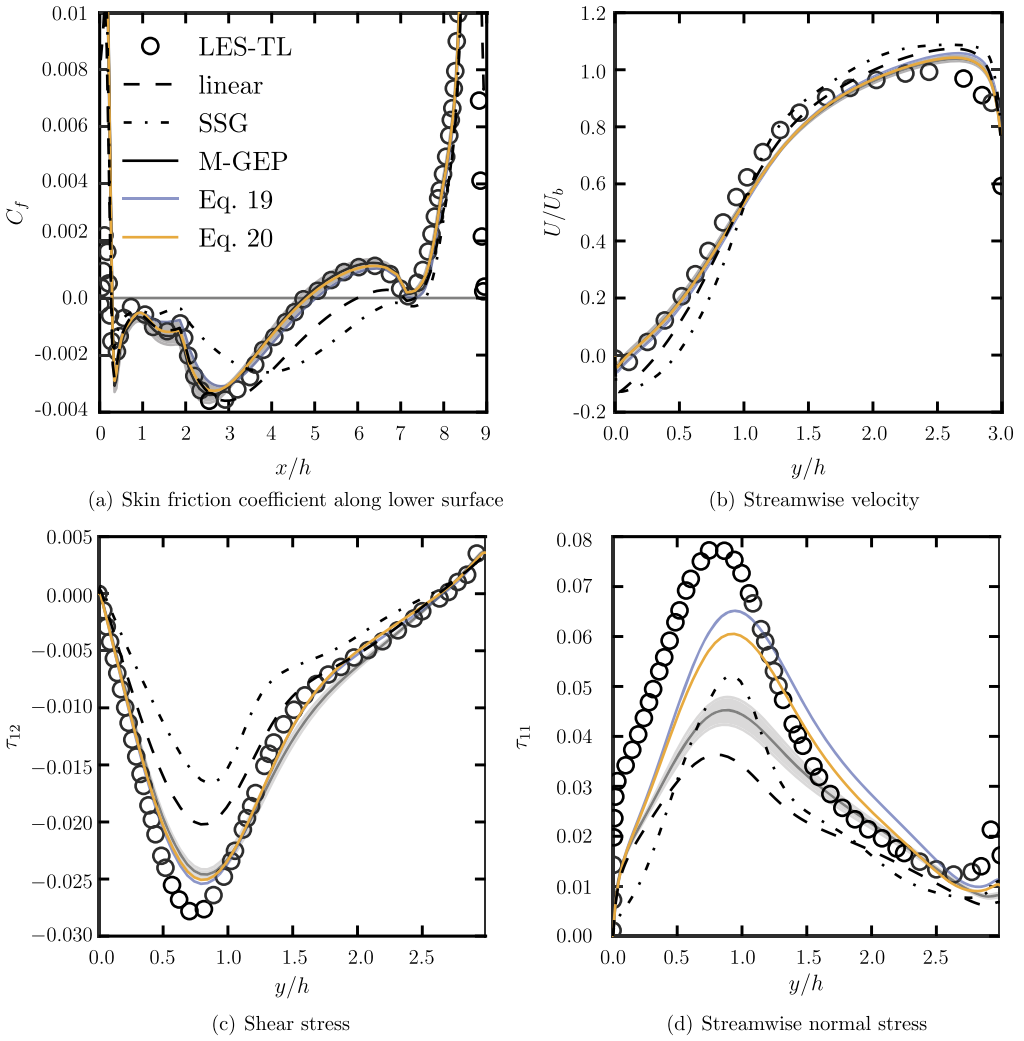


Fig. 7. *A posteriori* ratio of normal stresses as a function of y for each test location. M-GEP is represented as the 99.9% confidence interval for the class mean. Red is 99% confidence for decrease in relative error over linear. Left: PH, Right: BFS. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In the remainder of this section, the region III M-GEP models are removed, although it should be noted that they are not exceptionally poor performers — Fig. 5(b) shows that they are closer in performance to the baseline than the baseline is to the better models.

