Instantaneous vehicle fuel consumption estimation using smartphones and Recurrent Neural Networks

Kanarachos, S., Mathew, J. & Fitzpatrick, M.

Original citation & hyperlink:
Kanarachos, S, Mathew, J & Fitzpatrick, M 2019, 'Instantaneous vehicle fuel consumption estimation using smartphones and Recurrent Neural Networks' *Expert Systems with Applications*, vol. 120, pp. 436–447.
https://dx.doi.org/10.1016/j.eswa.2018.12.006

DOI 10.1016/j.eswa.2018.12.006
ISSN 0957-4174

Publisher: Elsevier

Title:
Instantaneous vehicle fuel consumption estimation using smartphones and Recurrent Neural Networks

Abstract:

The high level of air pollution in urban areas, caused in no small extent by road transport, requires the implementation of continuous and accurate monitoring techniques if emissions are to be minimised. The primary motivation for this paper is to enable fine spatiotemporal monitoring based on crowd sensing, whereby the instantaneous fuel consumption of a vehicle is estimated using smartphone measurements. To this end, a surrogate method based on indirect monitoring using Recurrent Neural Networks (RNNs) that process a smartphone's GPS position, speed, altitude, acceleration and number of visible satellites is proposed. Extensive field trials were conducted to gather smartphone and fuel consumption data at a wide range of driving conditions. Two different RNN types were explored, and a parametric analysis was performed to define a suitable architecture. Various training methods for tuning the RNN were evaluated based on performance and computational burden. The resulting estimator was compared with others found in the literature, and the results confirm its superior performance. The potential impact of the proposed method is noteworthy as it can facilitate accurate monitoring of in-use vehicle fuel consumption and emissions at large scales by exploiting available smartphone measurements.

Stratis Kanarachos, Jino Mathew, Michael E. Fitzpatrick

stratis.kanarachos@coventry.ac.uk, ab9932@coventry.ac.uk, ab6856@coventry.ac.uk, Faculty of Engineering, Environment and Computing, Coventry University, Priory Street, Coventry CV1 5FB, United Kingdom

Corresponding author: *Stratis Kanarachos, stratis.kanarachos@coventry.ac.uk Tel: +44(0)2477657720, Engineering & Computing Building - EC 4-07, Faculty of Engineering, Environment and Computing, Coventry University, 3, Gulson Road, Coventry, CV1 2JH*

Keywords:
Recurrent Neural Networks, vehicle fuel consumption, smartphone data, training, optimisation, soft sensor

1. Introduction

Road transport contributes about one-fifth of the world's total emissions of carbon dioxide (CO_2) and is the only primary sector in the EU where greenhouse gas emissions are still rising[1,2]. In the UK specifically, transport has become the most polluting sector[3]. Cities are required to monitor emissions and implement short- and long-term mitigation measures to avoid pollution episodes (Dey *et al*., 2017). For example, the Low Emission Zone operates to minimise the use of the most-polluting heavy diesel vehicles driving in London[4].

Researchers argue about the need to monitor emissions at a high spatiotemporal resolution (Madrazo and Clappier, 2018, Sun *et al*., 2018b). To this end, some methods combine sporadic air pollution sampling with dense traffic flow or vehicle count measurements (Zaldei *et al*., 2017, Forehead and Huynh, 2018). The latter predicts emissions by extrapolating the officially reported CO_2 values of the vehicles. However, this is not an accurate monitoring method as there is a difference of 30-40% between theoretical values and real-world emissions (Fontaras, Zacharo and Ciuffo, 2017). The discrepancy is mainly due to the mismatch between real driving behaviour and that assumed in the test protocols.

In-use monitoring using instrumented vehicles can alleviate this problem (Pucher, 2016). However, it is very difficult to scale up this approach (Boer, 2012). An alternative way, which can be scaled up, is to use indirect monitoring methods based on fuel consumption models. Different types of model have been used in the literature (Zhou *et al*., 2018). White-box models are detailed physical models usually developed by car or engine manufacturers (Rajamani, 2014). They are highly accurate and transparent. Nevertheless, they require detailed information that is generally not available such as engine friction or pumping losses. Grey-box models combine simple physical models and data obtained from controlled experiments. The most popular grey-box model is the Comprehensive Modal Emissions Model (CMEM), widely used in traffic-simulation. The accuracy of CMEM depends mainly on the vehicle

[1] https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-greenhouse-gases/transport-emissions-of-greenhouse-gases-10
[2] https://data.worldbank.org/indicator/EN.CO2.TRAN.ZS
[3] https://www.independent.co.uk/environment/air-pollution-uk-transport-most-polluting-sector-greenhouse-gas-emissions-drop-carbon-dioxide-a8196866.html
[4] https://tfl.gov.uk/modes/driving/low-emission-zone

speed and data fidelity. For example, when average speed profiles are used, the fuel consumption estimate can be less than half of the actual value (Turkensteen, 2017).

Black box models are based only on data, usually obtained from naturalistic driving trials. The models are clustered into average and instantaneous fuel consumption models. Wu and Liu (2011, 2012) investigated the performance of Back-Propagation (BP) neural networks (NNs) and Radial Basis (RB) neural networks for predicting average fuel consumption. Both NNs used the make of car, weight of the vehicle, engine style, vehicle and transmission type as input variables. The BP-NN estimated fuel consumption with an accuracy between 93-98.2%, and the RB-NN between 97.7-98.2%. In the same category, Masikos *et al*. (2014) studied a General Regression NN for the energy consumption of an electric vehicle. The model performed with an accuracy of 96% using the inputs: the battery's states of health and charge, the use of auxiliary equipment, the vehicle's weight, the day of the week, the month, the hour band of the day, the slope and class of the road segment, the ambient temperature and relative humidity, and the driver's average energy consumption rate. Yamashita *et al*. (2018) employed a BP-NN depending on the vehicle position, driving time, speed, tri-axial accelerations, angular velocity, engine revolutions, throttle position and throttle nozzle. The accuracy of the model was approximately 95%. Vilaca *et al*. (2015) compared different machine learning algorithms. Inputs to the models were the GPS-derived speed, vehicle acceleration, and road inclination. Vehicle acceleration was calculated indirectly from the GPS speed. The best performing algorithm was the Boosted Trees methods. An NN model with only one hidden layer presented the worst performance. Zeng, Miwa and Morikawa (2015) studied the use of Support Vector Machines (SVM). The inputs to the SVM were the trip distance, average travel speed, intersection density (number of intersections per km), engine displacement, and the coefficient of variance of speed. The correlation achieved with the SVM was 0.92, compared to only 0.86 for a BP-NN with one hidden layer. Du *et al*. (2017) explored a BP-NN model with the driver's gender, age, the vehicle's transmission type, fuel type, weight, mileage, speed, time, and location as inputs. The NN comprised one hidden layer and 13 neurons. The fuel prediction accuracy was 0.82. Table 1 compares different NN-based fuel consumption models found in the literature.

Table 1: Comparison of Neural Network– based fuel consumption models

	Spatio-temporal scale	NN type	Inputs			Performance
			Velocity data	CAN-bus data	GPS data	
Wu and Liu, 2011	Low	BP	×	×	×	MPE: 1.8-7%
Wu and Liu, 2012	Low	RB	×	×	×	MPE: 1.8-2.3%
Masikos *et al*., 2014	Low	GR	×	✓	×	MAPE: 3.96%
Yamashita *et al*., 2018	Low	N/A	✓	✓	×	MAPE: 95%
Zeng, Miwa and Morikawa, 2015	Low	BP	✓	✓	✓	R: 0.92
Vilaca *et al*., 2015	Fine	N/A	×	×	✓	RMSE 3-13 L/km
Du *et al*., 2017	Fine	BP	✓	✓	✓	R: 0.82

GR: General Regression, BP: Back-propagation, RB: Radial basis, MPE: Mean percentage error, MAPE: Mean Absolute Percentage Error, RMSE: Root mean squared error, R: correlation, N/A: not available, L/km: Litres per km

Smartphone data have been utilised for eco-driving purposes already (Kanarachos, Christopoulos and Chroneos, 2018). However, the focus was not on the instantaneous fuel consumption but the classification and improvement of driver behaviour (Meseguer *et al*., 2013; Orfila *et al*., 2015; Chen *et al*., 2017). This paper proposes the estimation of a vehicle's instantaneous fuel consumption using smartphone measurements. The potential impact is noteworthy as it can be scaled up and thus facilitate accurate monitoring of in-use vehicle fuel consumption and emissions at fine spatiotemporal accuracy. The novelty of the findings reported in the manuscript can be summarised as:

- The use of a Deep Neural Network model, in particular, a Recurrent Neural Network (RNN), for the estimation of the instantaneous fuel consumption.
- Smartphone accelerometer data and number of visible GPS satellites as inputs to the DNN-based soft sensor.
- A thorough investigation and comparison of different NN architectures and training methods for tuning the soft sensor.
- The derivation of a probabilistic criterion based on Bayesian binomial inference technique for switching between global and local training methods.
- An appreciation of performance loss when less information-rich inputs are utilised.

The rest of the paper is structured as follows: in Section 2 the challenges for tuning NN-based soft sensors are reviewed. Section 3 presents the steps for designing the soft sensor. Section 4 provides a comprehensive evaluation of the soft sensor for different training methods and inputs. Conclusions and future research directions are given in Section 5.

