UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Identifying Self-excited Vibrations with Evolutionary Computing

Erdbrink, C.D.; Krzhizhanovskaya, V.V.

Identifying Self-Excited Vibrations with Evolutionary Computing

Christiaan D. Erdbrink[1, 2, 3*] and Valeria V. Krzhizhanovskaya[1, 2, 4]

[1]University of Amsterdam, Amsterdam, The Netherlands
[2]ITMO University, Saint Petersburg, Russia
[3]Deltares, Delft, The Netherlands
[4]Saint Petersburg State Polytechnic University, Russia
chriserdbrink@gmail.com, v.krzhizhanovskaya@uva.nl

Abstract

This study uses differential evolution to identify the coefficients of second-order differential equations of self-excited vibrations from a time signal. The motivation is found in the ample occurrence of this vibration type in engineering and physics, in particular in the real-life problem of vibrations of hydraulic structure gates. In the proposed method, an equation structure is assumed at the level of the ordinary differential equation and a population of candidate coefficient vectors undergoes evolutionary training. In this way the numerical constants of non-linear terms of various self-excited vibration types were recovered from the time signal and the velocity value only at the initial time. Comparisons are given regarding accuracy and computing time. Dependency of the test errors on the algorithm parameters is studied in a sensitivity analysis. The presented evolutionary method shows good promise for future application in engineering systems, in particular operational early-warning systems that recognise oscillations with negative damping before they can cause damage.

Keywords: evolutionary computing, system identification, self-excited vibrations, differential evolution

1 Introduction

This paper explores the use of evolutionary computing (EC) for the identification of vibration types appearing in many problems in physics and engineering. As an illustrative example, we consider the flow-induced vibrations of large gates of hydraulic civil engineering structures, such as flood barriers and weirs. In particular, self-excited vibrations have been found to be a cause of threatening dynamic forces associated with fluid-structure interactions between the discharge-controlling gates and the turbulent flow moving past them (Kolkman 1976).

---

* Corresponding author, alternative email addresses: c.d.erdbrink@uva.nl and christiaan.erdbrink@deltares.nl

Like most flow-induced vibrations, gate dynamics can be suitably investigated by modelling the suspended gate body as a one degree-of-freedom mass-spring oscillator (Blevins 1990) which is classically described by second-order ordinary differential equations (ODEs). For oscillating objects submerged in water, the coefficients –representing mass, damping and stiffness– have additional hydraulic components (Naudascher & Rockwell 1994), which can only be estimated by dedicated studies under specific conditions that are typically absent in real-life scenarios.

Self-excited or 'self-induced' vibrations are widely studied because they describe many problems in engineering and physics. They are also analytically and numerically interesting objects of study (Verhulst 1996). This vibration type is defined by the driving force coming from the displacement of the oscillating body itself (Den Hartog 1956). That is, a self-sustained system exists without the need for external forcing. New energy is fed into the system through negative damping. The first two columns in Fig.1 show quintessential self-excitation cases: a negative damping constant and the Van der Pol oscillator. The latter famous example has a non-linear damping term and for high enough values of the parameter μ, so-called 'relaxation vibrations' occur which show sudden transitions with short moments of high velocity at certain parts of the period. The third example in the right column of Fig.1 shows an undamped oscillation with a non-linear mass term. A standard way of depicting non-linearities is in the phase-plane; a deformed limit cycle is usually a good telltale of non-linear behaviour.
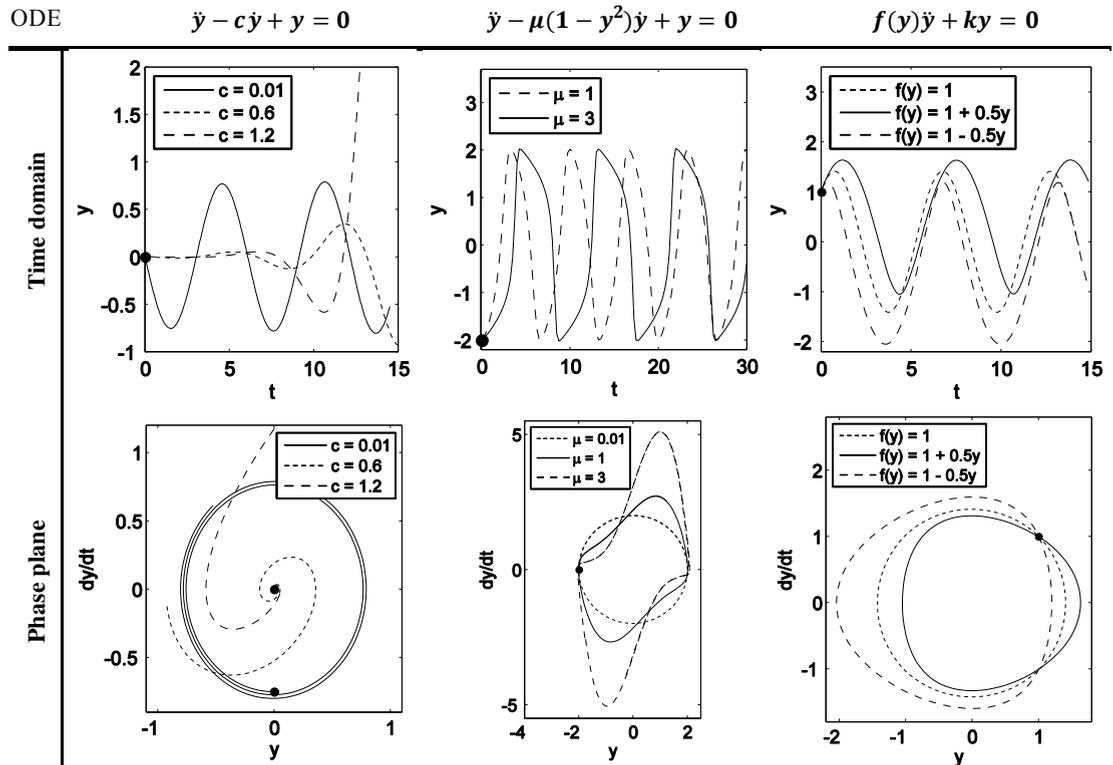
Figure 1: Vibration examples shown in time domain (upper row) and in phase plane (lower row). Left column: constant negative damping; middle column: non-linear damping by Van der Pol oscillator; right column: non-linear mass term. The corresponding ODE equations are written on top; y is the vertical displacement of the mass. The initial states are indicated by thick dots.