Fig. 6 plots Eq. (21) for both PH and BFS. The M-GEP sample is represented as a 99.9% confidence interval for the class mean. Also included is Eq. (19), Eq. (20) and the baseline. The red band is a 99% confidence interval for the relative increase over the linear model. The M-GEP mean alignment in the PH case is at least as good as linear for all y, with only momentary convergence at the point $\tau_{12} = 0$. Near the lower wall, the scaling k/ε breaks down so both the linear and M-GEP models become misaligned with the reference. The structural similarity of Eq. (19) and Eq. (20) has produced two highly similar ρ_a profiles, with only a slight deviation in the bulk regions.

A 99% confidence interval for the percentage increase in alignment over the linear model is also presented. The improvement near the wall is flattering, as the absolute value of $\rho_a(y)$ is small. In general, across the reverse flow region and shear layer, there is a reasonably consistent 20% improvement over the baseline. This rate of improvement as already noted reduces to zero in Fig. 6(a), because of the presence of two walls and therefore the necessary point of zero velocity gradient.



(a) Skin friction coefficient along lower surface

(b) Streamwise velocity

(c) Shear stress

(d) Streamwise normal stress

Fig. 8. Statistical results from the PH case. Profile for stress and velocity is $x/h = 4.0$. The M-GEP sample is represented as a 99.9% confidence interval for the mean.

By the same scaling argument as used above, $\tau_r \to 1$ not 0 when $y \to 0$ as observed in Fig. 7. This means that the algorithm is not successfully sensitizing the model to the near wall anisotropy with the time scale τ_l and motivates future near wall specific studies using representative time and length scaling of the independent variables. We can also observe initial incorrect growth of the ratio of the normal stresses in the region $y/h < 0.25$ as $\tau_r > 1$. However, for both the PH and BFS cases, the M-GEP sample τ_r quickly reduces to a value more representative of the reference data but the linear model never recovers. The relative decrease in error of τ_r is also included in Fig. 7. For both cases, the improvement in the shear layer is approximately 150%, although the width of this confidence interval grows in the bulk of the PH channel.

In terms of testing for the null hypothesis, defined in Section 1, there is compelling evidence that the *a priori* training is having an effect *a posteriori*. From the identified metrics, that have relevance to the fitness function, the M-GEP sample significantly outperforms the linear model. In particular, as highlighted, the partial correlation between the two measures $\mathrm{Fit}(a_{ij}^x)$ and $\rho_{a,\mathrm{PH}}$ for models which were not trained with a non-detrimental magnitude (region I models in Fig. 5(a)) is 0.93. The p-value for this statistic is 1×10^{-11}, which on any sensible statistical level would not cause us to reject the notion of a link between training and predictive performance.

5.3. Flow quantities for the PH case

In this section, the prediction of flow quantities by the M-GEP sample is compared against the linear baseline and also the established SSG EASM model [37]. The SSG model is included to show that the addition of non-linear terms does not