2. NN-based soft sensors design – Related literature

When it is impossible or impractical to build a sensor for measuring a quantity of interest, a soft sensor is often employed. Soft sensors estimate a feature by combining a system model with other physical measurements. NNs are often utilised to develop the system model, and RNNs are particularly suited for noisy dynamical systems. An increasing number of NN-based soft sensors have been developed for the automotive industry (Arsie, Pianese & Sorrentino, 2010; Capriglione *et al*., 2016; Acosta and Kanarachos, 2017; Acosta *et al*., 2017; Nweke *et al*., 2018).The most common RNN soft sensor types are the Nonlinear Autoregressive Networks with exogenous inputs (NARX-RNN), Long-Short Term Memory NNs (LSTM-NNs), and Echo-State Networks (Ferreira *et al*., 2012). Bianchi (2017) provided a benchmark study considering a set of test functions. A soft sensor's performance depends heavily on its architecture, which is problem-specific, and the training method (Pascanu, Mikolov & Bengio, 2013; Ferreira *et al.*, 2018).

The most popular method for training NNs is Back-Propagation (BP). However, BP often converges to local minima and presents poor performance. An alternative is to

have population-based optimisation algorithms. For example, Ibrahim and El-Amary (2017) employed Particle Swarm Optimisation (PSO) to train an RNN soft sensor for voltage instability and improved the estimation performance by 2.4% compared to BP. In another RNN soft sensing application concerning a manufacturing process, Patel *et al*. (2017) employed a Genetic Algorithm (GA) and reduced the mean squared error (MSE) by 11% compared to BP. Differential Evolution (DE) was used by Duchanoy *et al*. (2017) to train an RNN sensor that estimates the tyre-contact-patch area for a vehicle. Remarkably, population-based algorithms have a significantly higher computational burden compared to gradient-based algorithms (Piotrowski *et al*., 2014).

Additionally, in many cases when the ratio of the NN size to the population size is high, standard population-based algorithms perform worse than gradient-based ones (Piotrowski *et al*., 2016). This is due to the lack of exploitation capability of population-based algorithms in high-dimensional spaces. Unique large-scale algorithms have been developed for solving such high-dimensional problems; however, their performance in training NNs and specifically NN-based soft sensors has not been investigated (Ismkhan, 2017; Nakib *et al*., 2017; Peng & Wu, 2017; Gao *et al*., 2018; Sun *et al*., 2018a). Large-scale optimisation performance is usually evaluated based on $3.0 \cdot 10^6$ function evaluations, which is considerably higher than the few thousand typically applied in soft sensor training (Piotrowski and Napiorkowski, 2018). Table 2 summarises the strengths and weaknesses of NN-based soft sensor training algorithms.

Table 2: Qualitative comparison of NN training algorithms considered in this paper

	Global optimisers	High-dimensional problems	Computational load
Gradient-based algorithms	×	✓	✓
Population-based algorithms(standard version)	✓	×	×
Large-scale optimisation algorithms	✓	✓	×

3. Fuel consumption soft sensor design

3.1 Fuel consumption soft sensor: Experimental data & inputs

Road grade, driving style, and start-stop conditions are the principal factors increasing fuel consumption, by up to 20%, 25% and 40% respectively (Fontaras, 2017). Kumar *et al*. (2016) highlighted the relative importance of acceleration. Modern smartphones have embedded sensors that measure a vehicle's position, velocity, acceleration, as well as altitude, and therefore can infer the above critical parameters. Nevertheless, some of the smartphone signals are noisy, and others can present significant errors. For example, the GPS position can have errors up to 75 m, when contact with satellites is lost. Consequently, the uncertainty and discrepancies influence the estimation accuracy negatively compared to other data sources like the Controller Area Network (CAN).

To gather the data a large number of field trials was conducted in Coventry, UK. Different driving styles were exhibited to produce an information-rich dataset. We recorded the time, GPS position (latitude, longitude, altitude), speed, acceleration (longitudinal, lateral, vertical), and the number of visible satellites during the trials. The sampling rate was 1 Hz. The data were acquired using an off-the-shelf smartphone application (AndroSensor). The CAN-bus of the vehicle through the OBDII port provided the fuel consumption data. The soft sensor was trained using a dataset comprising 3693 samples.

3.2 Fuel consumption soft sensor: Type and architecture

To select a suitable NN type a comparison between the nonlinear autoregressive network with exogenous inputs (NARX-RNNs) and Long-Short-Term-Memory NN (LSTM-NN) was conducted. Since a suitable NN architecture is not known *a priori*, a parametric analysis was performed.

3.2.1 NARX-RNN-based fuel consumption soft sensor

The NARX-RNN soft sensor architecture employed in this study is shown in Figure 1. The hidden layer includes a log-sigmoid activation function, and the output layer a linear one. The number of lags was varied between one and six. The number of neurons in each layer was in the range 27-62, in increments of five. The NARX-RNN was trained using the Levenberg-Marquardt (LM) method. To obtain statistically meaningful results each configuration was trained thirty independent times. The training termination criterion was 1000 function evaluations.

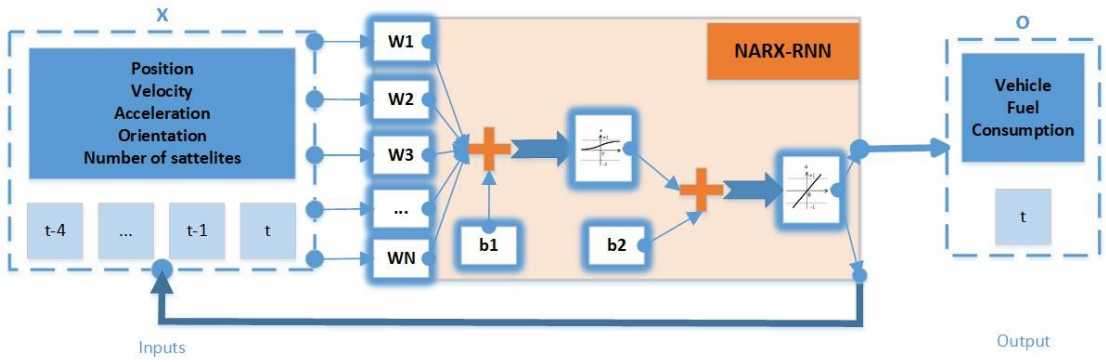


Figure 1: The proposed nonlinear autoregressive neural network with exogenous inputs (NARX-RNN) architecture for estimating the instantaneous fuel consumption

The output $o_{NN,t}$ of the NARX-RNN is a function of the inputs $[\mathbf{x_t}, \dots, \mathbf{x_{t-N}}]$, outputs $[o_{NN,t-1}, \dots, o_{NN,t-N}]$, the neural network weights $\mathbf{W} = [\mathbf{W_1}, \mathbf{W_2}, \dots, \mathbf{W_N}] = [\dots, W_i, \dots]$, the nonlinear activation function $\sigma = 1/(1 + e^{-a})$and biases $\mathbf{b_1}$ and $\mathbf{b_2}$:

$$o_{NN,t} = f(\mathbf{x_t}, \mathbf{x_{t-1}}, \dots, \mathbf{x_{t-N}}, o_{NN,t-1}, \dots, o_{NN,t-N}, \mathbf{W}, \mathbf{b_1}, \mathbf{b_2}) \quad (1)$$

where N is the lag. The Mean Squared Error e_{fuel} is the performance metric for the NARX-RNN:

$$e_{fuel} = \frac{1}{N_s}\sum_{i=1}^{N_s}(\mathbf{o_{meas}} - \mathbf{o_{NN}})^2 \quad (2)$$

where $\mathbf{o_{meas}} = [\dots, o_{meas,t-1}, o_{meas,t}, o_{meas,t+1}, \dots]$ is fuel consumption measurements vector and $\mathbf{o_{NN}} = [\dots, o_{NN,t-1}, o_{NN,t}, o_{NN,t+1}, \dots]$ is the estimated one. Figure 2 illustrates the results of the parametric analysis. There is an area, lags between 2-5 and hidden neurons in the range [32-47], where similar results were

produced. The minimum error value of $e_{fuel\_min} = 0.66$ was obtained for five delay inputs and 47 neurons in each hidden layer.