Returning to the example of gates of hydraulic structures, identification of negative damping is the engineer's chief concern. Due to structural limitations, however, amplitudes do not grow boundlessly

(Blevins 1990); this gives rise to a certain f(y) in the damping term. Non-linear effects were observed in several laboratory studies of gate vibrations (Billeter & Staubli 2000).

Spectral analyses of gate motion or force signals that are commonly carried out do not detect such non-linearities and, arguably as a result of this, are not good tools for predicting the occurrence of self-excited vibrations with dangerously high dynamic forces on the structure. Furthermore, previous research has shown that the complex interactions between the moving structure and the turbulent flow, as observed in physical scale models, are not fully and readily captured by physics-based numerical simulations (Erdbrink et al. 2014a). Other previous work includes the outline of a data-driven system for classicification and avoidance of gate vibrations by combining spectral analysis and machine learning (Erdbrink et al. 2013) and multi-scale simulations of flow impact on hydraulic structures (Erdbrink el al. 2014b).

The aim of our present study is to devise and test a computational method that solves the inverse problem of identifying self-excited vibrations from only a displacement (output) signal $y(t)$, that is without using a force (input) signal. This has to be done in a way that permits speed-up to practical time frames for real-world operational early-warning systems.

System identification studies embarked on the task of inferring ODE models a few decades ago (Åström & Eykhoff 1971). However, the fixed structure acting as a vehicle for the parameter optimization was usually not an ODE itself, but rather an easy-to-compute basis function like a polynomial. The advent of modern heuristics (Rothlauf 2011) and the steady increase in computing power has enormously boosted possibilities for regression of all kinds (e.g. by artificial neural networks), but many techniques do not provide clear insights into the working of the system. A breakthrough came with the birth of genetic programming (GP) by Koza (1992), which has been applied intensively to system identification problems ever since (Bongard & Lipson 2007). Genetic programming, part of the evolutionary algorithms family, proved to be ideal for evolving the underlying equation structures by means of symbolic regression, e.g. Babovic & Keijzer (2000). Schmidt & Lipson (2009) elegantly demonstrated the power of GP for identifying non-linear dynamical systems by discovering physical laws and functions automatically from experimental data.

In our study it will be assumed that the main part of the structure is already known, this is the basic second order ODE that holds for all vibrations without external forcing. The search for the remaining unknown non-linear terms could benefit from symbolic modelling, but we choose to focus on finding only the coefficients for a number of reasons. First, we are interested in developing a computationally efficient tool for quick assessment that is more easily usable in early-warning systems. Second, in many practical situations, such as the vibrations in this study, only a reduced number of hypotheses for equation structure exist thanks to domain expertise input. The third reason is that in the development of GP, the problem of determining non-trivial numerical constants (i.e. coefficients) is often overlooked. Lastly, the chosen differential evolution (DE) method facilitates a future extension of the algorithm to GP-based system identification, as the algorithmic set-up is similar.

