Int J Adv Manuf Technol (2014) 74:1–16
DOI 10.1007/s00170-014-5829-0

ORIGINAL ARTICLE

The prediction of profile deviations when Creep Feed grinding complex geometrical features by use of neural networks and genetic programming with real-time simulation

James Griffin

Received: 17 October 2013 / Accepted: 1 April 2014 / Published online: 29 May 2014

Abstract The capability to generate complex geometry features at tight tolerances and fine surface roughness is a key element in implementation of Creep Feed grinding process in specialist applications such as the aerospace manufacturing environment. Based on the analysis of 3D cutting forces, this paper proposes a novel method of predicting the profile deviations of tight geometrical features generated using Creep Feed grinding. In this application, there are several grinding passes made at varying depths providing an incremental geometrical change with the last cut generating the final complex feature. With repeatable results from coordinate measurements, both the radial and tangential forces can be gauged versus the accuracy of the ground features. The results of the tangential force were found more sensitive to the deviation of actual cut depth from the theoretical one. However, to make a more robust prediction on the profile deviation, its values were considered as a function of both force components. In addition, the power signals were obtained as these signals are also proportional to force and deviation measurements. Genetic programming (GP), an evolutionary programming technique, has been used to compute the prediction rules of part profile deviations based on the extracted radial and tangential force and correlated with the initial "gauging" methodology. It was found that using this technique, complex rules can be achieved and used online to dynamically control the geometrical accuracy of the ground features. The GP complex rules are based on the correlation between the measured forces and recorded deviation of the theoretical profile. The mathematical rules are generated from Darwinian evolutionary strategy which provides the mapping between different output classes. GP works from crossover recombination of different rules, and the best individual is evaluated in terms of the given *best fitness value so far* which closes on an optimal solution. Once the best rule has been generated, this can be further used independently or in combination with other close-to-best rules to control the evolution of output measures of machining processes. The best GP terminal sets will be realised in rule-based embedded coded systems which will finally be implemented into a real-time Simulink simulation. This realisation gives a view of how such a control regime can be utilised within an industrial capacity. Neural networks were also used for GP rule verification.

Keywords Grinding · Cutting forces · Spindle power · Profile deviations · Genetic programming · Neural networks · Creep Feed grinding simulation

1 Introduction

For difficult to cut features, there is a need to monitor sensitive output machining parameters to ensure the features are cut to tight tolerances in terms of geometrical accuracy. With some complex geometrical features, there is also a need to produce surface quality that is of very high standards: fine surface roughness (e.g. Ra$<$1 μm) with the tight geometrical accuracies (e.g. h7). There can be a number of different factors that change the cutting conditions which inherently change the cutting path. Among the factors that can make an impact on the grinding conditions can be listed as follows: cutting fluids, stiffness/type of the grinding wheel, depth of cut, cutting speed and feed rate. The changing cutting output conditions can mean the cut features can deviate which is critical to the manufacture of aerospace disk/blade fixtures within aerospace turbines. This essentially means that if the machining is performed outside the required geometrical accuracies through

J. Griffin (✉)
Universidad de Chile, Piso 4, Torre Central, Beauchef 850, Santiago, Chile
e-mail: jgriffin@ing.uchile.cl

worn grinding wheels, undesirable profile deviations can exist. There has already been a lot of research into correlating the change in value of cutting forces in grinding with cutting depth variation and implicitly with the part profile deviations. However, in relation to complex geometry profiles, evolutionary-inspired correlation techniques to monitor cutting forces and profile deviations have not been conducted before. This paper will look at correlating both the radial (Fx) and tangential (Fy) forces in Creep Feed grinding with geometrical deviations of the cut profiles following a succession of depth of cuts. Also correlated to force is the spindle power measurement which can also be used to monitor geometrical tolerances. The correlation is made by an evolutionary computing technique, namely genetic programming (GP). This is an evolutionary-inspired computing technique based on *Darwinian fitness strategies* to gain the correct solution from a survival of the fittest individual search paradigm.

A feed rate model was proposed by Hekman and Liang [5, 6] where the optimisation of feed rate would ensure that one pass would achieve specified tolerances instead of extra additional passes which are often required for the loss in precision. This loss in precision is due to mechanical deflection associated with the inherent compliance of the tool. There have been various other forms of research undertaken to help eliminate the problem.

For instance, robotic deburring used an approach to minimise the effects of machine compliance using force control. This is where a force model utilises a force trajectory and copies the error from magnitude to the next machining pass [10, 18]. Other work within robotic deburring controlled the feed rate which gave a constant force and therefore constant deflection [19]. The work discussed here looks at controlling the deflection/deviation of grinding through different parameters which motivated the research direction within this paper. Hekman and Liang use a dynamic model [5, 6] to provide varying horizontal feed rates based on cost functions correlated with weightings of process time and dimensional error. This dynamic model provided good results and materialised in 13 % increase in surface parallelism. Some of the remaining error however can be attributed to thermal expansion of the workpiece, grinding wheel and machine as grinding has an important thermal output component.

Similar research to the dynamic feed rate control model presented by Chen [2] refers to the control of spark-out grinding pass to ensure the surface roughness values are kept within acceptable limits. However, in controlling the surface roughness, there are additional factors that need to be controlled such as the machine stiffness, grinding deflection and wheel sharpness.

Razavi et al. [13] investigate the different grinding wheel topologies and effects of various coolant applications. This is a very important issue as both deflections of the grinding wheel and workpiece can cause the profile deviations which are attributed to high burst temperatures during the abrasive process.

Axinte and Axinte [1] investigated the cutting efficiency of different coolant fluids used for milling, drilling, tapping and Creep Feed grinding (Fig. 1). This is where the deviation of the ground surface would be measured against a theoretical profile which is considered as the perfect cutting case. Certain cutting fluids provide better cutting conditions with more lubrication to cutting process and ensuring more accurate cuts are obtained. However, the paper is not investigating ways to monitor/minimise the profile deviations generated during multi-pass Creep Feed process.

Machine learning techniques have played a vital role in assisting machine process monitoring. Of the many techniques applied to monitoring, neural networks (NNs) are considered the most extensive; however, in the last decade, a number of different classifiers with a greater emphasis towards evolutionary strategy techniques have emerged as successors to this type of classifier. One successor that is providing more robust results in terms of control is GP. GP if programmed correctly can use mini-neural networks and evolve in evolutionary fashion to give the best mapping for different presented demes of information. This is considered very useful for machine monitoring where many

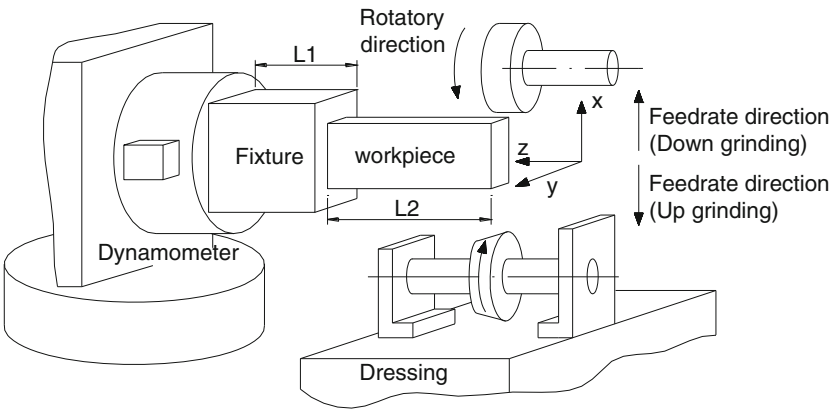


Fig. 1 A sketch of Creep Feed grinding setup. It is used to carry out the geometrical form tests

different cutting conditions and training/test data are present. Griffin and Chen [3] used NNs to monitor acoustic-extracted signals correlated to different grinding phenomena such as cutting, ploughing and rubbing which essentially leads to more efficient grinding. Such NN paradigms can be used for correlating force-extracted signals with the profile deviation differences. However, this technique might be inaccurate to apply to multiple output criteria. In addition, NNs as a classifier suffer in terms of multiple application classifications. This is in regard to modelling many different cutting conditions or even different machining processes. This is where other classifying techniques such as GP are more versatile and provide different rules from being run individually with concentrated data sets and then merged together to provide a robust classifier for many different conditions and/or different machining processes as seen in previous work [4]. For instance, GP has been used in difficult pattern recognition situations such as the work presented by Howard et al. [7, 8, 9]. This is where GP is configured to recognise ships and vehicles from satellite or aircraft synthetic aperture radar (SAR) imagery. The evolutionary technique provided rules that can differentiate between noise, definite targets and even maybe targets using statistical functions and mathematical functions. In providing distinguishing rules, this technique was thought extremely valuable for the multi-condition monitoring environment. Currently, there is very little or no work with GP in monitoring of cutting processes on which this paper reports on. The GP rules can be linked together giving multiple outputs in a 'divide and conquer' fashion [17] looking at two or three classifications at any one time. NNs can give multiple outputs; however, the accuracy starts to diminish when faced with overlapping data sets and, in some circumstances, GP provides a more robust solution to NN and this can be shown in the grinding form simulation (see Section 6). Both accurate GP and NN models are used to display the control authority of such techniques when faced with dynamic environments controlling deviations to obtain tight geometrical tolerances.