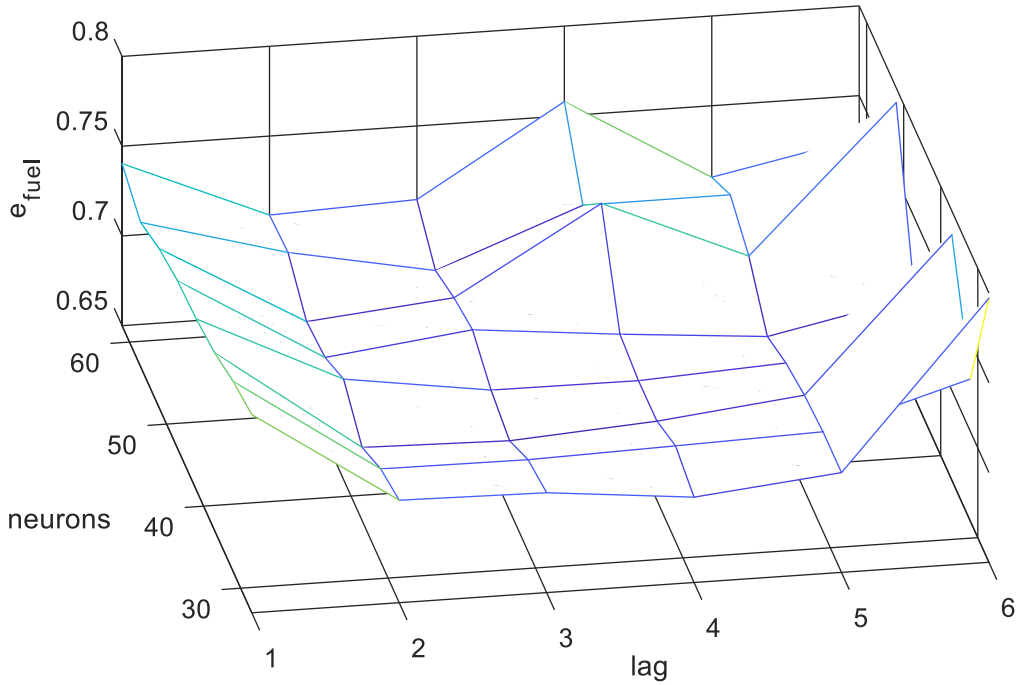


Figure 2: Mean squared error results e_{fuel} in kpl (kilometres per litre) for the NARX-RNN soft sensor using the Levenberg-Marquardt training algorithm. Parametric analysis for different number of time lags (1-6) and number of hidden neurons in each layer (27-62).

3.2.2 LSTM-NN-based fuel consumption soft sensor

The LSTM-based soft sensor comprised one hidden layer, followed by a regression layer. A parametric analysis with 100, 200, 300, 400, 500 hidden units was conducted. The stochastic gradient descent method (SGD) and Adam optimiser (ADAM) (Pascanu *et al*., 2014, Kingma *et al*., 2014) were applied to train the LSTM. Thirty (30) independent runs were conducted to obtain statistically meaningful results. The same objective function e_{fuel} as with NARX – RNNs was employed. The training termination criterion was 6000 iterations. Table 1 lists the results of the parametric analysis. As observed, the best performance with SGD is obtained for 50 hidden units, while with ADAM optimiser the best performance is for 400 and 500 units. The NARX-RNN seems to perform better compared to LSTM, as the minimum error value obtained using the latter was $e_{fuel\_min} = 0.97$.

Table 1: Mean squared error results $\bar{e}_{fuel}$ in kpl (kilometres per litre) for the LSTM-NN. Parametric analysis for different number of hidden units 50-700 and two different training methods: stochastic gradient descent method (SGD) and ADAM optimiser.

$e_{fuel}\ [kpl]$		
Units	SGD	ADAM
50	1.78	1.17
100	1.88	1.07
200	1.97	1.00
300	1.99	1.03
400	2.00	0.97
500	2.04	0.97
600	2.03	0.99
700	2.02	1.01

3.3 Fuel consumption soft sensor: Tuning

Since it is unknown *a priori* which method tunes the soft sensor better, a range of training algorithms were explored to select the most appropriate one. These can be clustered into three main categories: gradient-based, population-based, and large-scale population-based optimisation algorithms.

In the first category, the popular Levenberg–Marquardt (LM) and Bayesian Regularisation Back-Propagation algorithms were included. From the second category the standard Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO), the contrast-based Fruit Fly Optimisation (c-FOA), the c-FOA with group policy (c-FOA/g), the cloud model-based FOA (CMFOA), and the chaotic Fruit Fly Optimisation algorithm (DLSC-FOA) were studied. CMFOA has been shown to obtain better or competitive performance for most test functions compared with three improved FOAs and seven state-of-the-art intelligent optimization algorithms (Wu, Zuo and Zhang, 2015). DLSC-FOA has been shown to have a competitive performance compared to GA, PSO and other FOA algorithms (Du *et al*., 2018). The c-FOA version implemented in this paper is from Kanarachos *et al*. (2018), while a description of c-FOA/g is in Appendix A. From the third category we explored the

SHADE algorithm (Tanabe and Fukunaga, 2013), LSHADE (Tanabe and Fukunaga, 2014), LSHADE44 (Polakova, 2017), the Enhanced Unidimensional Search and adaptive Enhanced Unidimensional Search algorithms (Gardeux *et al*., 2017), Differential Search Algorithm (Civicioglu, 2012) and Self-adaptive Differential Evolution with Multi-trajectory Search (Zhao *et al*., 2010).

Training was completed for a fixed number of 6000 function evaluations. The threshold was selected based on the number of function evaluations required for gradient-based algorithms to converge, as well as considering the number of function evaluations reported in other scientific contributions (Ibrahim and El-Amary, 2017; Patel *et al*., 2017; Duchanoy *et al*., 2017). The same criterion was applied to all algorithms to allow a fair comparison.

4. Results & discussion

4.1 Soft sensor tuning using gradient-based, population-based and large-scale algorithms

The NN was trained using a dataset comprising 3693 samples. To avoid overfitting, 70% of the samples were used for training, 15% of the samples for testing, and 15% of the samples for validation. Furthermore, the number of hidden units was iteratively optimised. The gradient-based algorithms were terminated for 1000 epochs. For the population-based algorithms, 85% of the data were used for training, and 15% of the data for testing. They were terminated for 6000 function evaluations. The NN weights W_i were bounded in the range $W_i \in [-10,10]$. The NN weights were initialised randomly. The population size was $N = 100$ for all large-scale algorithms following Piotrowski (2018). GA, PSO, CMFOA and DLSC-FOA had the same population size. The population size for c-FOA/g was N=10, equal to the number of groups. For comparison reasons, we selected the same population size for c-FOA. Each training algorithm was run 30 independent times to obtain statistically meaningful results.

Table 3 lists the results for the mean squared error e_{fuel} in kilometres per litre $[kpl]$. They are the mean value μ, standard deviation σ, median value $\widetilde{D^*}$ and minimum value min of e_{fuel} for the thirty independent runs. The results presented consider the

training, testing and validation data. Bayesian-Regularisation achieved the best performance in the gradient-based group, c-FOA/g in the group of population-based algorithms and LSHADE44 in the group of large-scale algorithms. It is known that Bayesian-Regularisation reduces overfitting (Cawley & Talbot, 2007).

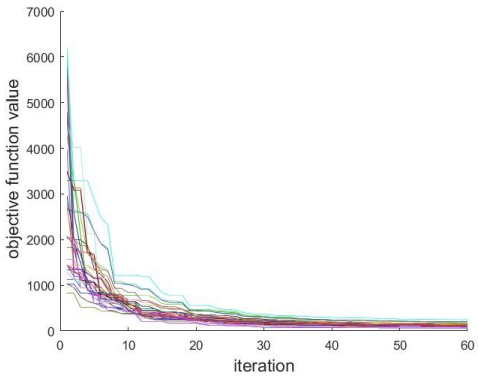
Table 3. Training results obtained following 30 independent runs. e_{fuel} is the minimum mean squared error value e_{fuel} in kilometres per litre $[kpl]$. The training algorithms studied are: Levenberg–Marquardt Back Propagation (LM-BP), Bayesian Regularisation Back Propagation (BR-BP), Genetic Algorithm (GA), Particle Swarm Optimisation (PSO), contrast-based FOA (c-FOA & c-FOA/g), cloud-based FOA (CM-FOA), chaotic-based-FOA (DLSC-FOA), SHADE, LSHADE, LSHADE44, Enhanced Unidimensional Search (EUS), adaptive Enhanced Unidimensional Search algorithm (aEUS), Differential Search Algorithm (B-DSA), and Self-adaptive Differential Evolution with Multi-trajectory Search (SaDE-MMTS).