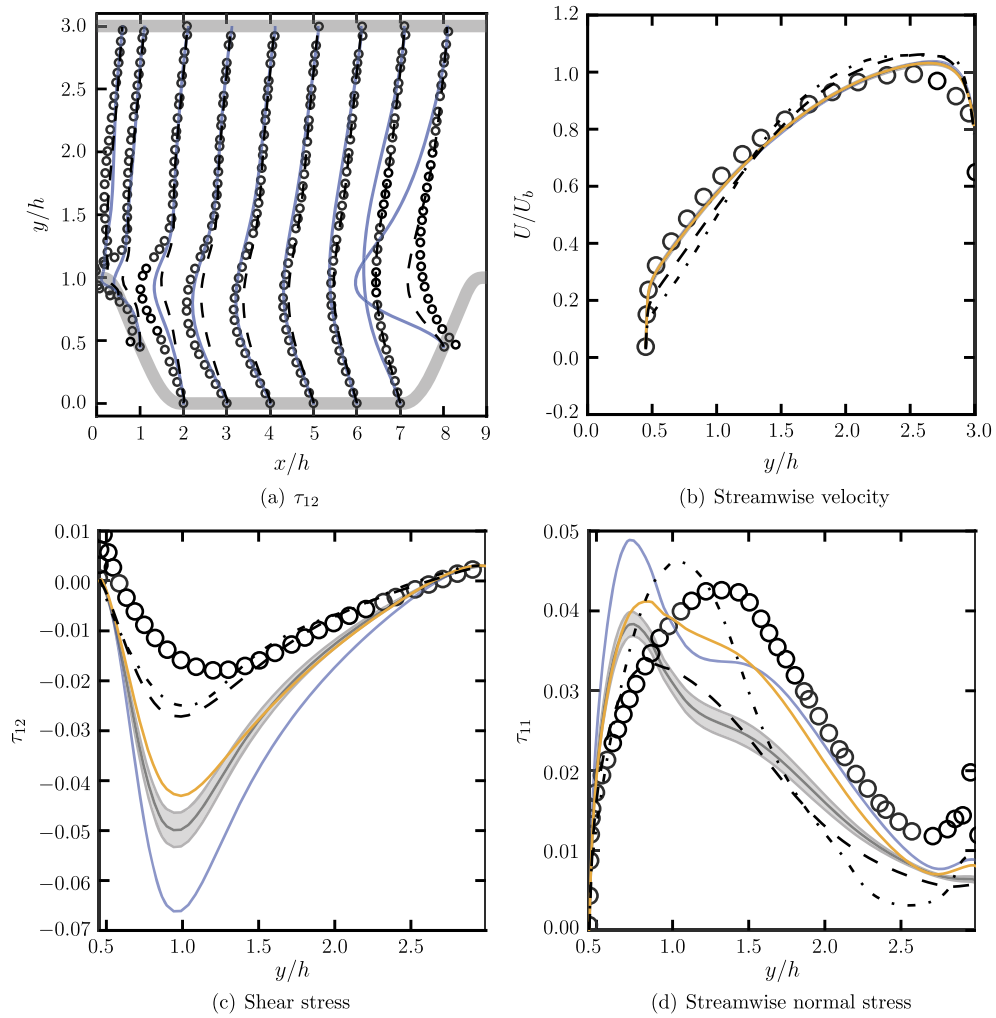


Fig. 9. Flow quantities in 'off-design' conditions. (a) highlights on-design vs off-design via Eq. (19) compared against baseline. (b)–(d) show statistical results at the streamwise location $x/h = 8.0$. Colors as in Fig. 8.

necessarily equate to improved performance. We focus on the PH case, as this is different from the original training data and raises important points of discussion.

Fig. 8(a) is a plot of skin friction coefficient along the lower surface and it is evident that the SSG model, which performs worse than the linear model, does not correctly predict flow reattachment, rather relying on the base of the upwind side of the hill to correct the flow field. In contrast, the M-GEP sample excellently predicts the wall shear stress in the reverse flow region.

The velocity and shear stress profiles in 'on design' conditions, Fig. 8(b)–(c), show remarkably small confidence intervals for the M-GEP mean, which is once again significantly better than that of the linear and SSG models. In Fig. 8(d), the confidence interval is much wider for τ_{11} in the shear layer. This is because in a shear flow the pressure-strain correlation drains from the streamwise normal stress and it is precisely this term being modeled. Hence we can expect bigger variations from model to model. Equation (19) and Eq. (20) are included in Fig. 8 to show their agreement with each other.

Fig. 8(a) shows that the C_f predicted by the M-GEP models closely matches the baseline on the upwind side of the hill. There is no reason to expect the trained models to perform well in this region as the training data is markedly different. In the region $7 \leq x/h \leq 9$, there is a strong favorable pressure gradient, where as the training data is from an adverse pressure gradient flow region. We can see from Fig. 9(a) that the negative shear stress at the location $x/h = 8$ is much too large. This plot contains just the linear and Eq. (19) models as comparison to the LES. Note, Eq. (19) is actually the worst model at this location as it is the best trained to adverse pressure gradient flow. Fig. 9(b)–(d) show the same statistical quantities as Fig. 8(b)–(d). The velocity prediction between all M-GEP shows this region is strongly dominated by the geometry and not the turbulence model, hence global C_f is not affected. The Reynolds stress, particularly the shear stress, is overpredicted. Interestingly Eq. (19) and Eq. (20) are markedly different for both τ_{11} and τ_{12}. This shows that despite similar prediction and similar functional form, there is a high level of uncertainty in model performance in 'off design' conditions.

Because the models do not respond well to the favorable pressure gradient should not count against the methodology; in fact quite the opposite. As the models perform well in the adverse pressure gradient region and poorly in the favorable; we can further conclude that the training has created model terms in an appropriate way. We further reiterate the proof of concept nature of this study and that the current models are not fit for purpose. However, in future studies when the training data is expanded, there is confidence that applicable RANS models will be produced.