2 Methodology

We use the evolutionary algorithm DE to optimise the coefficients of second-order ODEs with the goal of identifying self-excited and non-linear vibrations. Differential evolution is a competitive derivative-free meta-heuristic global optimization method (Storn & Price 1997). It has a natural robustness that makes it stand out from earlier EC methods, such as genetic algorithms. Its performance has grown by several improvements (Das & Suganthan 2011), most notably the use of dynamic control parameters. We apply a recent version of DE by Choi et al. (2013) that has self-adaptive control parameters which are varied by drawing from the Cauchy distribution. The pseudo-code of the algorithm is given below.

Initialization

-Initialize population of NP vector individuals $X_{1,G}, \dots , X_{NP,G}$ where $X_{i,G} = [x_{1,i,G}, x_{2,i,G}, \dots , x_{Cj,i,G}]$ and where C is the number of coefficients that are being evolved, G the generation ($G = 0, \dots , G_{max}$). Entries $x_{ji,0}$ are uniformly random from [-1,1] for $i = 1, \dots , NP$ and $j = 1, \dots , C$.

-Initialize control parameters $CR_{i,0} = 0.25$, $F_{i,0} = 0.6$ (acc. to Choi et al. 2013) and adaptation parameters $CR_{avg,0} = CR_{i,0}$ and $F_{avg,0} = F_{i,0}$

-Generate target data $y(t)$, $y'(t)$ and divide into training and test sets.

FOR $R = 1$ to R_{max} DO   % run loop
  FOR $G = 1$ to G_{max} DO   % generation loop
    FOR $i = 1$ to NP DO   % individuals loop
     Main loop: Differential Evolution
     -Determine fitness $f(X_{i,G})$ of individuals (see routine)
     -Mutation: generate a mutant vector $V_{i,G} = X_{r1} + F_{i,G}*(X_{r2} - X_{r3})$ from three donor vectors X_{r1}, X_{r2}, X_{r3} randomly selected from the individuals of generation $G-1$.
     -Crossover: generate a trial vector $U_{i,G}$ composed of $u_{j,i,G}$ ($j = 1, \dots , C$) by applying the rule IF rand[0,1] $\leq CR_{i,G-1}$ OR $j = j_{rand}$ THEN $u_{j,i,G} = v_{j,i,G}$, ELSE $u_{j,i,G} = x_{j,i,G-1}$, where j_{rand} is a random integer $1 \leq j_{rand} \leq C$ and $v_{j,i,G}$ is an entry of $V_{i,G}$.
     -Selection: determine fitness $f(U_{i,G})$ of trial vectors. IF $f(U_{i,G}) \geq f(X_{i,G-1})$ THEN $X_{i,G} = U_{i,G}$ and inherit associated fitness and control parameters, ELSE $X_{i,G} = X_{i,G-1}$ and leave fitness and control parameters unchanged.
    END FOR
    -Update control parameters $F_{i,G}$ and $CR_{i,G}$ by adding a randomly drawn number from the Cauchy distribution $Ca(0,0.1)$ to the mean value of the control parameters of all successfully evolved vectors. Truncate if necessary.
    -Replace a non-fittest individual by a newly generated individual.
  END FOR
  Post analysis
  Compute test error of run R by solving the ODE with the winning set of coefficients and determining the mean absolute error of all predicted values compared to the test values.
END FOR
Compute mean duration and mean and min of test errors of all runs $R \dots R_{max}$.

Fitness computation

-Insert the coefficients of each candidate vector in the fixed, assumed ODE equation structure.

-Apply Runge-Kutta, with adaptive step-size and predetermined relative error tolerance for numerical integration.

-Fitness := -1*MAE*penalty, where MAE is the mean absolute error of the training data compared to the result from solving the ODE with candidate coefficients. The penalty punishes candidate models for which the integration failed to determine values at all training times, penalty := ((size of training set – size of candidate set) / size of candidate set)*100 and penalty = 1 if the integration was completely successful.

The generated synthetic data set is randomly divided into a training set and a test set, based on a chosen percentage of data to be used for training. After the evolution has ended, the unseen target points are used to quantify the predictive power of the candidate model by computing a test error.

In order to compare the test errors of different target data sets, we normalize the mean absolute test error (MAE) as follows:

$$NMAE(\hat{\boldsymbol{y}}, \boldsymbol{y}) := \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - \bar{y}}{\sigma_y} - \frac{y_i - \bar{y}}{\sigma_y}\right| = \frac{1}{n}\frac{1}{\sigma_y}\sum_{i=1}^{n}|\hat{y}_i - y_i| = \frac{MAE(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\sigma_y}, \tag{1}$$

where $\hat{\boldsymbol{y}}$ contains the predictions and $\boldsymbol{y}$ the target values for testing, $\bar{y}$ is the mean of $\boldsymbol{y}$ and σ_y is the standard deviation of $\boldsymbol{y}$. All results in the figures and in the table of this paper are given as normalized mean absolute test errors (NMAE).

The computations were done unparallellized on a single Intel i7 processor, 2.93 GHz, 8 Gb RAM.