e_{fuel} $[kpl]$					
	μ	σ	$\widetilde{D^*}$	min	$\bar{t}\_comp$ / s
LM-BP	$6.9{\cdot}10^{-1}$	$1.5{\cdot}10^{-1}$	$6.5{\cdot}10^{-1}$	$5.8{\cdot}10^{-1}$	54
BR-BP	$6.7{\cdot}10^{-1}$	$2.3{\cdot}10^{-1}$	$5.8{\cdot}10^{-1}$	$4.1{\cdot}10^{-1}$	3932 s
GA	$1.3{\cdot}10^{2}$	$4.7{\cdot}10^{1}$	$1.2{\cdot}10^{2}$	$6.1{\cdot}10^{1}$	3620
PSO	$3.3{\cdot}10^{3}$	$2.1{\cdot}10^{3}$	$2.4{\cdot}10^{3}$	$1.9{\cdot}10^{3}$	4124
c-FOA	$1.5{\cdot}10^{1}$	$1.3{\cdot}10^{1}$	$1.1{\cdot}10^{1}$	2.6	2887
c-FOA/g	1.0	0.2	$9.8{\cdot}10^{-1}$	$9.1{\cdot}10^{-1}$	3835
CM-FOA	$1.7{\cdot}10^{6}$	$3.1{\cdot}10^{5}$	$1.6{\cdot}10^{6}$	$1.1{\cdot}10^{6}$	3181
DLSC-FOA	$3.3{\cdot}10^{5}$	$1.5{\cdot}10^{5}$	$2.5{\cdot}10^{5}$	$1.7{\cdot}10^{5}$	3506
SHADE	$5.4{\cdot}10^{5}$	$9.5{\cdot}10^{4}$	$5.5{\cdot}10^{5}$	$3.4{\cdot}10^{5}$	3539
LSHADE	$4.3{\cdot}10^{4}$	$2.8{\cdot}10^{4}$	$1.1{\cdot}10^{4}$	$3.5{\cdot}10^{4}$	2671
LSHADE 44	$1.3{\cdot}10^{3}$	$3.2{\cdot}10^{2}$	$1.3{\cdot}10^{3}$	$7.8{\cdot}10^{2}$	3195
EUS	$2.1{\cdot}10^{3}$	$1.1{\cdot}10^{3}$	$1.7{\cdot}10^{3}$	$9.6{\cdot}10^{2}$	3986
aEUS	$2.1{\cdot}10^{3}$	$1.1{\cdot}10^{3}$	$1.8{\cdot}10^{3}$	$9.6{\cdot}10^{2}$	4017
B-DSA	$2.2{\cdot}10^{4}$	$7.1{\cdot}10^{3}$	$2.1{\cdot}10^{4}$	$1.4{\cdot}10^{4}$	130
SaDE-MMTS	$4.6{\cdot}10^{3}$	$9.9{\cdot}10^{2}$	$4.9{\cdot}10^{3}$	$3.2{\cdot}10^{3}$	3571

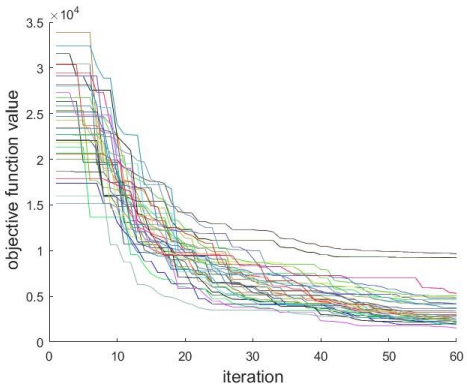
μ: mean value, σ: standard deviation, $\widetilde{D^*}$: median value, min: minimum value, $\bar{t}\_comp$: average computational cost for one solution

The statistical analysis rejected the null hypothesis that BR-BP and c-FOA/g data samples come from the same distribution at a 1% significance level. In particular, the Kruskal-Wallis test returned a value of $p = 3.8 \cdot 10^{-11}$ with $\chi^2 = 43.7$.

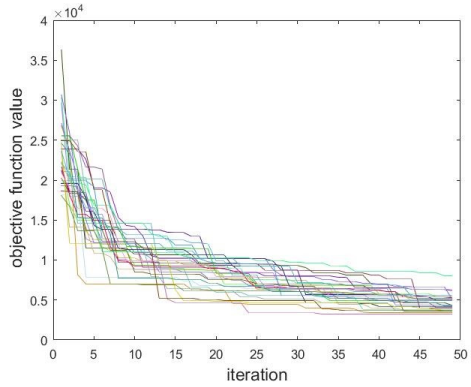
In general, the population-based algorithms produced worse results compared to the gradient-based ones. Furthermore, their computational burden was more significant. Figure 3 illustrates the convergence diagrams for the best-performing population-based algorithms: GA, PSO, Sade-MMTS, LSHADE44, c-FOA and c-FOA/g. The convergence diagrams show the evolution of the best objective function value for each independent run. On the longitudinal axis is the number of iterations, a function of the population size (and local search for Sade-MMTS), and on the vertical axis the objective function value. In SHADE44 the population size is diminishing while converging and hence requires more iterations. As observed, c-FOA/g achieved the best convergence rate.



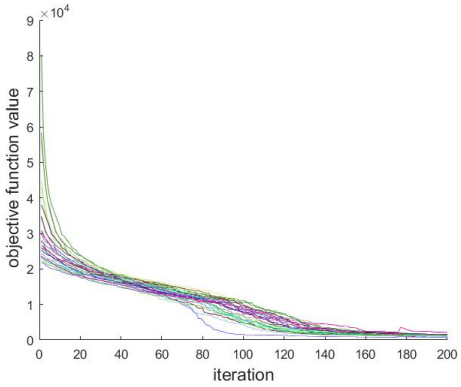
GA



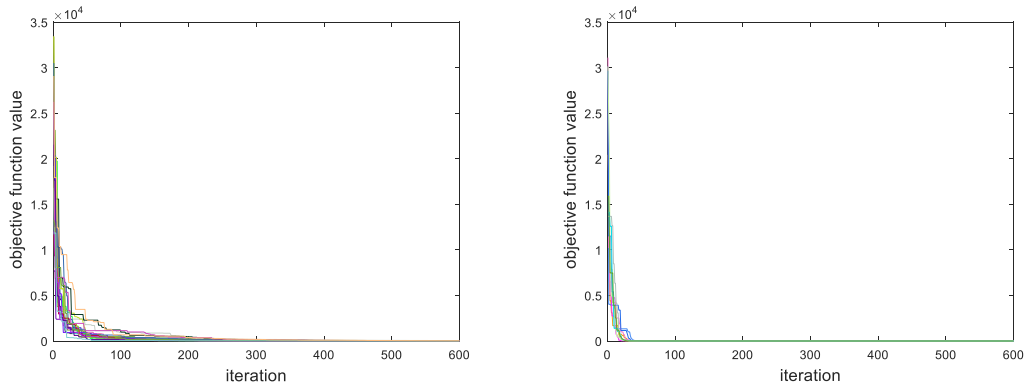
PSO



SaDE-MMTS



LSHADE44



c-FOA c-FOA/g

Figure 3 Convergence diagrams for the Genetic Algorithm (GA), Particle Swarm Optimisation (PSO), Self-adaptive Differential Evolution with Multi-trajectory Search (SaDE-MMTS), Linear population size reduction Success-History Based Adaptive Differential Evolution Algorithm (LSHADE44), contrast-based Fruit Fly optimisation (c-FOA) and contrast-based Fruit Fly optimisation with group policy (c-FOA/g). The graphs show the evolution of the best objective function value for each independent run. Termination criterion was set to 6000 function evaluations. The number of iterations depends on the population size and whether local search algorithms were applied.

4.2 Soft sensor tuning by combining global and local search methods

Population-based algorithms are expected to be more effective in global search, while gradient-based ones are expected to be better in local search. To this end, it was examined whether the combination of c-FOA/g (global search) and BR-BP (local search) produces better results. c-FOA/g was first applied and then BR-BP was initiated using as starting vector the final solution of c-FOA/g. BR-BP was run for 50 epochs.

A vital issue in the combination of global and local search methods is the decision when to switch between them. In this paper, an automatic trigger using a Bayesian binomial inference technique was applied (Gunawan and Papalambros, 2006). The trigger is based on the assumption that c-FOA/g optimisation can be modelled as a Bernoulli process, such as a coin toss, whose probabilities of "success" and "failure" are p and $(1-p)$, respectively. "Success" is when a population member produces an objective function value lower than a threshold (function of the best current value), while "failure" is the opposite case. Given N_t independent trials (population members), the probability of having r successes out of these trials follows a Binomial

distribution: $r \sim Bin(N_t, p)$. Given r successes out of N_t trials, the probability distribution of p can be calculated using Bayes' theorem:

$$f(p\backslash r) = \frac{f(p) \cdot f(r\backslash p)}{\int_0^1 f(p) \cdot f(r\backslash p) \cdot dp} \tag{3}$$

The posterior distribution is the distribution of interest. It is the estimate of p based on the outcome of the trials. The prior distribution is our knowledge about p before the information from the trials. Using a uniform prior and a Binomial likelihood function in Equation (3), results in a Beta posterior distribution, where $\alpha = r$ and +1 and $\beta = N_t - r + 1$.

$$f(p\backslash r) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \cdot \Gamma(\beta)} \cdot p^{\alpha - 1} \cdot (1 - p)^{\beta - 1} \tag{4}$$

In other words, p is distributed according to a Beta distribution whose two parameters depend on the outcome of the trials: $p \sim Beta(r + 1, N_t - r + 1)$. One very important feature of Bayes' theorem is that it facilitates an updating scheme to account for additional information. Suppose that after the N_{pop} initial trials, we conduct N_2 additional trials and observe r_2 more successes. In Bayes' theorem, the posterior distribution from the N_{pop} trials can be used as the prior distribution for the N_2 trials, thus creating a chain of analysis based on additional information.

In case the best objective function value $e_{fuel}[k]$ in iteration k has changed significantly from $e_{fuel}[k-1]$ in iteration $k-1$, the prior distribution in iteration k is assumed uniform. In the opposite case, the prior distribution resulting from iteration $k-1$ is employed. The switching criterion is triggered when the lower limit of the confidence interval of success is greater than 50%. In that case, it is assumed that c-FOA/g has entered the exploitation phase, and therefore we switch to the gradient-based local search algorithm.