2 Profile deviations from Creep Feed grinding

The profile deviation tests were carried out on a Makino A55 Machine Centre. Creep Feed grinding cutting conditions are as follows: 140-mm-diameter aluminium oxide wheel with dedicated profile; v=35 m/s, successive a_{p1}; $a_{p2}=a_{p1}/2$; $a_{p3}=a_{p1}/6$; $a_{p4}=a_{p1}/40$ with intermittent wheel re-dressing; and 70-bar coolant supply. The material of the test block was Inconel 718 where the grinding trials followed a different cutting procedure for each face of the test block:

Face 1: dressing followed grinding with ap1
Face 2: re-dressing followed by grinding successively with ap1, ap1 and ap2
Face 3: re-dressing followed by grinding successively with ap1, ap1, ap2, ap2 and ap3
Face 4: re-dressing followed by grinding successively with ap1, ap1, ap2, ap2 and ap3, followed by re-dressing and a finish cut with ap4

These four cuts would allow measurements to take place in terms of measuring the profile at the different stages when machining the targeted aerospace feature. The work block consists of four faces cut twice at either end provided two finished profiles. The flood coolant has 7 parts of oil out of 20 parts of aqueous solution.

Here, it can be seen that the diamond dresser was used to provide the complex geometrical form (see Fig. 2). For Creep Feed grinding, the coolant was supplied to the interface between the wheel and the workpiece. The workpiece is fixed to the dynamometer through a fixture which allows the measurement of signals which were extracted during this work.

To evaluate the deviations from the theoretical profile, a Coordinate Measuring Measurement (CMM) System (Mitutoyo Eoro-C-A121210 with 2-mm stylus diameter) was used. Such measurements correlated with the associated force could then be used to control the profile deviations and ultimately ensure both automated and accurate cutting conditions. Figure 2 gives an example of the theoretical profile reference the complex geometry. The measurement points, *A*, *B*, *C*, *D*, *E*, *F*, *G*, *H* and *I*, are all deviation means to give a more gaussian-like value for each profile deviation. Each of these means would sum together, and the total mean value would then be used for the force correlation. In addition, the same respective power correlations would be made to the corresponding force to profile deviation mappings. The deviation measurements and force correlations for classifications were mainly taken from face 1 as this was considered to be the most accurate form to measure deviation knowing the wheel had been dressed and the block had not been cut before.

The force measurements were taken from a dynamometer housed within the A55 machine (dynamometer, Kistler 9272 four-axis dynamometer; accelerometer, Kistler 8692C10M1 thread mounted; and spindle power sensor, Load Control PPC3). For large geometrical deviations, power measurements were found to be sensitive to change; however, a more robust sensing strategy is required for smaller deviational

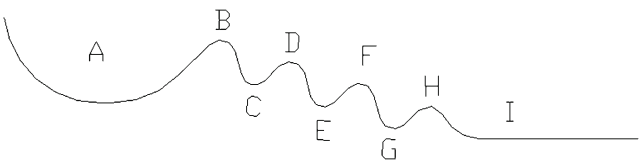


Fig. 2 Theoretical profile and respective CMM measurement points

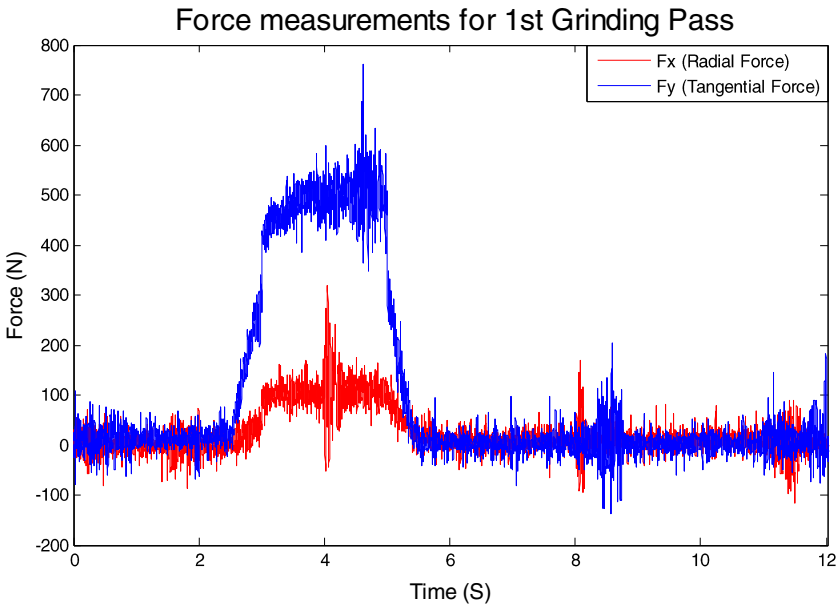


Fig. 3 Fx and Fy force measurements for first grinding pass (ap1)

changes. More focus is placed on mappings between force Fx and geometrical deviations as Fx was found to be more sensitive to geometrical change. However, power to force mappings was necessary to implement a Creep Feed grinding geometrical control simulation in Simulink. It was considered that the more wear that was present on the grinding wheel, the more force was needed to cut the profile. This impacted on the more deviations occurring from the theoretical profile. The mean deviations ranged from 0.1 to 0.33 mm. An example of the force measurement for a first grinding pass with a_{p1} is displayed in Fig. 3 and the corresponding power signal in Fig. 4.

3 Genetic programming and regression analysis

With force and deviations for six individual cuts, it was necessary to use a regression analysis tool where two variables (Fx and Fy) were correlated with one another, and the deviation of the ground profiles would be presented as training and test data. For real online monitoring, this is considered as a useful technique to obtain as many training cases as possible in order to give the classifier the amount data it needs to be able to map all possible combinations that can occur during machining. If however once such an online paradigm is used within a monitoring system, it is then possible to update the

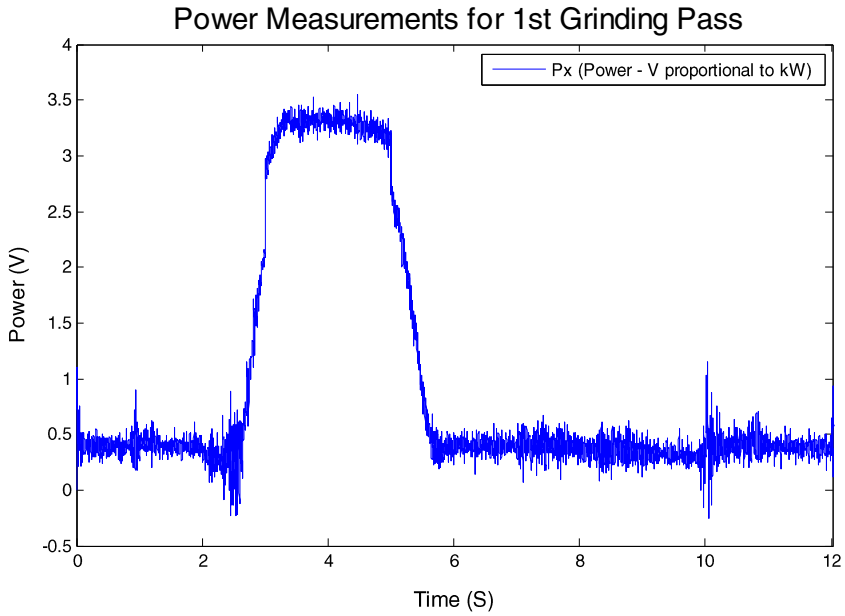
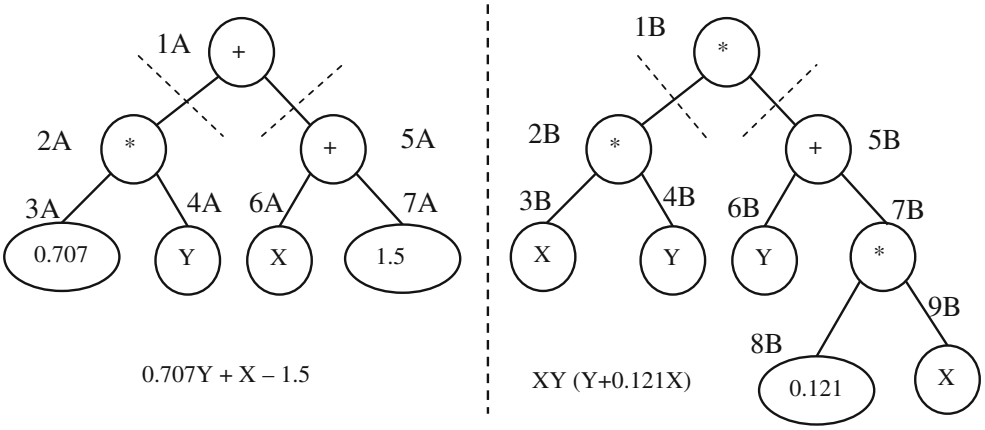


Fig. 4 Px power measurements for first grinding pass (ap1)

Fig. 5 **a** Parent 1. **b** Parent 2

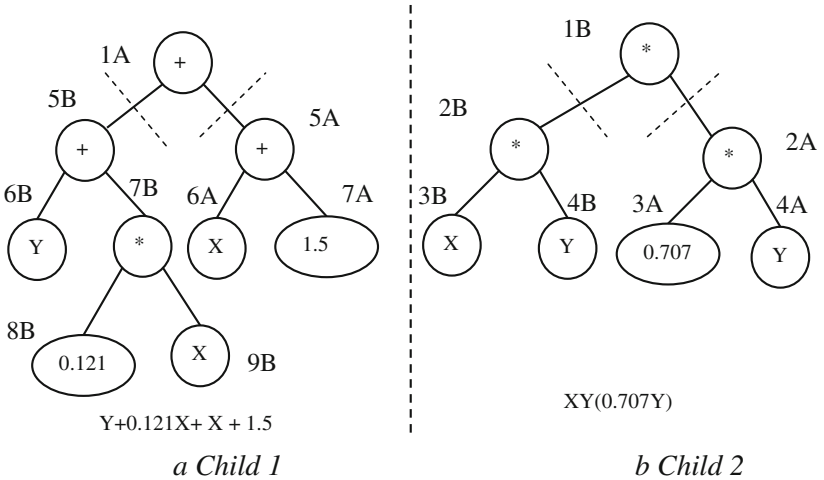


a Parent 1 *b Parent 2*

rules from using feedback where new cases (unforeseen circumstances) are updated to the classifier in the form of variable weights and operating limits, and the system evolves in terms of classifying new extremes within machining. Figure 3 shows the different values for Fx and Fy forces; this is in regard to the beginning and end of each grinding pass.