3 Numerical experiments set-up

The numerical experiments consist of three parts: a validation case, the self-excited cases and a sensitivity analysis. The results are reported in Sections 4.1 to 4.3.

The case of forced vibrations for a linear system with constant coefficients is used for validation:

$$m\ddot{y} + c\dot{y} + ky = F_0 \sin(\omega t), \tag{2}$$

where y is the displacement, t is time (the independent variable) and all other symbols are physical constants. Newtonian notation is used for time derivatives. Together with the real-valued initial conditions y_0 and $\dot{y}_0$, equation 2 constitutes an initial value problem that will be solved in two ways: (i) non-linear regression on the analytical solution and (ii) regression on a fixed ODE structure:

The first approach uses the sum of the general and particular solution of forced vibration with viscous damping as an assumed equation structure:

$$y(t) = C_1 e^{C_2 t} \sin(C_3 t + C_4) + C_5 \sin(C_6 t + C_7), \ \ C_i \in \mathbb{R} \tag{3}$$

The second approach stays at the level of ODE:

$$C_1'\ddot{y} + C_2'\dot{y} + C_3'y = C_4'\sin(C_5't), \quad y_0 = C_6', \quad \dot{y}_0 = C_7', \ C_i' \in \mathbb{R} \tag{4}$$

For both approaches, the coefficients are initialized randomly between -1 and 1. They are stored in a seven-dimensional vector and optimized via DE, as described in the pseudo-code in Section 2. A variation of the second approach where only five coefficients are evolved is also considered, where the initial conditions (IC) are assumed known.

Practically all non-linear vibration problems defy full analytical treatment, so there is no closed-form equation available for $y(t)$. For these problems we work with the ODE structure $m\ddot{y} + c\dot{y} + ky = 0$, where mass m or damping c are replaced by a first or second order polynomial term in y to account for the non-linearity. The results of this are summarised in Section 4.2. Also, a non-linear mass system and a system with time-varying stiffness (Mathieu equation) are examined. These are all unforced oscillators where chaos does not play a role.

4 Results

4.1 Validation: Forced vibrations with constant coefficients

The following target function is defined for the validation runs:

$$2.5\ddot{y} + 0.35\dot{y} + 0.3y = 3.0 \sin(2.2t), \text{ with initial conditions } y_0 = 0, \ \dot{y}_0 = 0.25. \tag{5}$$

Fig.2 (left) shows the sampled target data-set and an example of a candidate solution. The total data set consists of 465 points, meets the Nyquist criterion, and is split in training and test data in different ratios (in Fig.2. half of the data are training points, and half are test points). Population size was set at 80, and 250 generations were computed.
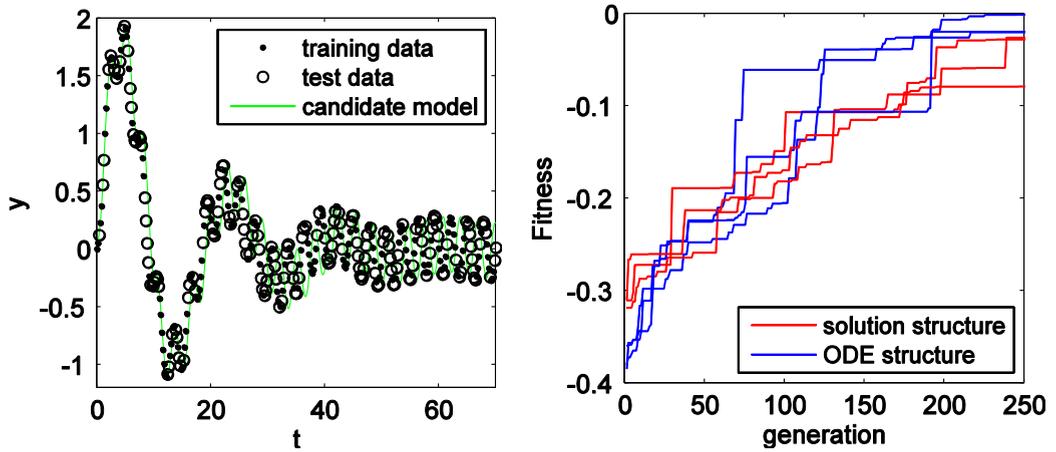
Figure 2: Differential Evolution applied to regression on the signal of a forced vibration. Left: the target data. In this example 50% of the data is used for training. Right: The best-so-far fitness of three runs, for an assumed structure of the analytical solution and for the ODE structure.

The right plot in Fig.2. shows three examples of how the solutions improved with generations. The two applied expression structures were laid out in the previous section, the only necessary addition is that the fitness evaluation of the analytical solution structure runs differs from the pseudo-code in Section 2 because there is no need to solve an ODE; the candidate values follow right away after substitution in the assumed $y(t)$ expression.