6. Conclusions

This paper has presented a novel implementation of GEP that is suitable for regressing tensor expressions. Tensor models have been formed by maximizing an uncertainty measure from the literature. The models were then used as *a posteriori* predictive tools and found to be very effective on a similar, yet different, problem to their training data. A link between *a priori* training and *a posteriori* prediction was successfully established in this single-objective, proof of concept study. This demonstration was the main purpose of this paper, not the specifics of the presented RANS models. Future studies can now build on this result to begin to look at more sophisticated techniques for high fidelity data preparation.

The philosophy of the new framework is to allow freedom to search for the global optima (or at least acceptable local optima) and as a consequence the main focus has been on expanding the notions of freedom of expression from the standard GEP to enable constraint-free tensor regression. Note that local optima is emphasized, the space of all functions is infinite and the truncation of the phenotype, by specifying a maximum genotype length, almost certainly makes the discovery of the global optima impossible.

This approach of using uncertainty measures to drive model formulation should now be expanded. The modularity of EAs allows for diversity in objective functions. The application of GEP to symbolically regress RANS model components is also entirely novel. In this regard we have also tested the feasibility of using ρ_a as an objective for RANS model formulation. This can be swapped for L^p norms or more complex measures discussed in Section 2. These objectives can also be simultaneously optimized in a multi-objective framework.

The evolutionary system is aggregating towards solutions that are useful predictive tools to an engineer. Whilst again it should be reiterated that this is only one specialized case, the potential for use in a design loop using high fidelity data to feed RANS models which then in turn feed the direction of future high fidelity tests is apparent. As a consequence, it would be a natural progression to test on complex geometries. The poor performance in the favorable pressure gradient region during *a posteriori* testing shows that models trained on one flow regime should not activate in another. Multiple data sets can be used to alleviate this issue. Intelligent choice of independent variables is then required, say by expanding Eq. (18) to include damping functions sensitive to pressure gradients and rotation. There always remains the option of using different sets independently and either applying these damping functions to the respectively trained models,

$$a_{ij}^{x} = f^{\mathrm{rot}} a_{ij}^{\mathrm{rot}} + f^{\mathrm{pg}} a_{ij}^{\mathrm{pg}} + \cdots, \tag{29}$$

or using the fitness values as a model weighting during runtime when a specific flow regime is detected. Of specific interest is the near-wall region and the scaling issue of the current choices of V_{ij}^{n} and I_m. With the current non-dimensionalization using $\tau = k/\varepsilon$, the near-wall region reduces to a linear model because as $y \to 0$, $\tau \to 0$ and so, by consequence, does a_{ij}^{x}. Therefore a specific near-wall scaling is required, which may be activated via functions from the SST — see Section 4.1

Beyond the current application, the concept of plasmids is more than just a minor tweak and allows for many previously inaccessible optimization problems. Hard to visualize rank three tensors could be regressed using host individuals of rank three, with a plasmid population of rank two operations and another plasmid of rank one operations. Beyond regression, the inclusion of plasmids is a method for allowing complete freedom of expression in trees that have more than one type of node. This idea can be extended to complex decision trees where the plasmid itself is a lower level decision. Such an example would be adaptive mesh refinement where the plasmid denotes an expression for the sub-tree mesh expansion ratio or some other scalar quantity.

Acknowledgements

This work is funded in part by veski and in part by an EPSRC Doctoral Training Centre grant (EP/G03690X/1).

Appendix A. M-GEP parameters

Table A.1 is a list of the parameters for the current study. Where appropriate, the symbol from the main text has been included. The list of available variables has been kept small and potential difficulties arising from complex arguments is negated. Further, the maximum size of a host expression tree is not particularly large at 23. This provides quite a sharp truncation of the phenotype and a bound on the functional complexity. In practical examples, one would also include an objective that minimizes function length and tree depth. Note that $|r_{\min}| = |r_{\max}| \approx 2C_{\mu}$. This bound is chosen by the same reasoning used in the scaling of the velocity gradient, as discussed in Section 4.2. The study presented is robust to the choices of the parameters; perturbations of the space have not been presented.