Once the data had been regressed ensuring a sufficient amount of data was obtained for training, it would then be converted into text form and applied to GP algorithm. GP is based around *Darwinian fitness measures* where both terminals and nodes are selected to ensure the correct functionality is obtained to distinguish different states which in this case are the deviations. The GP algorithm works on the recombination of trees where the nodes and terminals are interchanged with other parents (see Fig. 5a, b) in a tree exchange-like manner. This recombination paves the way for new children with hopefully better genetic material which in short ensures a fitter functionality than what was previously tested (for example, the parents). The children are then made up by a parameter number of individuals which are randomly changed from the recombination process (see Fig. 6a, b). These recombined (crossover points are marked with dashed lines within the respective figures, and this is where the branches are interchanged at the root node level) trees provide integer values which signify either the phenomenon class or, in this case, a regression function that relates to the forces Fx as *X* and Fy as *Y* (for the classification discussed in this paper, the actual GP output corresponds to X1 and X2, respectively) which correlates to the corresponding profile deviation. In addition, for power, Px relates X1 in a separate GP realisation (GP(2)). For the example displayed in Fig. 6a the right-hand branch of Fig. 5a is exchanged with the right-hand branch of Fig. 5b, and for Fig. 6b, the right-hand branch is exchanged with the left-hand branch of Fig. 5a. In summary, Fig. 6a, b display the children of Fig. 5a, b. *The survival of fittest strategy* is based on the children population being tested for the closest match in regard to the tested data. This individual is known as the *fittest individual* (the fittest individual using mathematical functions, random values and the terminal force values; *X* and *Y* equate to the deviation value) and provides the bases for the next recombination, and ultimately, the children become the new parents which have new children as the next population for

Fig. 6 **a** Child 1. **b** Child 2



a Child 1 *b Child 2*

GP evaluation. The resulting tree function starts to get closer to the input training and test data sets and is either halted from the *fittest individual* providing the functionality to distinguish all data correlations or a parameter such as the amount of generations is obtained.

In the research carried out within this paper, GP has been used to control crisp deviation outputs and could be used directly in a control environment. The results displayed in the Section 4 display the output strings and other GP output parameters assisted by tables displaying the accuracy of such advanced evolutionary classifiers. The reader should note that some of the correlations have been purposely made in error to check the robustness of the classifier and still GP outputs a value that is very close to the desired control within a changing environment.

4 Evolutionary grinding profile deviation classifier

This section investigates and tabulates grinding profile data that is blotted from regression data techniques and applied to the GP classifier. For comparison purposes, a NN was used to classify the same data set and comparisons are made between the two classifier systems.

Figure 7 displays the accuracy versus complexity which is in terms of GP tree size (number of tree levels and nodes) required to provide the required fitness to output the correct/near correct deviation values. From Fig. 8, it is possible to see that tree structural complexity in terms of size increases, and the fitness measure decreases which is directly proportional to accuracy increase.

Figures 7, 8 and 9 display the difficult classification problem where GP is applied to different problem data sets, namely the power to force, power to deviation and force to deviation relationship data sets. The GP (GP(1)) paradigms used here used the following terminal sets: +, −, / and * which affords a powerful classifier capability however in some cases to the detriment of bloated rule sets. To get around bloated rule sets (difficult to implement in real-time systems), bloat constraints were employed to limit the growth and obtain accurate classifications of the presented data sets [16]. Note both Figs. 8 and 9 display a decrease in tree size which is indicative of bloat control trying to establish rules that can be easily transferred into a simulation model and finally an embedded control system.

However, in obtaining the force to deviation classification, the accuracy was very low and required a more free-form process to obtain a more complex rule base system which can be seen from GP output results given by Figs. 10, 11, 12, 13 and 14. This produced a more accurate rule set; however, when looking at (3) from Table 1, it can be appreciated that this free-form rule is too complex to be applied in a real-time simulation realisation. Finally, looking at GP(1) results

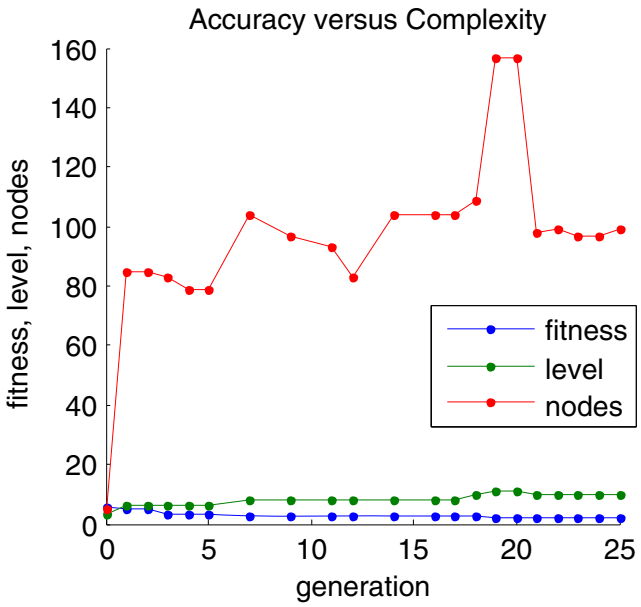


Fig. 7 GP accuracy versus complexity for power and force

(reference Table 2), it was noticed that the GP relationship between power and force was not accurate enough to apply to a real-time control realisation and therefore another route needed to be taken.

GP(1) gives a powerful mathematical rule set; however, it suffers from accuracy when giving multiple classification output. This led to looking at NN realisations where the classifications were high; however, the very complex relationships between power and force gave poor results that could not be used in such a simulation realisation. This led to a focus in a solution between NN and GP; the GP(2) paradigm (reference Table 2) where the following terminal sets were used: if, >, <, >= and <=. Using such terminal sets affords more constraints when bloat control is applied; hence, the output

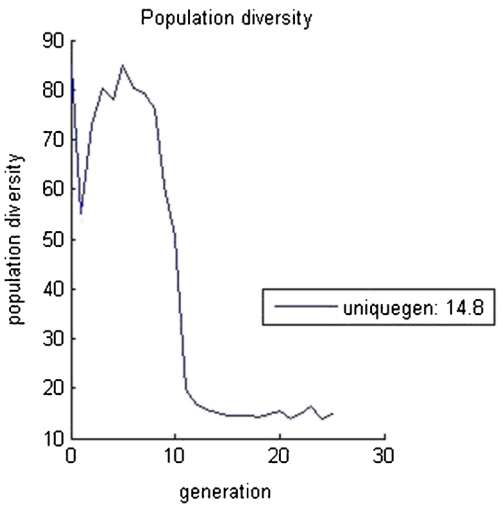


Fig. 8 GP population diversity for power and force to form deviation

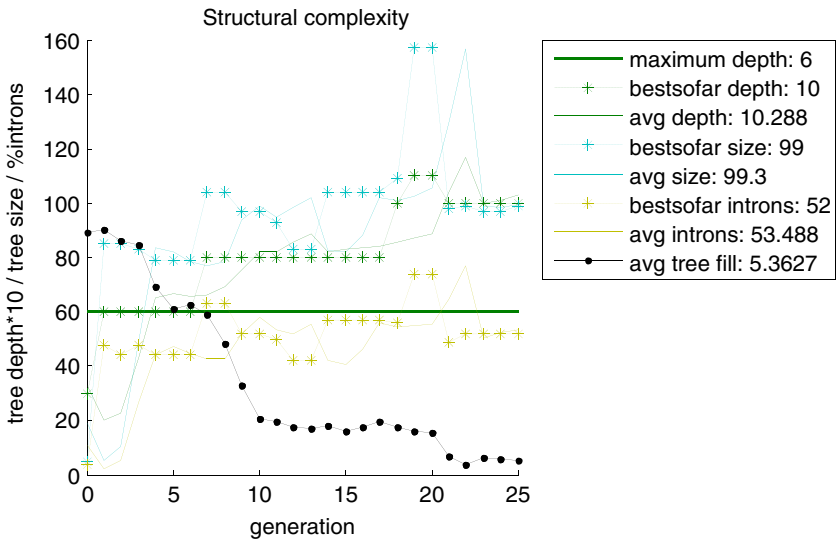


Fig. 9 GP structural complexity for power and force to form deviation

tree rules are much smaller and accurate (see Figs. 14, 15, 16 and 17).