To assess the performance of the proposed hybrid algorithm c-FOA/g+ BR-BP-50 it is compared to the performance of the BR-BP-x for different number x of training epochs. The training results for 30 independent runs are presented in Table 4. The

returned value of $p = 8.4 \cdot 10^{-5}$ indicates that the Kruskal-Wallis rejects the null hypothesis that the data samples produced using c-FOA/g + BR-BP-50 and BR-BP-100 come from the same distribution at 1% significance level. The same holds for the datasets c-FOA/g + BR-BP-50 and BR-BP-250 where $p = 4.1 \cdot 10^{-3}$.

Table 4. Training results obtained following 30 independent runs. e_{fuel} is the minimum mean squared error value e_{fuel} in kilometres per litre $[kpl]$. The training algorithms studied are: BR-BP-1000 (Bayesian Regularisation Back Propagation terminated after 1000 epochs), BR-BP-750, BR-BP-500, BR-BP-250, BR-BP-100, BR-BP-50 and hybrid c-FOA/g+BR-BP-50.

e_{fuel} $[kpl]$					
	μ	σ	$\widetilde{D^*}$	min	$\bar{t}\_comp$ / s
BR-BP-1000	6.7·10−1	2.3·10−1	5.8·10−1	4.1·10−1	3932 s
BR-BP-750	$5.5 \cdot 10^{-1}$	$1.5 \cdot 10^{-1}$	$5.3 \cdot 10^{-1}$	$3.3 \cdot 10^{-1}$	3425 s
BR-BP-500	$5.6 \cdot 10^{-1}$	$1.4 \cdot 10^{-1}$	$5.2 \cdot 10^{-1}$	$4.2 \cdot 10^{-1}$	1638 s
BR-BP-250	$5.5 \cdot 10^{-1}$	$2.4 \cdot 10^{-1}$	$5.0 \cdot 10^{-1}$	$3.5 \cdot 10^{-1}$	753 s
BR-BP-100	$4.8 \cdot 10^{-1}$	$3.7 \cdot 10^{-2}$	$4.8 \cdot 10^{-1}$	$4.0 \cdot 10^{-1}$	332 s
BR-BP-50	$5.9 \cdot 10^{-1}$	$3.0 \cdot 10^{-2}$	$5.9 \cdot 10^{-1}$	$5.4 \cdot 10^{-1}$	116 s
c-FOA/g + BR-BP-50	$4.3 \cdot 10^{-1}$	$4.9 \cdot 10^{-2}$	$4.1 \cdot 10^{-1}$	$3.5 \cdot 10^{-1}$	1060 s

μ: mean value, σ: standard deviation, $\widetilde{D^*}$: median value, min: minimum value, $\bar{t}\_comp$: average computational cost for one solution

The most robust results are obtained with the combined c-FOA/g + BR-BP-50. The minimum value obtained was $e_{fuel} = 0.35\ kpl$. This represents approximately 6% of the actual fuel consumption. The correlation between estimated and measured values is 0.96. Figure 4 compares graphically in the time domain the measured and estimated fuel consumption values for the training dataset. The fuel consumption is measured in kilometres per liter (kpl). The blue line represents the actual values and the red line the estimated ones. As observed, the soft sensor follows very well the measured values. It only misses some very high peaks, which have negligible contribution to the total fuel consumption. The soft sensor performance is robust (can generalise in other datasets) as it performs well also in other datasets, not used in the training process. Figure 5 shows the performance in four other datasets. It is noticed that during the

field trials the number of passengers, tyre pressure and ambient temperature were not constant.

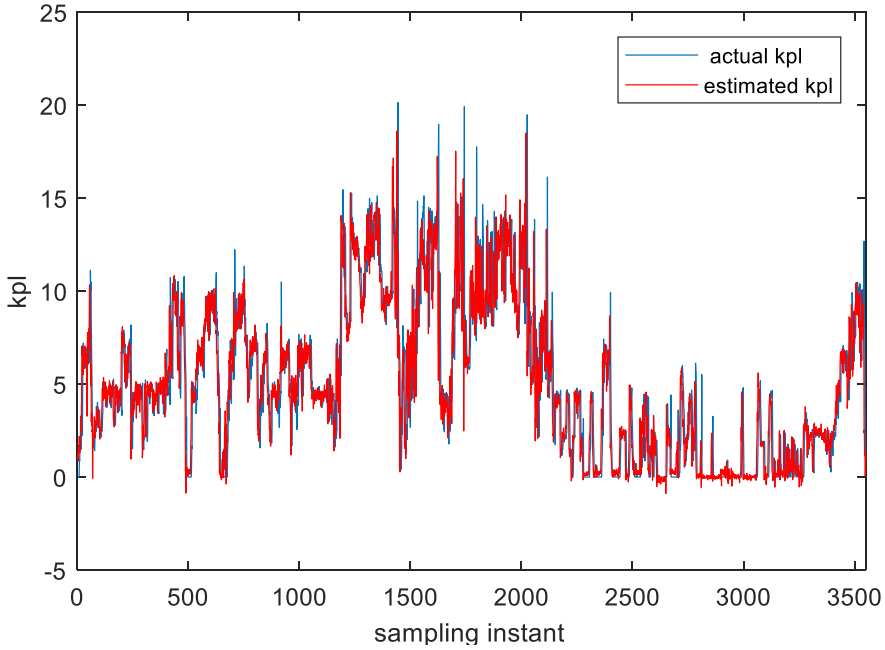
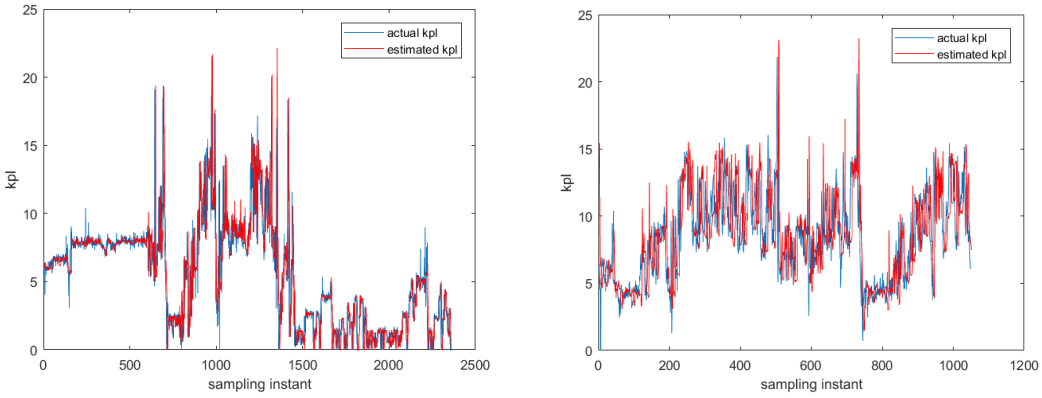
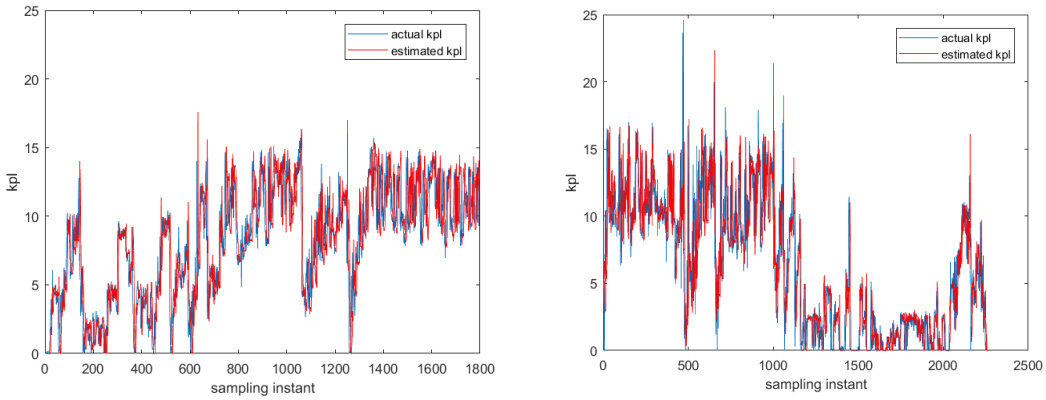


Figure 4: Comparison between the actual vehicle fuel consumption (blue line) and the estimated one (red line). Fuel consumption is measured in kilometres per liter [kpl]



a b



c d

Figure 5: Comparison between the actual vehicle fuel consumption (blue line) and the estimated one (red line) for several sample routes. The sample routes were not used in the training process of the soft sensor. Fuel consumption is measured in kilometres per liter $[kpl]$