Table A.1
Values of all parameters used in M-GEP optimization.

Parameter	Value	Notes
N	200	population size
	300	number of generations
	3	number of genes in host chromosome
	2	number of genes in plasmid chromosome
h	3	head length in plasmid and host gene
$r_{\min}$	-0.2	random constant minimum
$r_{\max}$	0.2	random constant maximum
	2	tournament size
	V^1_{ij}, V^2_{ij}, V^3_{ij}, V^4_{ij}	host terminal set
	I_1, I_2, ?, 1.0, 0.01	plasmid terminal set: ? is a random constant
	$+$, $-$, p	host chromosome function set: p is plasmid arity one symbol
	$+$, $-$, $\times$	plasmid function set

References

[1] S.B. Pope, A perspective on turbulence modeling, in: Modeling Complex Turbulent Flows, Springer, 1999, pp. 53–67.
[2] K. Hanjalić, Will RANS survive LES? A view of perspectives, J. Fluids Eng. 127 (5) (2005) 831–839.
[3] R. Vinuesa, A. Noorani, A. Lozano-Durán, G.K.E. Khoury, P. Schlatter, P.F. Fischer, H.M. Nagib, Aspect ratio effects in turbulent duct flows studied through direct numerical simulation, J. Turbul. 15 (10) (2014) 677–706.
[4] R. Vinuesa, P. Schlatter, J. Malm, C. Mavriplis, D.S. Henningson, Direct numerical simulation of the flow around a wall-mounted square cylinder under various inflow conditions, J. Turbul. 16 (6) (2015) 555–587.
[5] R. Vinuesa, S.M. Hosseini, A. Hanifi, D.S. Henningson, P. Schlatter, Direct numerical simulation of the flow around a wing section using high-order parallel spectral methods, in: Internation Symposium on Turbulence & Shear Flow Phenomena, TSFP-9, 2015, p. 30.
[6] V. Michelassi, L.-W. Chen, R. Pichler, R.D. Sandberg, Compressible direct numerical simulation of low-pressure turbines—part II: effect of inflow disturbances, J. Turbomach. 137 (7) (2015) 071005.
[7] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, 1st edition, MIT Press, 1992.
[8] C. Ferreira, Gene expression programming: a new adaptive algorithm for solving problems, Complex Syst. 13 (2) (2001) 87–129.
[9] J. Ling, J. Templeton, Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty, Phys. Fluids 27 (8) (2015) 085103.
[10] S. Jakirlić, R. Jester-Zürker, C. Tropea, Report on the 9th ERCOFTAC/IAHR/COST workshop on refined turbulence modelling, in: ERCOFTAC Bulletin, vol. 55, 2001.
[11] R. Manceau, Report of the 10th joint ERCOFTAC (SIG-15)/IAHR/QNET-CFD workshop on refined turbulence modelling, in: ERCOFTAC Bulletin, vol. 57, 2003.
[12] L. Temmerman, M.A. Leschziner, Large eddy simulation of separated flow in a streamwise periodic channel constriction, in: 2nd Symposium on Turbulence and Shear-Flow Phenomena, 2001.
[13] J. Fröhlich, C.P. Mellen, W. Rodi, L. Temmerman, M.A. Leschziner, Highly resolved large-eddy simulation of separated flow in a channel with streamwise periodic constrictions, J. Fluid Mech. 526 (2005) 19–66.
[14] M. Breuer, N. Peller, C. Rapp, M. Manhart, Flow over periodic hills—numerical and experimental study in a wide range of Reynolds numbers, Comput. Fluids 38 (2) (2009) 433–457.
[15] J. Hunt, A. Savill, Guidelines and criteria for the use of turbulence models in complex flows, in: Prediction of Turbulent Flows, 2005, pp. 291–343.
[16] W. Edeling, P. Cinnella, R.P. Dwight, Predictive RANS simulations via Bayesian model-scenario averaging, J. Comput. Phys. 275 (2014) 65–91.
[17] E. Dow, Q. Wang, Quantification of structural uncertainties in the k–ω turbulence model, in: 52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 2011, p. 1762.
[18] M. Emory, R. Pecnik, G. Iaccarino, Modeling structural uncertainties in Reynolds-averaged computations of shock/boundary layer interactions, in: AIAA Paper, vol. 479, 2011, p. 2011.