Table 1 displays the tree to string output for classifying the (1) force and power to form deviation (2) force to form deviation and, continued in Table 1, (3) power to force data sets. The string output is obtained from the fittest individual when one of the conditional parameters has occurred within the algorithm; the string is based on equating the tree, terminal and function nodes to give the desired solution such as the profile deviation value. The data is then segmented into a training and test set; the test set contains 50 % unseen examples, and the accuracy of the classifier is measured on the correct deviation for a given set of forces (Fx and Fy). For all data sets, the GP algorithm randomly finds the functional node set; this was due to the complexity of the correlating force data set in a function that would give the correct profile deviations. Table 2 is used to display the results given by both the NN and GP classifiers. It can be seen that the GP paradigm outperforms the NN in all data set cases. In addition, the GP classifier can evolve all three data rule sets and use together as multiple-rule system providing control for all the presented cases.

GP(2) (see Table 2) gave results that were both accurate, fast and easily applicable to bloat control which in short ensured that such classification relationships can be applied to real-time realisation. Figure 14 displays an optimised tree set that allows the link relationship between tangential force and form deviation. This optimised tree set is easily implementable into a real-time realisation. Figures 15 and 16 display the power relationships to both tangential and radial forces, respectively; again, these relate to an optimised rule set that can be easily realised into a simulation realisation. Finally, Fig. 16 displays the power to form deviation relationship

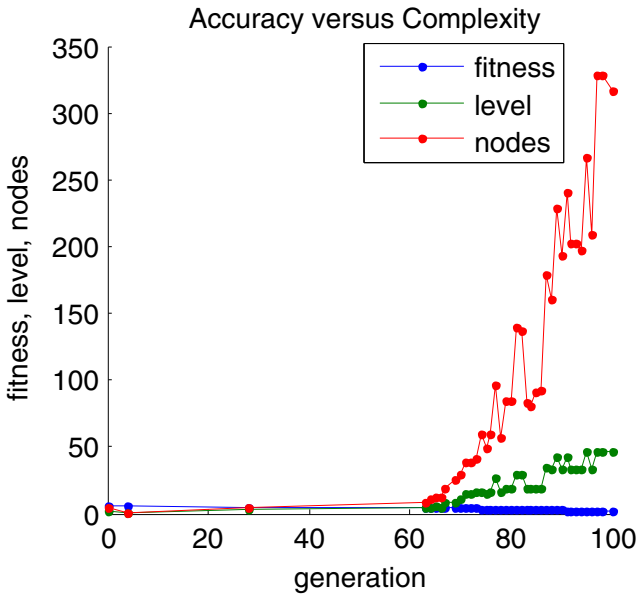


Fig. 10 GP accuracy verses complexity for force to form deviation

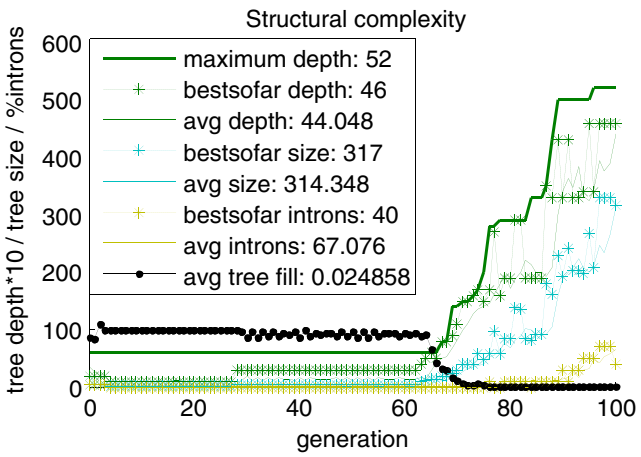


Fig. 11 GP structural complexity for force to form deviation

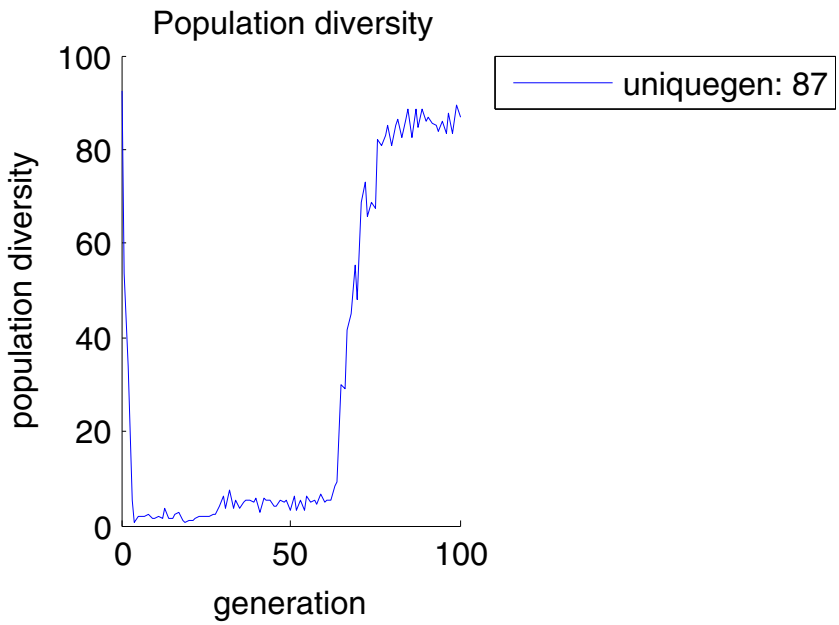


Fig. 12 GP population diversity for force to form deviation

which is an optimised tree rule set and easily transferable into a simulation realisation. In summary, when comparing the two GP model strategies, GP(2) has better bloat control and can separate more multiple complex data segregation when compared with GP(1).

5 Neural network grinding profile deviation classifier

A large number of researchers have reported the application of using NN models for the classification of phenomena of interest when applied to tool condition monitoring [15, 12]. A feed-forward neural network model was used with the back-propagation learning strategy to provide the segregation of data [14]. Commonly, NNs are used for pattern recognition in image analysis or sound waves in signal analysis. The NN consists of a complex interconnection of units which are otherwise known as nodes or neurons. The general layout for a NN consists of a set of neuron layers connected together through complex connections; this layout and features are known as the network architecture. This is where NNs can be applied to the classification of different geometrical deviations based on the input correlations of tangential and radial forces as well as that of spindle power.

A multi-layer NN is required due to the more complex data presented by force and power signals. This type of data is not only nonlinear but also n-dimensional. The basic logic function network classifiers (such as OR, NOT and AND) use a linear data separation approach; however, with the separation of much larger data sets, there is need for a more dynamic

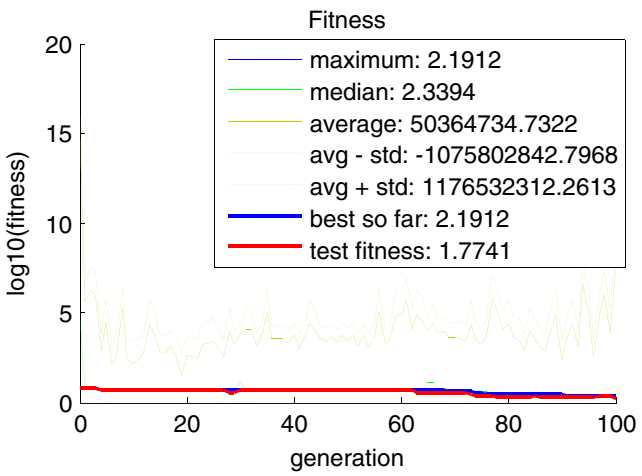


Fig. 13 GP fitness for grinding force to form deviation

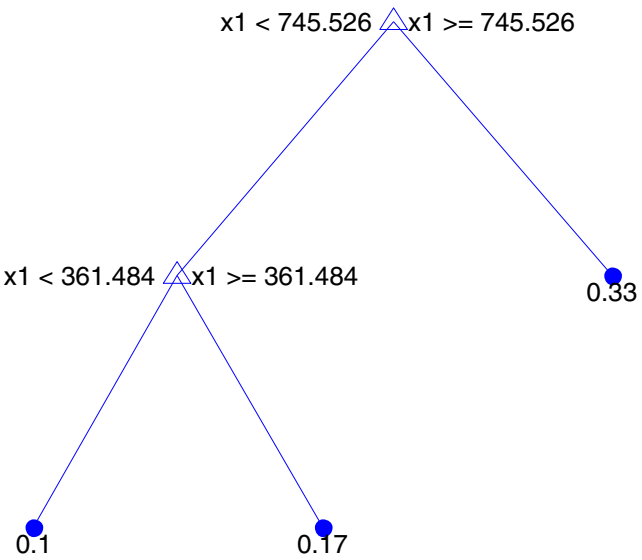


Fig. 14 GP tree for tangential grinding force to form deviation

Table 1 GP functional tree output for start end, both profile deviation data sets

(1) treeanswer_force and power to deviation = [myif(times(mydivide(mydivide(X1,X2),plus(X2,X2)),mydivide (plus(X2,X1),times(X3,X2))),minus(mydivide(myif(X2,X3,X2),plus(X3,X3)),mydivide(myif(X1,X2,0.64656), myif(X2,myif(times(0.53668,X1),times(X3,X2),minus(X2,X2)),X1))),myif(myif(times(X1,X1),times(X3,X2), minus(X2,X2)),times(minus(X3,X2),plus(X2,X3)),mydivide(minus(X1,X2),mydivide(X2,X2))))]

where ans >= 0.70594 & <0.109505267=0.1 and ans >=0.144128114 & <0.257213314 = 0.17 and ans >=0.287872681& < 0.316570507 = 0.33 if values exist in between these demarcations a percentage utilization between the deviation classes is given which allows such a system to be used in real time