4.3 Discussion

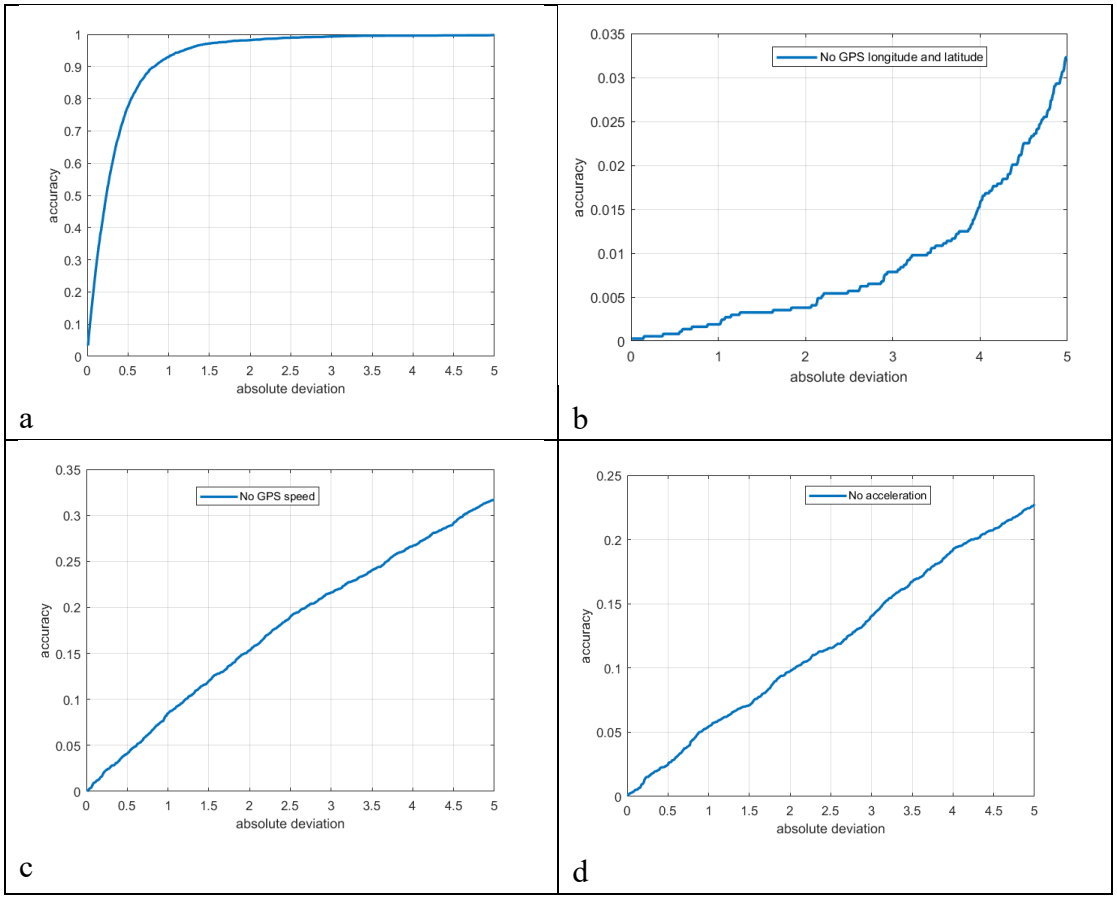
In this paper, we use the smartphone GPS position and altitude, speed, triaxial acceleration, and the number of visible satellites as inputs to the fuel consumption soft sensor. In the previous literature fewer variables, for example, only the GPS position or GPS speed, have been used. To answer the question whether all inputs are required for the soft sensor, a qualitative parametric analysis was conducted. In particular, the shape of the Regression Error Characteristic (REC) curve under incomplete information was studied (Bi, 2003). The change in the curve reveals the relative importance of the missing information. Additionally, a quantitative comparison between the proposed and other methods found in the literature was performed.

The Regression Error Characteristic (REC) curve plots the error tolerance on the x-axis versus the percentage of points predicted within the tolerance on the y-axis. The error is the absolute difference $|\mathbf{o_{NN}} - \mathbf{o_{meas}}|$between the estimated fuel consumption values $\mathbf{o_{NN}}$ and those measured $\mathbf{o_{meas}}$. The error is measured in kpl. The area over the curve (AOC) is an estimate of the expected error. The smaller the AOC, the lower the error. In Figure 6, all the REC curves used in the parametric analysis are plotted.

Figure 6a illustrates the soft sensor performance when the complete set of inputs is available. It has the smallest AOC compared to the rest. Furthermore, its shape fits the shape of a robust REC. For very tight tolerances the number of misses is high,

however the performance improves significantly (>80%) as soon as the threshold increases to a reasonable value (0.5 kpl). Figure 6b is obtained when the GPS longitude and latitude information is disregarded as an input. As observed, the performance is inferior compared to a random estimator (below the diagonal). Instead, when GPS speed or acceleration are ignored the sensor behaves almost like a random estimator, see Figures 6c and 6d respectively (REC curve is very close to the diagonal). The sensor acts again worse than a random estimator when the GPS altitude information is missing (Figure 6e). Finally, in Figure 6f shows the performance when the number of satellites is missing. The sensor behaviour is better than that of a random estimator.

From the qualitative analysis it is possible to conclude that the most critical parameters are the GPS longitude, latitude and altitude, while the least significant variable is the number of visible satellites.



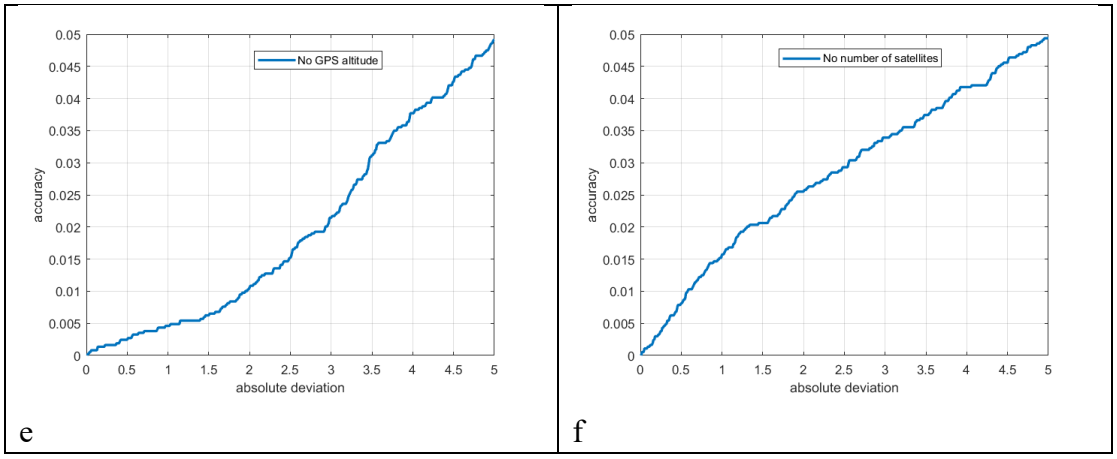


Figure 6: Regression Error Characteristics (REC) curves for the soft sensor regression performance: a) REC curve when the complete input set is available b) REC curve when GPS latitude and longitude are disregarded c) REC curve when GPS speed is disregarded d) REC curve when acceleration is disregarded e) REC curve when GPS altitude is disregarded f) REC curve when the number of satellites is disregarded.

To validate the above qualitative results the soft sensor regression performance was studied when fewer input variables are used. The following three input sets were considered available from the original training dataset: i) GPS position, ii) GPS position + GPS speed, iii) GPS position + speed + altitude. In each case, 30 independent runs were conducted to obtain statistically valid results. The combined c-FOA/g + BR-BP-50 hybrid algorithm was employed. The soft sensor performance e_{fuel} was compared to one when the complete input set is available (GPS position + speed + altitude + accelerations + number of satellites), as well as with other methods found in the literature (Vilaca *et al.*, 2015, Capraz *et al*., 2016). Table 5 lists the results. For the boosted tree method (Vilaca *et al.*, 2015) we considered two cases. In the first case, as in Vilaca *et al.*, 2015, the GPS position, speed and altitude were considered as inputs.
In the second case, only the GPS position and speed were taken into account.

For the proposed RNN soft sensor the best performance is achieved when the complete input set is considered. The performance is improved by approximately 35%, and 25% compared to when only the GPS position and speed are considered respectively. Compared to the boosted tree and support vector regression methods the proposed method performs better. This is most probably because the history of the input values is taken into account and therefore the model is less sensitive to the instantaneous values of the inputs.

Table 5. Fuel consumption e_{fuel} tuning results with c-FOA/g+BR-BP for different sets of inputs: GPS position, GPS position + GPS speed, and GPS position + GPS speed + GPS altitude. Fuel consumption is measured in kpl

	$e_{fuel}\ [kpl]$			
	μ	σ	$\widetilde{D^*}$	min
GPS position	$6.6 \cdot 10^{-1}$	$3.3 \cdot 10^{-2}$	$6.6 \cdot 10^{-1}$	$6.0 \cdot 10^{-1}$
GPS position + speed	$5.7 \cdot 10^{-1}$	$3.1 \cdot 10^{-2}$	$5.7 \cdot 10^{-1}$	$4.9 \cdot 10^{-1}$
GPS position + speed + altitude	$5.4 \cdot 10^{-1}$	$3.3 \cdot 10^{-2}$	$5.4 \cdot 10^{-1}$	$4.6 \cdot 10^{-1}$
GPS position + speed + altitude + accelerations + number of satellites	$4.3 \cdot 10^{-1}$	$4.9 \cdot 10^{-2}$	$4.1 \cdot 10^{-1}$	$3.5 \cdot 10^{-1}$
Boosted tree: GPS position + speed + altitude (Vilaca *et al*, 2015)	$8.1 \cdot 10^{-1}$	0	$8.1 \cdot 10^{-1}$	$8.1 \cdot 10^{-1}$
Boosted tree: GPS position + speed (Vilaca *et al*, 2015)	$7.4 \cdot 10^{-1}$	0	$7.4 \cdot 10^{-1}$	$7.4 \cdot 10^{-1}$
Support Vector Machines: GPS position + speed + altitude (Capraz *et al*., 2016)	1.7	$3.5 \cdot 10^{-1}$	1.7	1.4
Support Vector Machines: GPS position + speed (Capraz *et al*., 2016)	2.6	$4.0 \cdot 10^{-2}$	2.6	2.5

5. Conclusions

This paper presents a novel vehicle fuel consumption soft sensor based on Recurrent Neural Networks. Most of the contributions in the literature focus on average consumption models, while this paper presents a model based on instantaneous data. The particular challenge in the proposed method is the handling of inaccuracy in the inputs, comprising smartphone measurements of GPS position (latitude, longitude, altitude), speed, acceleration (longitudinal, lateral, vertical), and the number of visible satellites.