The resulting coefficients reflect the multimodality of this optimization problem, since for example $\sin(t) = \sin(t+2\pi)$. It was generally found that the less successful computed functions capture the low-frequency damped free vibration quite well, but give a rather poor estimate of the forced vibration.

Figure 3 below gives an overview of the results based on 10 runs per plotted point. The plot on the left gives test errors expressed as NMAE, according to equation 1. The plot on the right shows the average runtime in seconds.
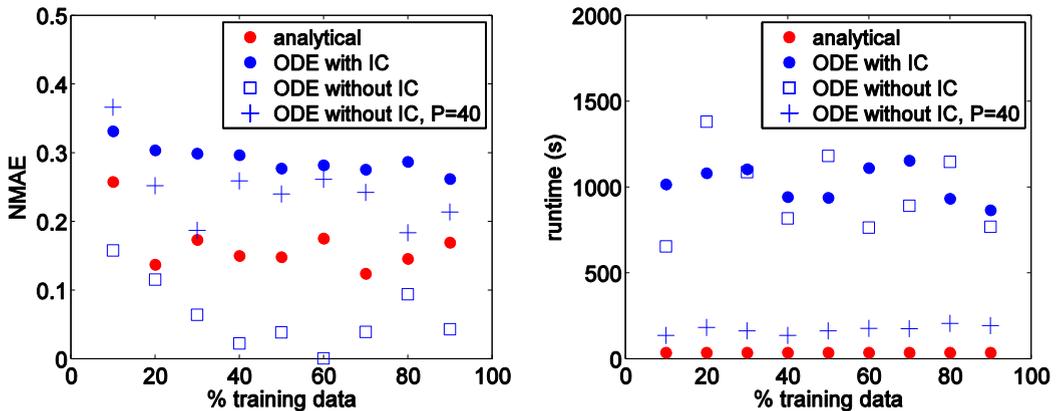

Figure 3: Results of evolving the forced vibration based on analytical solution structure and ODE structure as a function of the percentage of data used for training. Left: test errors (NMAE); right: computing time. IC stands for initial conditions, P stands for population size. Every point represents an average of 10 runs, for an evolution of 250 generations with a population of 80; except for the ODE with initial conditions, for which only five lengthy runs were made for each training data set.

The plots show that the analytical structure requires the least computation time, but it is significantly less accurate than the ODE structure with five evolved coefficients where the initial conditions are known ("ODE without IC"). The analytical structure is more accurate than the ODE case that also evolves the two initial conditions ("ODE with IC"). An attempt to reduce the computation time for the ODE structure by using a smaller population of 40 ("ODE without IC, P=40") resulted in higher test errors and computation times comparable to the analytical runs. The results show that including or excluding the two initial conditions makes no difference for computation time. Additionally, the validation proves that there is little overall dependence on the percentage of data used for training. The test errors are only slightly worse when less than 40% of the data is used for training. Computational factors related to the convergence of the DE algorithm and the ODE solution process are apparently dominant. In particular, it was found that the settings of relative tolerance that determine the number of iterations of the ODE solver during the error computation have a profound influence on the standard deviation of the achieved total runtimes.

4.2   Self-excited vibrations

Table 1 shows the results of computing coefficients for various self-excited vibrations. All computations had a population of 50 individuals with 120 generations computed, and used 50% of 500 data points for training, and the remainder for testing. For each case the mean and standard deviation of the NMAE test error is computed over 25 evolutionary runs.

Vibration	Target ODE	#C	Average of NMAE	Standard deviation of NMAE
linear, constant coeff.*	$2.5\ddot{y} + 0.35\dot{y} + 0.3y = 3.0\sin(2.2t)$	5	$39.2 \cdot 10^{-3}$	$69.5 \cdot 10^{-3}$
Negative damping	$\ddot{y} - 0.1434\dot{y} + y = 0$	1	$3.44 \cdot 10^{-3}$	$0.89 \cdot 10^{-18}$
		2	$3.07 \cdot 10^{-3}$	$1.77 \cdot 10^{-18}$
		3**	$3.23 \cdot 10^{-3}$	$1.88 \cdot 10^{-3}$
Non-linear damping	$\ddot{y} + (-0.450 + 2.728y + 1.903y^2)\dot{y} + y = 0$	3	$287 \cdot 10^{-3}$	$373 \cdot 10^{-3}$
Van der Pol oscillator	$\ddot{y} - 1.2218(1 - y^2)\dot{y} + y = 0$	2	$3.13 \cdot 10^{-3}$	$0.42 \cdot 10^{-3}$
		3	$3.26 \cdot 10^{-3}$	$0.82 \cdot 10^{-3}$
Non-linear mass	$(1.7120 - 1.4815y + 0.4903y^2)\ddot{y} + y = 0$	3	$3.28 \cdot 10^{-3}$	$0.0776 \cdot 10^{-3}$
Mathieu equation***	$\ddot{y} + (0.25 + 0.34\sin(2.18t))y = 0$	3	$70.8 \cdot 10^{-3}$	$25.9 \cdot 10^{-3}$

Table 1: Computation results based on normalized mean absolute error (NMAE). *#C* denotes the number of evolved coefficients. For each case, 25 runs of 120 generations were done with a population size of 50, using 250 training points and 250 test points.