(2) treeanswer force to deviation = [minus(0.1746,mydivide(mydivide(plus(mydivide(plus(times(X2,X2),X2), minus (0.71303,X2)),minus(plus(mydivide(minus(plus(X2,minus(minus(0.71303,X1),X2)),myif(X2,X1,mydivide(mydivide (0.27435, myif(times(X1,0.46921),times(X1,X2),X1)),times(X2,X1)))),X2),X1),myif(X2,myif(X1,X2,X1),X2))),X2),plus (mydivide(plus(times(X2,X2),X1),minus(minus(plus(mydivide(plus(times(X2,X2),X2),minus(0.71303,X2)),minus(plus (mydivide(0.67279,X2),X1),myif(X2,myif(X1,X2,X1),X2))),0.50975),X2)),minus(plus(X2,minus(minus(0.71303, 0.86212),X2)),myif(X2,myif(X1,X2,X1),minus(X2,X2))))))]

where ans >=0.070594 & <0.109505267 = 0.1 and ans >= 0.144128114 & <0.257213314 = 0.17 and ans >=0.287872681& <0.316570507 = 0.33 if values exist in between these demarcations a percentage utilization between the deviation classes is given which allows such a system to be used in real time

(3) treeanswer force to deviation = [mydivide(times(minus(X1,minus(X2,times(minus(times (times(minus(X1,0.66651),times(minus(X1,0.66651),0.047566)),0.047566),minus(0.047566,times(minus(X1,times (minus(X1,0.66651),mydivide(X2,minus(X2,times(minus(0.047566,minus(X1,times(minus(X1,0.66651),times(minus (X1,0.66651),0.047566)))),0.047566))))),mydivide(minus(X2,times(times(minus(0.047566,minus(0.047566,times(minus (X1,0.66651),times(minus(X1,0.66651),0.047566)))),0.047566),0.047566)), minus(X2,times(minus(times(minus(times(minus (X1,0.047566),times(minus(X1,0.66651),mydivide (times(minus(X1,minus(X2,times(minus(0.66651,minus(0.047566, times(minus(X1,times(minus (mydivide(X1,plus(times(0.097488,X2),X1)),X1),mydivide (X2,minus(X2,minus(X1,X2))))), mydivide (X2,minus(X2,times(minus(times(times(minus(X1,minus(X2, times(minus(0.66651,minus(0.047566, times (minus(X1,times(minus(X1,X1),mydivide(X2,plus(0.068922,0.40312)))),mydivide(X2,minus (X2,times(minus(times (times(minus(X1,0.66651),times(minus(X1,0.66651),times(minus(X1,0.66651),0.047566))),0.047566),minus(times (minus(0.047566,0.66651),0.047566),times(minus(X1,0.66651), times(minus(0.047566,0.66651),0.047566)))),0.047566)))))), 0.047566))),times(minus(X1,0.66651),0.047566)),0.047566),minus(minus(X1,0.66651),times(minus(X1,0.66651),times (minus(0.047566,0.66651),0.047566)))),minus(0.66651,minus(0.047566,times(minus(X1,times(minus(X1,X1),mydivide(X2, minus(X2,minus(X1,times(minus(X1,0.66651),times(minus(X1,0.66651),0.047566)))))),mydivide(X2,minus(X2,times(minus (times(times(minus(X1,0.66651),times(minus(X1,0.66651),times(minus(X1,0.66651),0.047566))),0.047566),minus (minus(X1,0.66651),times(minus(X1,0.66651),times(minus(0.047566,0.66651),0.047566)))),0.047566)))))))),0.047566)), 0.047566),minus(X2,times(times(minus(0.047566,minus(0.047566,times(minus(X1,0.66651),times(minus(X1,0.66651), 0.047566)))),0.047566),0.047566))))),0.66651),0.047566),minus(minus(X1,0.66651),times(minus(X1,X1),times(minus(X1, 0.66651),0.047566)))),0.047566))))))),0.047566))),0.047566),minus(X2,times(times(minus(0.047566,minus(0.047566,times (minus(X1,0.66651),times(minus(X1,0.66651),0.047566)))),0.047566),0.047566)))]

where ans >= 0.099744 & <0.115591 = 0.1 and ans >= 0.160132 & <0.241507=0.17 and ans >= 0.256763 & <0.393479 = 0.33 if values exist in between these demarcations a percentage utilization between the deviation classes is given which allows such a system to be used in real time

(4) treeanswer power to tangential force = [times(times(plus(times(X1,X1),X1),mydivide(plus(X1,0.29724), times(X1,0.2503))),plus(minus(mydivide(X1,X1),myif(X1,0.10943,X1)),myif(X1,plus(0.82334,mydivide(plus (minus(mydivide(X1,mydivide(plus(X1,0.2503),times(X1,0.2503))),0.2503),myif(0.085098,plus(minus (mydivide(X1,mydivide(plus(X1,0.10943),times(X1,0.2503))),0.10943),X1),times(X1,0.085098))),times (X1,0.2503))),X1)))]

Where output tests were as follows:

334.7905135	838.7270407	533.7835903	983.046523	961.4171817
936.8083368	592.0601508	494.4422407	516.8914283	258.3366798
485.603015	242.7257455	340.6060942	854.0336369	543.3110098
1,001.110987	979.0676146	953.9879919	602.685434	503.2308267
526.1013669	262.7491321	494.2258835	246.8540862	346.4701897
869.4770516	552.9211292	1,019.33832	996.8769997	971.3221466
613.4036334	512.0952599	535.3910583	267.1973927	502.9230649
251.0156841	352.3828038	885.0572868	562.6139512	1,037.728527
1,014.845339	988.8108033	624.2147516	521.0355432	544.7605054
271.6814658	511.6945621	255.2105436	358.3439398	900.7743447
572.3894787	1,056.281607	1,032.972634	1,006.453964	635.1187911
530.0516794	554.209711	276.2013554	520.5403779	259.4386689

Table 1 (continued)

364.3536011	916.6282273	582.247714	1,074.997563	1,051.258888
1,024.251631	646.1157543	539.1436711	563.7386775	280.7570655
529.4605151	263.7000641	370.4117909	932.6189364	592.1886597
1,093.876397	1,069.704101	1,042.203804	657.2056437	548.3115209
573.3474076	285.3485996	538.4549763	267.994733	

learning system that takes all the information into consideration and maps the data in both parallel and gradient descent segregation fashion such as that seen by the back-propagation feed-forward network [11]. A multi-layer perceptron (MLP) utilising the back-propagation learning rule is presented in Fig. 18 for illustration purposes.

As displayed in Fig. 18, each of the inputs P_1 to P_4 is multiplied by a changing weight function and is associated to a target vector; in this example a_1 to a_2, respectively. This is called the associations of input–output pairs and provides the supervised training data (test and verification data set have both data that has been seen by the network in training (supervised) and data that has not been seen (testing the generalisation of the network)). Each neuron has a summation function which sums up the weighted (for example, $w_{1,1}{}^{1}$ and $w_{1,2},{}^{1}$ to $w_{n,n},{}^{2}$ reference to Fig. 18) and input bias (bias input variable) connections. The transfer function (for nonlinear problems, a differential transfer function, such as Tan-sigmoid, is used) is required to map the nonlinear input–output relations which are obtained for each neuron and updated in an iterative fashion towards the desired target set. Back-propagation is so called as the weights that are updated from the error between the actual output and the desired output which in short is from the back to the front. This method segregates the different classes based on the supervised training data given to the NN. The summation of weights and bias values are multiplied together by a differential transfer function to give a neuron output.

The output of each neuron is a function of its inputs. Specifically, the output of the *j*th neuron is for any layer and is described by the following equations:

$$U_j = \sum \left(Pi \times w_{ij}\right) \tag{1}$$

$$a_i = F\left(U_j + t_j\right) \tag{2}$$

For every neuron, j, in a layer, each of the i inputs, X_i to that layer, is multiplied by a previously established weight, w_{ij}. These are all summed together, resulting in the internal value of the operation, U_j. This value is then biased by a previously established threshold value t_j and sent through an activation function, F (nonlinear or linear), giving the NN output, a_i.