To this end, different types of Deep Neural Networks, architectures and tuning methods were systematically examined. The latter included Levenberg–Marquardt Back Propagation (LM-BP), Bayesian Regularisation Back Propagation (BR-BP), Success-History Based Adaptive Differential Evolution Algorithm (SHADE), Linear population size reduction Success-History Based Adaptive Differential Evolution Algorithm (LSHADE, LSHADE44), Enhanced Unidimensional Search (EUS), adaptive Enhanced Unidimensional Search algorithm (aEUS), Differential Search Algorithm (B-DSA), Self-adaptive Differential Evolution with Multi-trajectory Search (SaDE-MMTS), cloud-based Fruit Fly Optimisation, chaotic Fruit Fly Optimisation and contrast-based Fruit Fly Optimisation.

1. A comparison between Long-Short-Term-Memory Neural Networks and Recurrent Neural Networks (RNN) showed that the second type is more appropriate for the instantaneous fuel consumption estimation of vehicles. The optimal RNN architecture was derived following a parametric analysis, where the number of hidden layers and neurons were iteratively optimised.

2. The best training results were achieved when the contrast-based Fruit Fly Optimisation with group policy (c-FOA/g) was first applied followed by Bayesian Regularisation Back Propagation (BR-BP). An empirical criterion based on a Bayesian binomial inference technique has been proposed for automatically switching between global and local search.

3. The soft sensor estimation error is less than 6% of the actual fuel consumption value and the correlation achieved is 0.96. A comparison to other methods found in the literature confirmed the superior performance of the proposed sensors. A principal component analysis was conducted and the relevant significance of each input was determined. The most important inputs are the GPS longitude, latitude, and altitude. All inputs were found to contribute to the overall accuracy of the soft sensor.

The unprecedented market penetration of smartphones is an enabler for the massive implementation of soft sensors. The potential impact of the presented soft sensor in Intelligent Transportation Applications is significant as it relies only on smartphone

measurements and can estimate accurately the vehicle fuel consumption. In the future, it is foreseen to investigate the performance of the soft sensor by combining analytical vehicle models and data-based approaches to make the results more interpretable.

Acknowledgments

We would like to thank Dr Stavros-Richard Christopoulos and Manuel Acosta for contributing to the collection of the smartphone and vehicle fuel consumption data. MEF and JM are grateful for funding from the Lloyd's Register Foundation, a charitable foundation helping to protect life and property by supporting engineering-related education, public engagement and the application of research.

References

Acosta, M., & Kanarachos, S. (2017). Tire lateral force estimation and grip potential identification using Neural Networks, Extended Kalman Filter, and Recursive Least Squares. Neural Computing And Applications. doi: 10.1007/s00521-017-2932-9

Acosta, M., Kanarachos, S., & Blundell, M. (2017). Virtual tyre force sensors: an overview of tyre model-based and tyre model-less state estimation techniques. Proceedings Of The Institution Of Mechanical Engineers, Part D: Journal Of Automobile Engineering, 095440701772819. doi: 10.1177/0954407017728198

Arsie, I., Pianese, C., & Sorrentino, M. (2010). Development of recurrent neural networks for virtual sensing of nox emissions in internal combustion engines. SAE International Journal of Fuels and Lubricants, 2(2), 354-361. http://dx.doi.org/10.4271/2009-24-0110

Belagiannis, V., Rupprecht, C., Carneiro, G., & Navab, N. (2015). Robust optimization for deep regression. Paper presented at the Proceedings of the IEEE International Conference on Computer Vision, , 2015 International Conference on Computer Vision, ICCV 2015 2830-2838. http://dx.doi.org/10.1109/ICCV.2015.324

Bi, J., Bennett, K., Regression error characteristic curves, Proceedings of 20th Int. Conf. on Machine Learning (ICML), Washington DC, USA, 2003.

Bianchi, F. M. (2017). Recurrent neural networks for short-term load forecasting: An overview and comparative analysis. Cham, Switzerland: Springer.

Boer, G. D., & Krootjes, P. (2012). The quality of floating car data benchmarked: An alternative to roadside equipment? Paper presented at the 19th Intelligent Transport Systems World Congress, ITS 2012, EU-00455.

Çapraz, A., Özel, P., Şevkli, M., & Beyca, Ö. (2016). Fuel Consumption Models Applied to Automobiles Using Real-time Data: A Comparison of Statistical Models. Procedia Computer Science, 83, 774-781. doi: 10.1016/j.procs.2016.04.166

Capriglione, D., Carratù, M., Liguori, C., Paciello, V., & Sommella, P. (2017). A soft stroke sensor for motorcycle rear suspension. Measurement: Journal of the International Measurement Confederation, 106, 46-52. http://dx.doi.org/10.1016/j.measurement.2017.04.011

Cawley, G. C., & Talbot, N. L. C. (2007). Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters. Journal of Machine Learning Research, 8, 841-861.

Chen, H., Guo, B., Yu, Z., Chin, A., Tian, J., & Chen, C. (2017). Which is the greenest way home? A lightweight eco-route recommendation framework based on personal driving habits. Paper presented at the Proceedings - 12th International Conference on Mobile Ad-Hoc and Sensor Networks, MSN 2016, 187-194. doi:10.1109/MSN.2016.038

Civicioglu, P. (2012). Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. Computers & Geosciences, 46, 229-247. doi: 10.1016/j.cageo.2011.12.011

Dey, S., Gupta, S., Sibanda, P., & Chakraborty, A. (2017). Spatio-temporal variation and futuristic emission scenario of ambient nitrogen dioxide over an urban area of eastern india using GIS and coupled AERMOD-WRF model. PLoS ONE, 12(1) doi:10.1371/journal.pone.0170928

Du, Y., Wu, J., Yang, S., & Zhou, L. (2017). Predicting vehicle fuel consumption patterns using floating vehicle data. Journal Of Environmental Sciences, 59, 24-29. doi: 10.1016/j.jes.2017.03.008

Du, T., Ke, X., Liao, J., & Shen, Y. (2018). DSLC-FOA : Improved fruit fly optimization algorithm for application to structural engineering design optimization problems. Applied Mathematical Modelling, 55, 314-339. doi: 10.1016/j.apm.2017.08.013

Duchanoy, C. A., Moreno-Armendáriz, M. A., Urbina, L., Cruz-Villar, C. A., Calvo, H., & de J. Rubio, J. (2017). A novel recurrent neural network soft sensor via a differential evolution training algorithm for the tire contact patch. Neurocomputing, 235, 71-82. https://doi.org/110.1016/j.neucom.2016.12.060

Ferreira, A. A., Ludermir, T. B., & De Aquino, R. R. B. (2012). Comparing recurrent networks for time-series forecasting. Paper presented at the Proceedings of the International Joint Conference on Neural Networks, http://dx.doi.org/10.1109/IJCNN.2012.6252459

Ferreira, M. D., Corrêa, D. C., Nonato, L. G., & de Mello, R. F. (2018). Designing architectures of convolutional neural networks to solve practical problems. Expert Systems with Applications, 94, 205-217. doi:10.1016/j.eswa.2017.10.052

Fontaras, G., Zacharof, N., & Ciuffo, B. (2017). Fuel consumption and CO_2 emissions from passenger cars in Europe – Laboratory versus real-world emissions. Progress In Energy And Combustion Science, 60, 97-131. doi: 10.1016/j.pecs.2016.12.004

Forehead, H., & Huynh, N. (2018). Review of modelling air pollution from traffic at street-level - the state of the science. Environmental Pollution, 241, 775-786. doi:10.1016/j.envpol.2018.06.019

Fruit Fly Optimization Algorithm, https://www.mathworks.com/matlabcentral/answers/uploaded_files/20100/Fruit%20Fly%20Optimization%20Algorithm_Second%20Edition.pdf, accessed on 25.03.2018.

Gao, K., Zhang, Y., Sadollah, A., Lentzakis, A., & Su, R. (2017). Jaya, harmony search and water cycle algorithms for solving large-scale real-life urban traffic light scheduling problem. Swarm and Evolutionary Computation, 37, 58-72. https://doi.org/10.1016/j.swevo.2017.05.002

Gardeux, V., H. Omran, M., Chelouah, R., Siarry, P., & Glover, F. (2017). Adaptive pattern search for large-scale optimization. Applied Intelligence, 47(2), 319-330. doi: 10.1007/s10489-017-0901-8

Gunawan, S. and Papalambros, P. (2006). A Bayesian Approach to Reliability-Based Optimization With Incomplete Information. Journal of Mechanical Design, 128(4), p.909.