*   Based on the ODE-runs with 50% training data and without evolving initial conditions.
**   Only three runs were made due to poor convergence of ODE solver.
*** The Mathieu equation describes not self-excited but parametrically excited vibrations.

The results in Table 1 show that the constant negative damping and non-linear mass cases have low test errors compared to the validation case of the forced linear vibration with constant coefficients. Moreover, their test errors show very little variation. For the negative damping case, it makes no difference whether the constant coefficient is found using a single coefficient, C_1, or a linear term with two coefficients, $C_1 + C_2 y$, or a second-order polynomial term with three coefficients, $C_1 + C_2 y + C_3 y^2$. Similarly, the Van der Pol oscillator shows a small, insignificant deterioration when a linear term is

added to the $C_1+C_2y^2$ term that is strictly required. The poor result for the non-linear damping case is due to suboptimal convergence of eight runs out of 25. Extending the runs to more generations will most likely improve the mean NMAE. The same can be said of the Mathieu equation, which belongs to a distinctly different class of parametrically excited vibrations.

The test errors of the best runs are plotted in Fig. 4 as function of their computation times. The errors are the minima of the NMAE values of the 25 runs for the vibrations mentioned in Table 1. There are two outliers: the negative damping evolved with a polynomial term took much longer to compute and the best run for the Mathieu equation is significantly less accurate. It is remarkable that the best non-linear damping run is slightly better than the other non-linear cases.
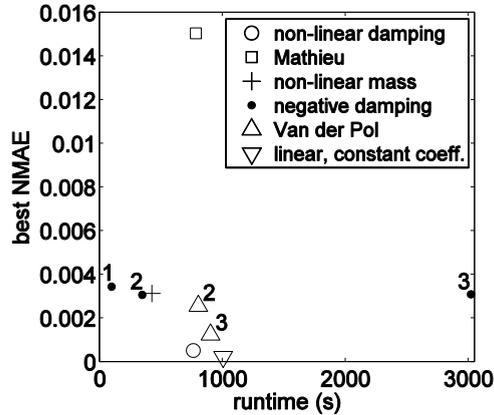

Figure 4: The best achieved test errors (NMAE) out of 25 runs as function of the computing time of the best runs for the cases listed in Table 1. The numbers inside the figure denote the number of evolved coefficients.

4.3   Sensitivity analysis

A sensitivity analysis was done to study the effect of different population sizes, number of generations and tolerance settings of the ODE solver. The results are summarized in Fig. 5. The test errors are NMAE values over 25 runs, as defined in equation 1.
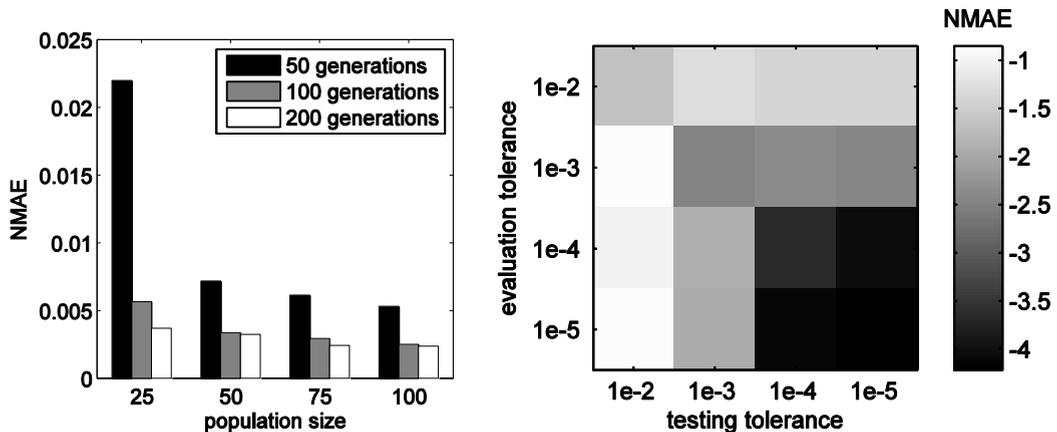

Figure 5: Sensitivity analysis results. Left: sensitivity on population size and number of generations showing normalized mean absolute errors (NMAE) of 25 runs. Right: sensitivity of test error (NMAE) on tolerances of ODE solvers for computing fitness values ("evaluation tolerance") and test errors ("testing tolerance"). On the axes, "1e-2" means 10^{-2}, etc. and the grey scale refers to base-10 logarithms of NMAE values.