Equation (3) describes the output error obtained from each neuron.

$$\mathrm{ME} = \frac{1}{\Omega}\sum_{i=1}^{\Omega}\left(t_i - a_i\right)^2 \tag{3}$$

where ME is the mean-squared error, and a_i (a_1 and a_2 in the example displayed in Fig. 18) is the output of the network corresponding to *i*th input P_1 to P_4. The error term of network is given from $(t_i - a_i)$ where t_i is the target vector or the desired value for given input vectors P_1 to P_4. The T is used to transpose the matrix to ensure both matrices are multiplied

Table 2 Results for NN and GP classifiers in regard to the respective data sets

Classifier	Data set	Generations	Error/fitness	Classifier size	Accuracy, %
NN	Power to deviation	2,000	5.24×10^{-30}	1/4/1 LS	87
NN	Force to deviation	40,000	1.4×10^{-9}	2/4/1 LS	68
NN	Power to force	40,000	8.8×10^{-1}	1/4/1 LS	48
GP(1)	Power/force to deviation	25	2.008	52 N/28 L	78
GP(1)	Force to deviation	25	2.88	103 N/14 L	74
GP(1)	Force to deviation	100	0.54	233 N/38 L	89
GP(1)	Power to tangential force	100	3.24	317 N/46 L	76
GP(2)	Power to deviation	25	0.154	5 N/3 L	92
GP(2)	Tangential force to deviation	25	0.112	5 N/3 L	94
GP(2)	Power to radial force	25	2.34	23 N/8 L	82
GP(2)	Power to tangential force	25	0.126	23 N/8 L	90

N nodes, *L* levels, *LS* layers

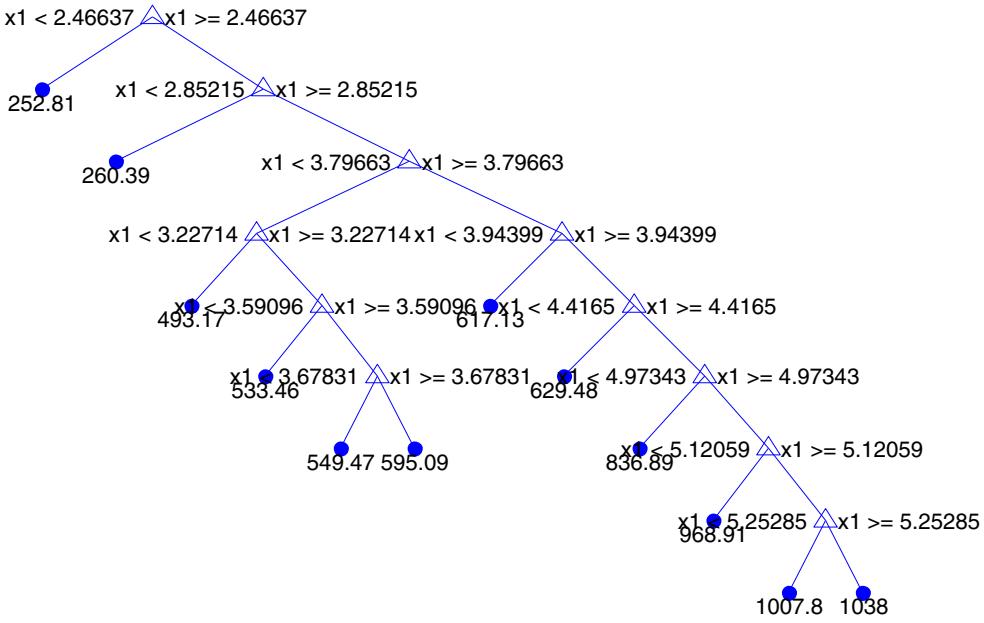


Fig. 15 GP tree for power to tangential grinding force

together to get a sum-squared error output. The error function can be applied to the NN in a batch training fashion at the end of data presentation or sequentially after each input–output pair.

For the back-propagation algorithm, the weight and bias update equations are as follows:

$$\Delta w_{ij}^{k} = -\alpha \frac{\partial \mathrm{ME}}{\partial w_{ij}^{k}} \tag{4}$$

$$\Delta b_{i}^{k} = -\alpha \frac{\partial \mathrm{ME}}{\partial b_{i}^{k}} \tag{5}$$

where α is the learning rate, which has a trade-off in value to ensure it is small enough to gain a true convergence but large enough to separate the data space in adequate time. Equations 4 and 5 are iteratively changed across the network along with other functions to provide learning sensitivity. This process of weight and input and error calculation propagates through the NN to provide the segregation rules which separate the data according to class (target vector). The b is a bias term used to influence the training weights and for NN training.

The NN was used to work as a secondary classifier to verify the GP rule sets and was thought a good second multiple class output classifier albeit not for massively

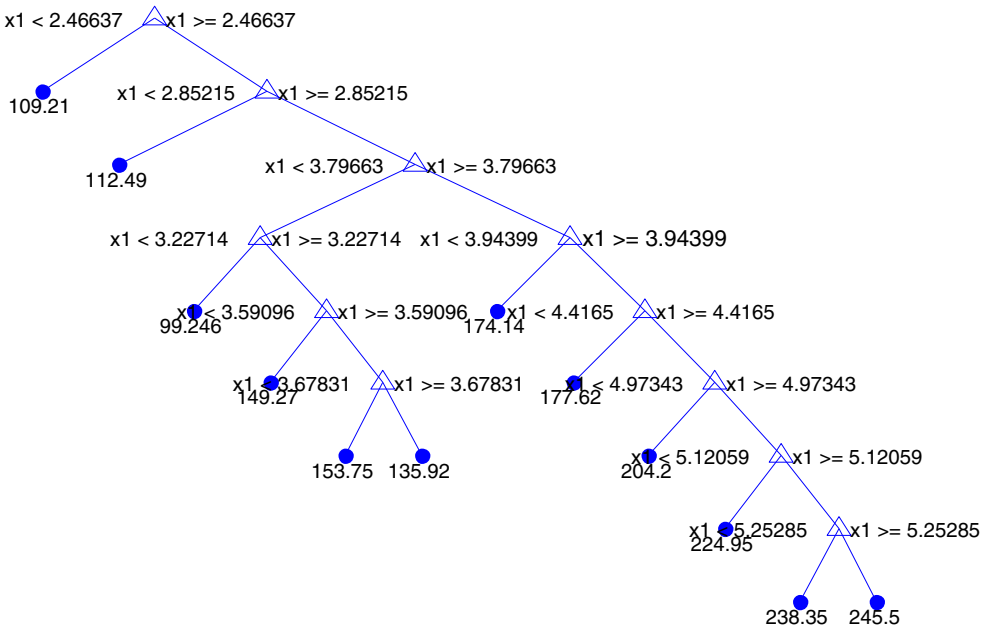


Fig. 16 GP tree for power to radial grinding force

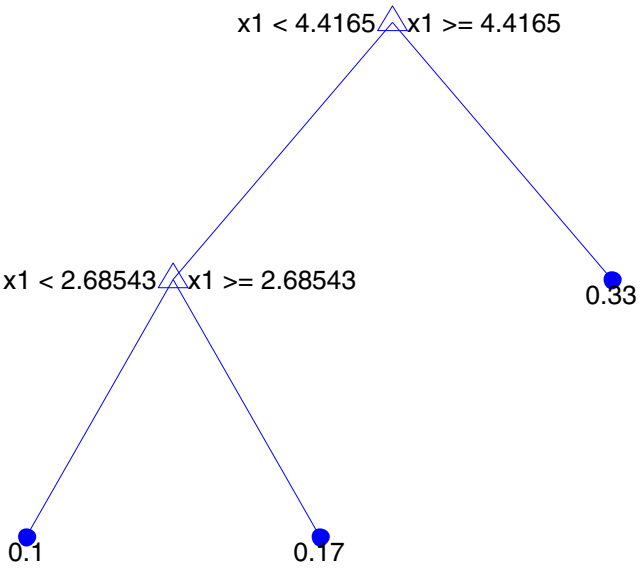


Fig. 17 GP tree for power to form deviation

overlapping data sets where local minimums can be found instead of global. The NN was therefore a good classifier for power to deviations and force to deviations; however, not power to force correlations as this was considered a more difficult problem to map. This is where GP was used with data mining type terminal sets such as >=, <= and if statements. Such terminal sets gave good results in terms of both fitness and size metrics (see Table 2 for results).

Figure 19 displays some encouraging results for NN classification of power to form deviation errors; however, such classifications are not as accurate as seen with GP classifications. This is due to GP providing more complex separation/demarcation between the different class sets. The NN results displayed here are encouraging where the error outputs are close to the desired values (see Fig. 19, red boxes to blue desired targets), and this further promotes why a NN paradigm is used to reinforce the GP control loop.

6 Evolutionary grinding profile deviation simulation model

Figures 20 and 21 display the real-time realisation of Simulink realisation which uses the grinding parameters to form the complex geometries of Creep Feed grinding profiles discussed in Section 2 of this paper. By changing the depth of cut (DOC), the power relationships are changed, which proportionally changes the force relationship. Both the power and force correlate to the changing geometric form deviations which are associated with the