[19] C. Gorlé, G. Iaccarino, A framework for epistemic uncertainty quantification of turbulent scalar flux models for Reynolds-averaged Navier–Stokes simulations, Phys. Fluids 25 (5) (2013) 055105.
[20] P. Spalart, M. Shur, M.K. Strelets, A. Travin, Direct simulation and RANS modelling of a vortex generator flow, Flow Turbul. Combust. 95 (2–3) (2015) 335–350.
[21] S. Parneix, D. Laurence, P. Durbin, A procedure for using DNS databases, J. Fluids Eng. 120 (1) (1998) 40–47.
[22] R. Pichler, R.D. Sandberg, V. Michelassi, R. Bhaskaran, Investigation of the accuracy of RANS models to predict the flow through a low-pressure turbine, in: ASME Turbo Expo 2015: Turbine Technical Conference and Exposition, American Society of Mechanical Engineers, 2015, V02BT39A036.
[23] B. Tracey, K. Duraisamy, J.J. Alonso, A machine learning strategy to assist turbulence model development, in: 53rd AIAA Aerospace Sciences Meeting, AIAA SciTech, 2015, p. 2015-1287.
[24] S.S. Haykin, Neural Networks and Learning Machines, 3rd edition, Pearson, Prentice Hall, 2009.
[25] E.J. Parish, K. Duraisamy, A paradigm for data-driven predictive modeling using field inversion and machine learning, J. Comput. Phys. 305 (2016) 758–774.
[26] B. Fabritius, Application of genetic algorithms to problems in computational fluid dynamics, Ph.D. thesis, University of Exeter, 2014.
[27] J. Weatheritt, R.D. Sandberg, Use of symbolic regression for construction of Reynolds-stress damping functions for hybrid RANS/LES, in: 53rd AIAA Aerospace Sciences Meeting, 2015, p. 2015-0312.
[28] J. Weatheritt, R.D. Sandberg, A new Reynolds stress damping function for hybrid RANS/LES with an evolved functional form, in: Advances in Computation, Modeling and Control of Transitional and Turbulent Flows, World Scientific, 2016, pp. 330–339.
[29] F.G. Schmitt, About Boussinesq's turbulent viscosity hypothesis: historical remarks and a direct evaluation of its validity, C. R., Méc. 335 (9) (2007) 617–627.
[30] H. Le, P. Moin, J. Kim, Direct numerical simulation of turbulent flow over a backward-facing step, J. Fluid Mech. 330 (1997) 349–374.
[31] L. Margulis, R. Fester, Symbiosis as a Source of Evolutionary Innovation: Speciation and Morphogenesis, MIT Press, 1991.
[32] F.R. Menter, Two-equation eddy-viscosity turbulence models for engineering applications, AIAA J. 32 (8) (1994) 1598–1605.
[33] F.R. Menter, Zonal two equation k–ω turbulence models for aerodynamic flows, in: AIAA Paper, vol. 2906, 1993, p. 1993.
[34] A.K. Hellsten, New advanced k–ω turbulence model for high-lift aerodynamics, AIAA J. 43 (9) (2005) 1857–1869.

[35] J. Otero, University of Southampton, Private communication, 2015.
[36] S. Pope, A more general effective-viscosity hypothesis, J. Fluid Mech. 72 (02) (1975) 331–340.
[37] T. Gatski, C. Speziale, On explicit algebraic stress models for complex turbulent flows, J. Fluid Mech. 254 (1993) 59–78.
[38] T. Craft, H. Iacovides, J. Yoon, Progress in the use of non-linear two-equation models in the computation of convective heat-transfer in impinging and separated flows, Flow Turbul. Combust. 63 (1–4) (2000) 59–80.
[39] S. Wallin, A.V. Johansson, An explicit algebraic Reynolds stress model for incompressible and compressible turbulent flows, J. Fluid Mech. 403 (2000) 89–132.
[40] K. Abe, Y.-J. Jang, M.A. Leschziner, An investigation of wall-anisotropy expressions and length-scale equations for non-linear eddy-viscosity models, Int. J. Heat Fluid Flow 24 (2) (2003) 181–198.
[41] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. 12 (6) (1998) 620–631.