The sensitivity analysis is based on the three-dimensional optimization problem of finding the coefficients of an unforced vibration with non-linear damping term $-0.4501+1.0283y+1.903y^2$. The results show that a population size of 50 yields much better results than a population size of 25 (Fig. 5, left), and 100 generations score far better than 50 generations. Further increases in population size and generations give considerably smaller improvements.

The right plot of Fig. 5 shows the effect of different combinations of termination settings for the numerical integration algorithm used for computing errors of the candidate ODE models ("evaluation tolerance") and of the winning model ("testing tolerance"). Unsurprisingly, stricter (lower) tolerances lead to more accurate results ("-4" in the colorbar refers to a NMAE value of 10^{-4}, etc.). The worst results occur for a strict evaluation tolerance in combination with a coarse testing tolerance. Furthermore, it is seen that relatively low test errors are found if the relative tolerance of the ODE solver for evaluation and testing are the same. Of course we need to realize that the lower the evaluation tolerance, the longer on average the runs are likely to be and that the testing tolerance should always be relatively strict in order to make a fair judgment. Based on these observations, a relative evaluation tolerance of 10^{-3} and a testing tolerance of 10^{-5} were chosen for the simulations in Sections 4.1 and 4.2.

5 Discussion

The evaluation accuracy of the candidate models depends on the numerical integration method. Apart from the choice of integration algorithm, the tunable parameter here is the (relative) residual error. A stricter integration error setting results in longer computation time and without further protection even occasionally crashes, but if the ODEs are solved too coarsely, promising candidates may not come out on top. The sensitivity analysis in Section 4.3 can be seen as a preliminary study for this issue. It is not surprising that the impact of the ODE solver settings on the solution accuracy also depends on the complexity of the displacement signal, in particular its degree of non-linearity. Ultimately, the dilemma lies in optimizing the integration scheme for specific problems, or maintaining robustness – at the cost of longer total computation times. Solving this dilemma involves judgment of stiffness of the worst possible candidate (e.g. for high μ Van der Pol cases). See Quateroni et al. (2010) for a treatment of different ODE solvers in MATLAB.

An important observation is that achieving a low test error is not synonymous with finding values close to the original coefficients of the target motion equation. This is due to the multimodality, or one could say ambiguity of the problem. The undamped natural frequency, for instance, is a function of the ratio of stiffness k and mass m, which can be 'found' by innumerable combinations of k and m. Ways to (partly) overcome ambiguous solutions in practice are a priori injection of knowledge about the physical domain and normalization (e.g. it is common to divide all terms by the mass coefficient). In fact, in real-life applications there is usually information available about parameter ranges. Estimates of the stiffness in a physical system, for instance, would lead to a constraint on the stiffness coefficient, transforming the free optimization problem into a constrained optimization problem. The new developments in automated discovery of physical laws do not exempt scientists from their duty of making thorough interpretations of the results (as also noted by Schmidt & Lipson 2009). If this remains an important task anyway, then domain knowledge might just as well be applied beforehand, thus avoiding a search through a multitude of unrealistic candidate solutions.

Other methods that infer ODEs may not suffer from a large computational load of solving candidates, but they often have other disadvantages worth mentioning. Some GP approaches for evolving dynamical systems perform training in the $y'(t)$ domain. This means that the original time series first has to be differentiated fully, which is not a trivial task due to noise present in real-life data (see Kronberger 2011). In this study it was only assumed that the initial values are known, in other words, the velocity is required only at the time of the first training point. Other approaches may use

structures that are easily computable, such as the discrete map used by Howard & Oakley (1994), but may provide insufficient insight in the system because the resulting expression has a format that is not easily interpreted. To be able to write the end result in the familiar ODE form is a necessary condition for serving engineering applications.

6 Conclusions and outlook on future work

The reverse engineering problem of finding the ODE expression of self-excited vibrations appears in innumerous applications. Particularly interesting is the idea of quick assessment of different vibration types. This enables operational and early-warning systems to recognise undesired vibrations in engineering installations before they inflict damage.

In this paper it was examined how the differential evolution algorithm can be applied to identify several vibration types by performing regression on the ODE that describes the displacement. The relatively simple case of a linear forced vibration allowed a comparison between using the analytical solution structure and an ODE structure. Irrespective of the percentage of available training data, the ODE structure produced more accurate results, but required significantly more computational time than the analytical solution, under the condition that the initial velocity is known. Next, a number of self-excited oscillations was identified, yielding reasonably accurate results. The presence of superfluous non-linear terms proved to have an influence on the achieved computation times, but not directly on test errors. A sensitivity analysis exposed the impact of the tolerance settings of the ODE solver.