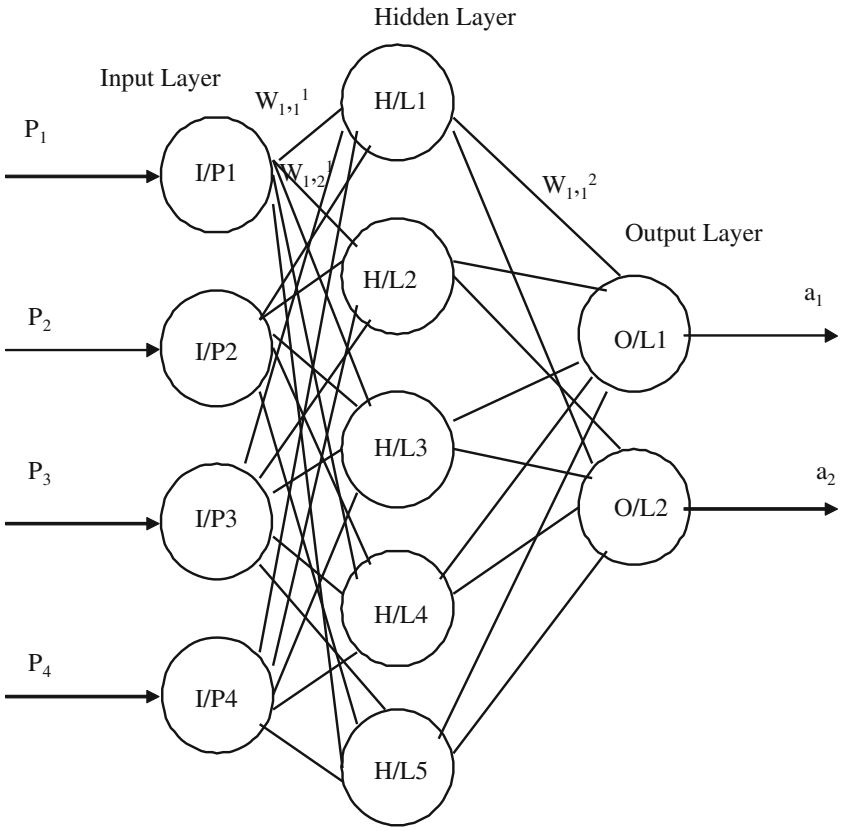


Fig. 18 A four-input NN with one hidden layer

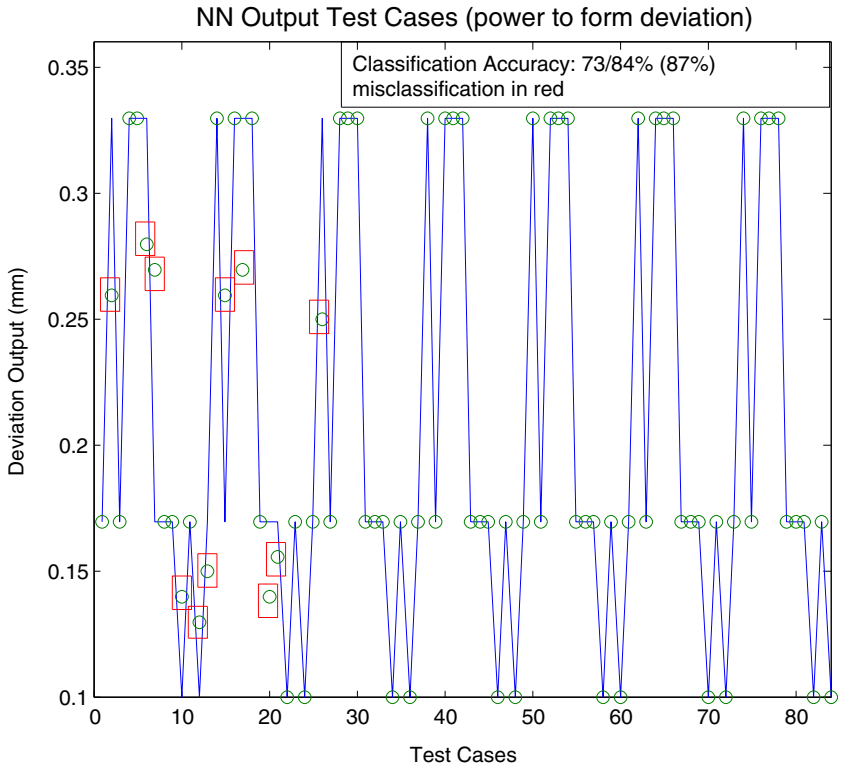


Fig. 19 NN output of grinding form deviations in mm

production of complex geometries during the manufacture of turbine blade root forms.

Figure 22 displays both the input and output characteristics for a Creep Feed grinding simulation for controlling deviations based on the power and force delivery. Also within Fig. 22 is a power relationship which is proportional to change in DOC, and such power relationships are correlated with forces which are both observed as factors to deviations in accurate grinding form geometries. Figure 23 displays the same simulation inputs and outputs as Fig. 22; however, instead of spindle power, the correlation between DOC, tangential and radial forces to form deviations is displayed.

7 Conclusions

The GP paradigm has displayed both the robustness and accuracy over other rival classifier systems such as that seen by NN. In providing more accurate correlations between forces (Fx—radial forces; Fy—tangential forces) and grinding

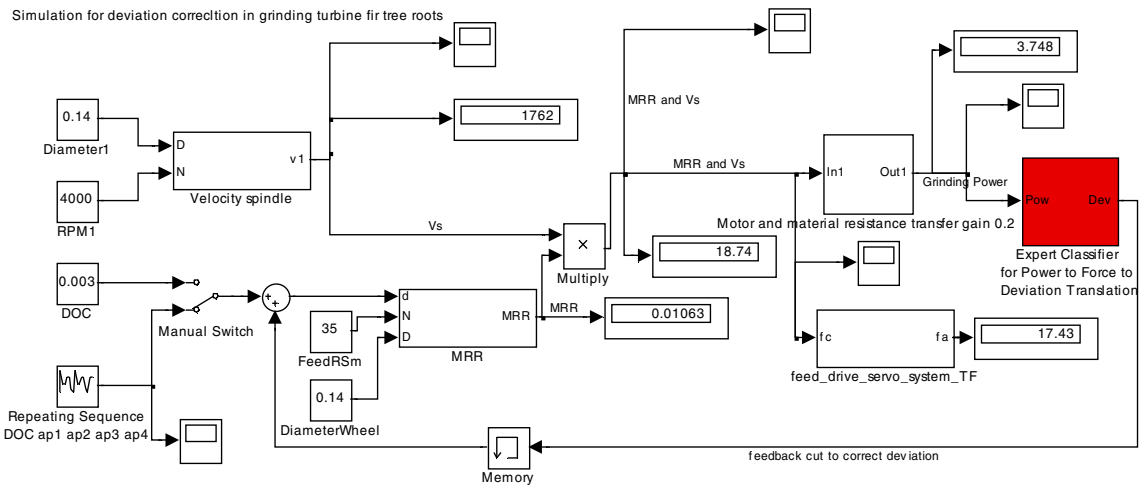


Fig. 20 Simulink simulation realisation for power/force to form deviation

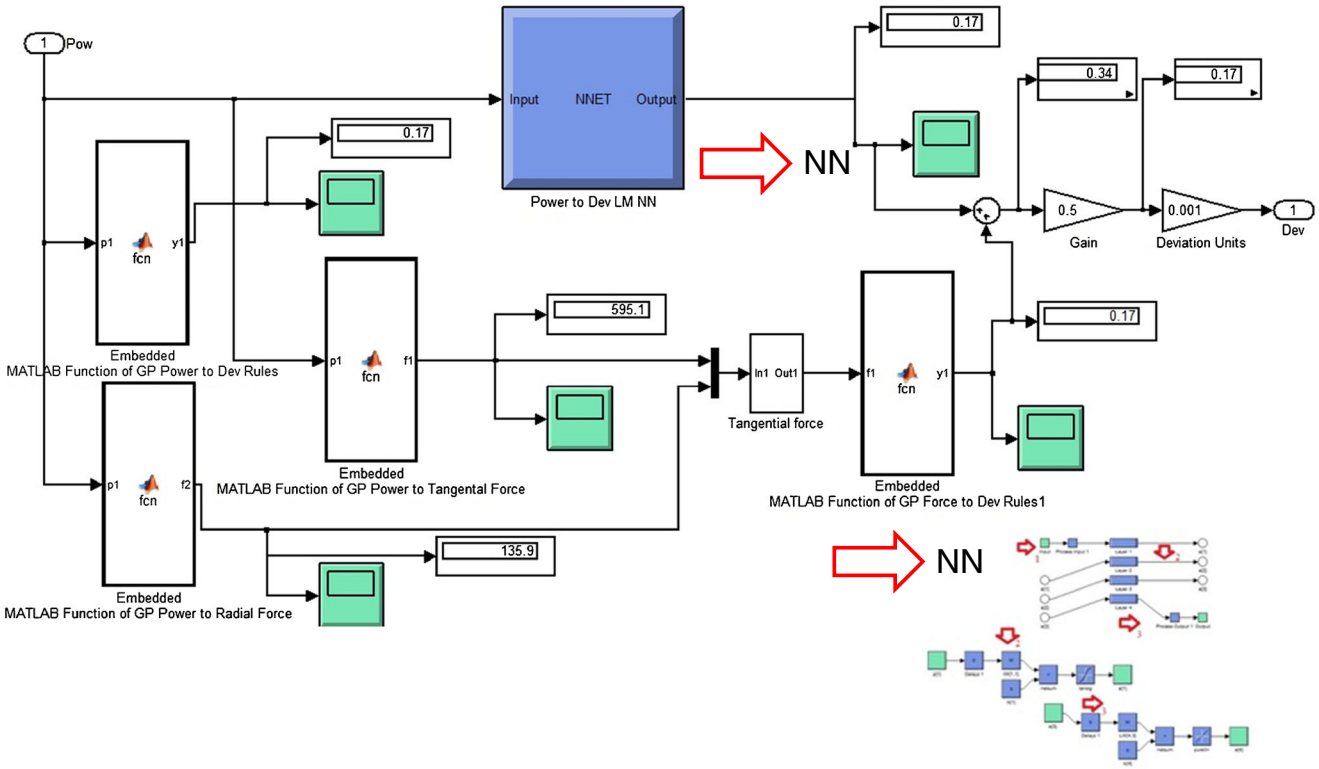


Fig. 21 Simulink simulation realisation for NN and GP control of force/power to form deviation (reference to red box of Fig. 20)

profile deviations, GP is the preferred choice for online machine monitoring purposes. GP can classify different demes of data and can then merge together to provide a multi-rule set controlling different conditions experienced within a

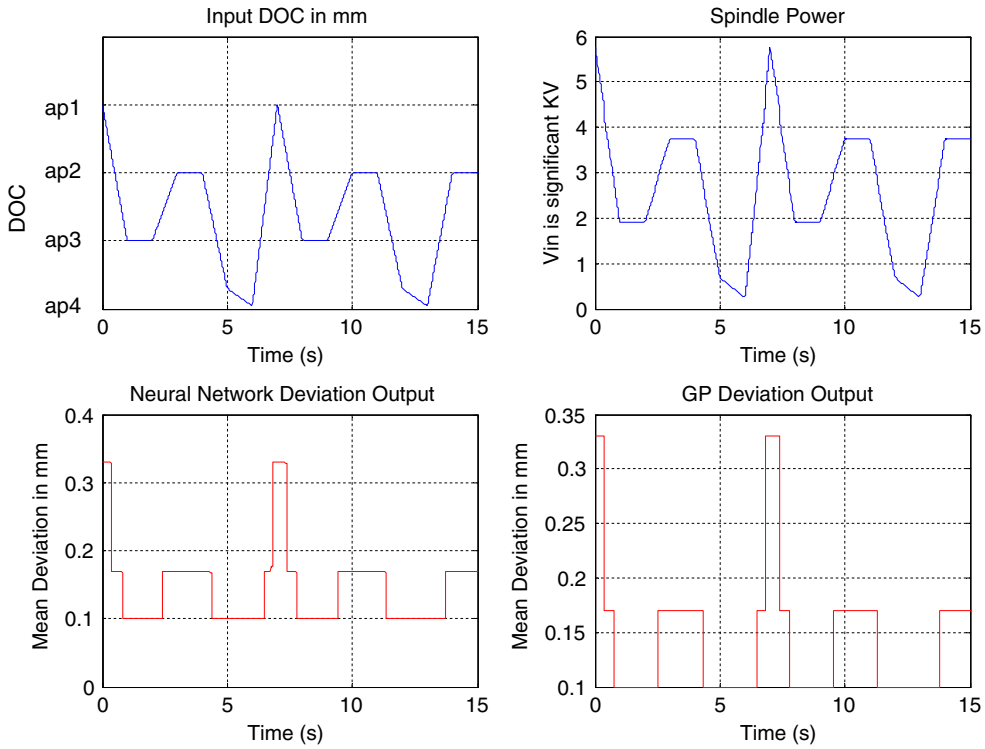


Fig. 22 Simulink simulation realisation DOC and spindle power input with NN and GP form deviation outputs

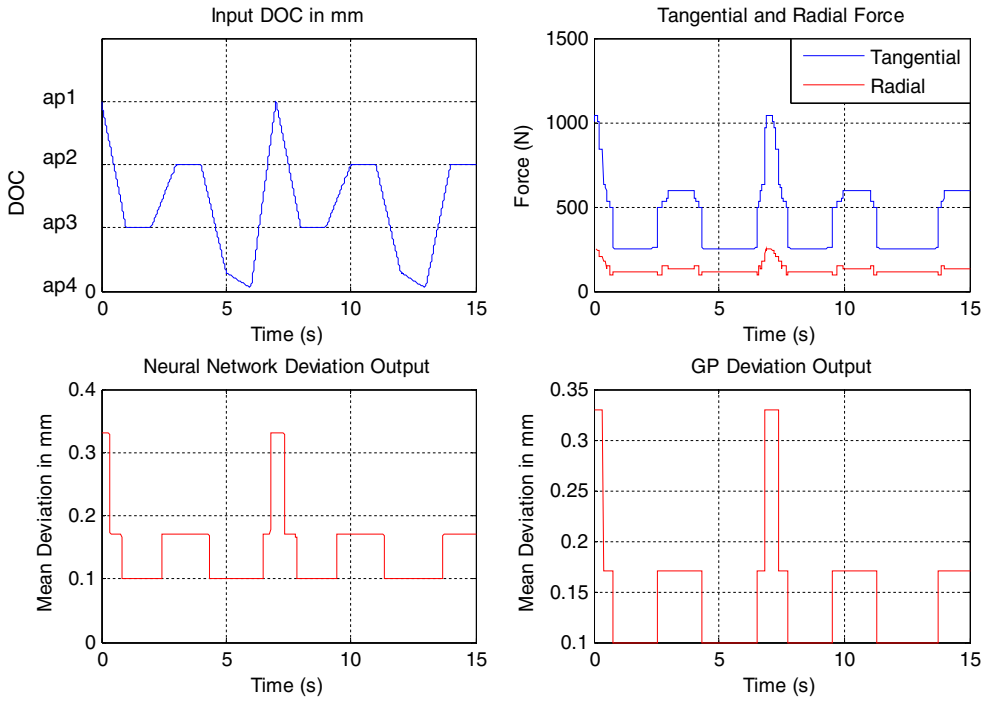


Fig. 23 Simulink simulation realisation DOC and both tangential and radial force inputs with NN and GP form deviation outputs

machining environment. The work carried out in this paper is particularly important to machining difficult to cut profiles where grinding wheel wear and workpiece deviations are a serious issue when manufacturing critical applications such as machining aerospace turbine blades.

To display a real-time realisation, a Simulink simulation is implemented in Figs. 20 and 21. Here, the relationships between material removal rates (MRR) and DOC are used to provide power and force correlations which ultimately correlate to the measured material deviations. This feedback control using both NN and GP paradigms can be further adapted to give tuned weightings which promote a type of online learning capability; however, an online measuring device is also required for such an idea to be realised. The GP paradigm gave the most complex mappings where the NN failed to give accurate mappings for complex power to force correlations which further reinforces the use for using GP rules for online grinding deviation control. Both classifier systems were used to map the spindle power to form deviation errors as this is considered a critical part of the control loop. This is where two classifiers give a more robust control regime based on both classifier systems giving accurate results (see Table 2). Future work will look into online tuning with respect to online measurement feedback.

Acknowledgments The authors are grateful to their colleagues M. Axinte and M. Daine from The University of Nottingham for their valuable technical support during the experimental investigation. Please also note that the experimental work was carried out at The University of Nottingham funded by EPSRC.

References

1. Axinte D, Axinte M (2003) A multicriteria model for cutting fluid evaluation. Proc IME B J Eng Manuf 217(10):1341–1353
2. Chen X, Rowe WB (1999) Modelling surface roughness improvement in grinding. Proc IME B J Eng Manuf 213(1):93–96
3. Griffin J, Chen X (2009) Characteristics of the Acoustic Emission during Horizontal Single Grit Scratch Tests – Part 2 Classification and Grinding Tests. International Journal Abrasive Technologies-Special Issue on: Micro/Meso Mechanical Manufacturing (M4 Process). Vol. 1, No 4
4. Griffin J, Chen X (2009) Multiple classification of the acoustic emission signals extracted during burn and chatter anomalies using genetic programming. Int J Adv Manuf Technol 45(11–12):1152–1168
5. Hekman KA, Liang SY (1998) Feedrate optimization and DOC control for productivity and parallelism in grinding. Mechatronics 9:447–462
6. Hekman KA, Liang SY (1998) Flatness control in grinding by depth of cut manipulation. Mechatronics 8(4):323–335
7. Howard D, Roberts SC et al (1999) Target detection in SAR imagery by genetic programming. Adv Eng Softw 30(5):303–311
8. Howard D, Roberts SC et al (1999) Evolution of ship detectors for satellite SAR imagery. Genet Program 1598:135–148
9. Howard D, Roberts SC et al (2006) Pragmatic genetic programming strategy for the problem of vehicle detection in airborne reconnaissance. Pattern Recogn Lett 27(11):1275–1288
10. Kurfess TR, Whitney DE (1992) Predictive control of a robotic grinding system. J Eng Ind 114(4):412–420
11. McCulloch WS, Pitts WH (1943) A logical calculis of the ideas immanent in nervous activity. Bull Math Biophys 5:115–133

12. Ozel T, Karpat Y (2005) Predictive modelling of surface roughness and tool wear in hard turning using regression and neural networks. Int J Mach Tools Manuf 45:467–479
13. Razavi HA, Kurfess TR et al (2001) Force control grinding of gamma titanium aluminide. Int J Mach Tools Manuf 43:185–191
14. Rumelhart DD, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536
15. Sick, B. (2002) On-line and indirect tool wear monitoring in turning with artificial neural networks: a review of more than a decade of research. Mechanical Systems and Signal Processing 16, pp 487–546
16. Silva, S. (2004) A genetic programming toolbox for Matlab—version 2. ECOS—Evolutionary and Complex Systems Group, University of Coimbra, Portugal, http://gplab.sourceforge.net/
17. Silva, S. and Y. T. Tseng. (2005) Classification of seafloor habitats using genetic programming. GECCO '05. Washington
18. Srivastava AK, Elbestawi MA (1995) Control strategy for multipass robotic grinding. Int J Robot Autom 10(3):114–119
19. Yoo, S. M. (1990) Computer simulation of the flexible disk grinding process: flat surface control using variable vertical feed speed. Symposium on Monitoring and Control for Manufacturing Processes. *ASME WAM PED* 44: 123–32