Ongoing work is looking at ways to use information from the frequency domain of the target data in the population, an idea found in Howard & Oakley (1994). Also, it should be studied more deeply how the degree of non-linearity affects the accuracy. The algorithm could furthermore benefit from fitness prediction as used by Schmidt & Lipson (2009), with the goal of speeding up and improving the evolution. A third point of attention for future work is to see which of the existing 'niching' methods for multimodal optimization are useful for the identification of vibrations. And finally, the algorithm will be applied to the sensor data from experiments with hydraulic gate vibrations.

Acknowledgements

References

Åström, K.J., Eykhoff, P. (1971). *System identification – a survey*. Automatica, Vol.7, pp.123-162.

Babovic, V., Keijzer, M. (2000). *Evolutionary algorithms approach to induction of differential equations*. In: Proc.of the 4th Int. Conf. on Hydroinformatics, Iowa City, USA, pp.251-258.

Billeter, P., Staubli, T. (2000). *Flow-induced multiple-mode vibrations of gates with submerged discharge*. J. of Fluids and Structures, Vol. 14, pg. 323-338.

Blevins, R.D. (1990). *Flow-induced vibration*. 2nd ed. Krieger Publ. Co., Malabar, Florida, USA.

Bongard, J., Lipson, H. (2013). *Automated reverse engineering of nonlinear dynamical systems*. PNAS, Vol.104, No.24, pp.9943-9948.

Choi, T.J., Ahn, C.W., An, J. (2013). *An adaptive Cauchy differential evolution algorithm for global numerical optimization*. The Scientific World Journal, Vol. 2013, Article ID 969734, http://dx.doi.org/10.1155/2013/969734.

Das, S., Suganthan, P.N. (2011). *Differential Evolution: a survey of state-of-the-art*. IEEE Transactions on Evolutionary Computation, Vol.15, No.1, pp.4-12.

Den Hartog, J.P. (1956). *Mechanical vibrations*. Dover Publ., Inc., New York, USA.

Erdbrink, C.D., Krzhizhanovskaya, V.V., Sloot, P.M.A. (2013). *Controlling flow-induced vibrations of flood barrier gates with data-driven and finite-element modelling*. In: Comprehensive Flood Risk Management: Research for Policy and Practice. Klijn & Schweckendiek (eds). CRC Press 2013, ISBN 978-0-415-62144-1, eBook ISBN 9780203374511. http://www.crcpress.com/product/isbn/9780415621441, http://www.amazon.com/Comprehensive-Flood-Risk-Management-ebook/dp/B00CLGI3NW, pp. 425-434.

Erdbrink, C.D., Krzhizhanovskaya, V.V., Sloot, P.M.A. (2014a). *Reducing cross-flow vibrations of underflow gates: experiments and numerical studies*. Submitted to Journal of Fluids and Structures, http://archiv.org/abs/1312.7868.

Erdbrink, C.D., Krzhizhanovskaya,V.V., Sloot, P.M.A. (2014b). *Free-surface flow simulations for discharge-based operation of hydraulic structure gates*. Journal of Hydroinformatics, Vol.16, no.1, pp.189-206, http://dx.doi.org/10.2166/hydro.2013.215, DOI: 10.2166/hydro.2013.215

Howard, E., Oakley, N. (1994). *The application of genetic programming to the investigation of short, noisy, chaotic data series*. In: Evolutionary Computing, Lecture Notes in Computer Science, Vol.865, pp.320-332.

Kolkman, P.A. (1976), *Flow-induced gate vibrations*. PhD thesis Delft University of Technology. Also publication 164 of Delft Hydraulics Laboratory (at present Deltares).

Koza, J.R. (1992). *Genetic Programming: on the programming of computers by means of natural selection*. MIT Press, Cambridge, MA.

Kronberger, G. (2011). Symbolic regression for knowledge discovery. PhD thesis Johannes Kepler university, Linz, Austria. Trauner Verlag, Linz, Autria.

Naudascher, E., Rockwell, D. (1994). *Flow-induced vibrations – an engineering guide*. Dover publications, New York.

Quateroni, A., Saleri, F., Gervasio, P. (2010). *Scientific computing with MATLAB and Octave*. 3rd ed., Springer-Verlag Berlin Heidelberg.

Rothlauf, F. (2011). *Design of modern heuristics*. Natural computing series, Springer-Verlag Berlin Heidelberg.

Schmidt, M., Lipson, H. (2009). *Distilling free-form natural laws from experimental data*. Science, Vol.324, no. 5923, pp.81-85, DOI: 10.1126/science.1165893.

Storn, R., Price, K. (1997). *Differential Evolution – A simple and efficient heuristic for global optimization over continuous spaces*. J. of Global Optimization, Vol. 11, No. 4, pp. 341-359.

Verhulst, F. (1996). *Nonlinear differential equations and dynamical systems*. 2nd ed. Springer-Verlag Berlin Heidelberg.