Ibrahim, A. and El-Amary, N. (2017). Particle Swarm Optimization trained recurrent neural network for voltage instability prediction. Journal of Electrical Systems and Information Technology. https://doi.org/10.1016/j.jesit.2017.05.001

Ismkhan, H. (2017). Effective heuristics for ant colony optimization to handle large-scale problems. Swarm and Evolutionary Computation, 32, 140-149. https://doi.org/10.1016/j.swevo.2016.06.006

Kan, Z., Tang, L., Kwan, M. -., & Zhang, X. (2018). Estimating vehicle fuel consumption and emissions using GPS big data. International Journal of Environmental Research and Public Health, 15(4) doi:10.3390/ijerph15040566

Kanarachos, S., Dizqah, A. M., Chrysakis, G., & Fitzpatrick, M. E. (2018). Optimal design of a quadratic parameter varying vehicle suspension system using contrast-based fruit fly optimisation. Applied Soft Computing Journal, 62, 463-477. https://doi.org/10.1016/j.asoc.2017.11.005

Kanarachos, S., Christopoulos, S. -. G., & Chroneos, A. (2018). Smartphones as an integrated platform for monitoring driver behaviour: The role of sensor fusion and connectivity. Transportation Research Part C: Emerging Technologies, doi:10.1016/j.trc.2018.03.023

Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

Kumar Pathak, S., Sood, V., Singh, Y., & Channiwala, S. (2016). Real world vehicle emissions: Their correlation with driving parameters. Transportation Research Part D: Transport And Environment, 44, 157-176. doi: 10.1016/j.trd.2016.02.001

Masikos, M., Demestichas, K., Adamopoulou, E., & Theologou, M. (2014). Mesoscopic forecasting of vehicular consumption using neural networks. Soft Computing, 19(1), 145-156. doi:10.1007/s00500-014-1238-4

Madrazo, J., & Clappier, A. (2018). Low-cost methodology to estimate vehicle emission factors. Atmospheric Pollution Research, 9(2), 322-332. doi:10.1016/j.apr.2017.10.006

Meseguer, J. E., Calafate, C. T., Cano, J. C., & Manzoni, P. (2013). DrivingStyles: A smartphone application to assess driver behavior. Paper presented at the Proceedings - International Symposium on Computers and Communications, 535-540. doi:10.1109/ISCC.2013.6755001

Nakib, A., Ouchraa, S., Shvai, N., Souquet, L., & Talbi, E. -G. (2017). Deterministic metaheuristic based on fractal decomposition for large-scale optimization. Applied Soft Computing Journal, 61, 468-485. https://doi.org/10.1016/j.asoc.2017.07.042

Nweke, H. F., Teh, Y. W., Al-garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor

networks: State of the art and research challenges. Expert Systems with Applications, 105, 233-261. doi:10.1016/j.eswa.2018.03.056
Orfila, O., Saint Pierre, G., & Messias, M. (2015). An android based ecodriving assistance system to improve safety and efficiency of internal combustion engine passenger cars. Transportation Research Part C: Emerging Technologies, 58, 772-782. doi: 10.1016/j.trc.2015.04.026
Pan, W. -T. (2012). A new fruit fly optimization algorithm: Taking the financial distress model as an example. Knowledge-Based Systems, 26, 69-74. https://doi.org/10.1016/j.knosys.2011.07.001
Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. Paper presented at the 30th International Conference on Machine Learning, ICML 2013, (PART 3) 2347-2355.
Patel G.C, M., Shettigar, A. K., Krishna, P., & Parappagoudar, M. B. (2017). Back propagation genetic and recurrent neural network applications in modelling and analysis of squeeze casting process. Applied Soft Computing Journal, 59, 418-437. https://doi.org/110.1016/j.asoc.2017.06.018
Peng, X., & Wu, Y. (2017). Large-scale cooperative co-evolution using niching-based multi-modal optimization and adaptive fast clustering. Swarm and Evolutionary Computation, 35, 65-77. https://doi.org/10.1016/j.swevo.2017.03.001
Piotrowski, A. P. (2014). Differential evolution algorithms applied to neural network training suffer from stagnation. Applied Soft Computing Journal, 21, 382-406. https://doi.org/10.1016/j.asoc.2014.03.039
Piotrowski, A. P., Napiorkowski, M. J., Kalinowska, M., Napiorkowski, J. J., & Osuch, M. (2016). Are evolutionary algorithms effective in calibrating different artificial neural network types for streamwater temperature prediction? Water Resources Management, 30(3), 1217-1237. https://doi.org/10.1007/s11269-015-1222-5
Piotrowski, A., & Napiorkowski, J. (2018). Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure?. Swarm And Evolutionary Computation. doi: 10.1016/j.swevo.2018.03.007
Polakova, R. (2017). L-SHADE with competing strategies applied to constrained optimization. Paper presented at the 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings, 1683-1689. doi:10.1109/CEC.2017.7969504
Pucher, G. (2016). Quantification of traffic related CO2 emissions through extended floating car data (XFCD). Gis.Science - Die Zeitschrift Fur Geoinformatik, 1, 23-29.
Rajamani, R. (2014). Vehicle dynamics and control. [Place of publication not identified]: Springer.
Sun, D., Zhang, K., & Shen, S. (2018a). Analyzing spatiotemporal traffic line source emissions based on massive didi online car-hailing service data. Transportation Research Part D: Transport and Environment, 62, 699-714. doi:10.1016/j.trd.2018.04.024
Sun, Y., Wang, X., Chen, Y., & Liu, Z. (2018b). A modified whale optimization algorithm for large-scale global optimization problems. Expert Systems with Applications, 114, 563-577. doi:10.1016/j.eswa.2018.08.027
Tanabe, R., & Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. Paper presented at the 2013 IEEE Congress on

Evolutionary Computation, CEC 2013, 71-78. doi:10.1109/CEC.2013.6557555

Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of SHADE using linear population size reduction. Paper presented at the Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014, 1658-1665. doi:10.1109/CEC.2014.6900380

Turkensteen, M. (2017). The accuracy of carbon emission and fuel consumption computations in green vehicle routing. European Journal Of Operational Research, 262(2), 647-659. doi: 10.1016/j.ejor.2017.04.005

Van Breugel, F., & Dickinson, M. H. (2014). Plume-tracking behavior of flying drosophila emerges from a set of distinct sensory-motor reflexes. Current Biology, 24(3), 274-286. https://doi.org/10.1016/j.cub.2013.12.023

Vilaça A., Aguiar A., Soares C. (2015) Estimating Fuel Consumption from GPS Data. In: Paredes R., Cardoso J., Pardo X. (eds) Pattern Recognition and Image Analysis. IbPRIA 2015. Lecture Notes in Computer Science, vol 9117. Springer, Cham

Wu, J. -., & Liu, J. -. (2011). Development of a predictive system for car fuel consumption using an artificial neural network. Expert Systems with Applications, 38(5), 4967-4971. doi:10.1016/j.eswa.2010.09.155

Wu, J., & Liu, J. (2012). A forecasting system for car fuel consumption using a radial basis function neural network. Expert Systems With Applications, 39(2), 1883-1888. doi: 10.1016/j.eswa.2011.07.139

Wu, L., Zuo, C. and Zhang, H. (2015). A cloud model based fruit fly optimization algorithm. Knowledge-Based Systems, 89, pp.603-617.

Yamashita, R. -., Yao, H. -., Hung, S. -., & Hackman, A. (2018). Accessing and constructing driving data to develop fuel consumption forecast model. Paper presented at the IOP Conference Series: Earth and Environmental Science, , 113(1) doi:10.1088/1755-1315/113/1/012217

Zeng, Weiliang & Miwa, Tomio & Wakita, Y & Morikawa, Takayuki. (2015). Exploring Trip Fuel Consumption by Machine Learning from GPS and CAN Bus Data. Journal of the Eastern Asia Society for Transportation Studies. 11. 906-921. 10.11175/easts.11.906.

Zhao, S., Suganthan, P., & Das, S. (2010). Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. Soft Computing, 15(11), 2175-2185. doi: 10.1007/s00500-010-0645-4

Zhou, M., Jin, H., & Wang, W. (2016). A review of vehicle fuel consumption models to evaluate eco-driving and eco-routing. Transportation Research Part D: Transport And Environment, 49, 203-218. doi: 10.1016/j.trd.2016.09.008

Appendix A: The contrast-based Fruit Fly Optimisation Algorithm with group policy (c- FOA/g)

The algorithm starts by arbitrarily defining the position (X_0, Y_0) of the first fruit fly in a coordinate system. Additional N–1 fruit flies are located, randomly, in the vicinity of (X_0, Y_0) according to Eq. (1).

$$\begin{aligned} X_{ij}[k] &= X_{0j}[k] \cdot \left(1 + M \cdot (2 \cdot rand_{N_{res}} - 1\right), j=1,2,\ldots,m \text{ and } i=1,\ldots,N \\ Y_{ij}[k] &= Y_{0j}[k] \cdot \left(1 + M \cdot (2 \cdot rand_{N_{res}} - 1\right), j=1,2,\ldots,m \text{ and } i=1,\ldots,N \end{aligned} \quad \text{(A1.1)}$$

where $k = 1,2, \ldots, K_{max}$ is the iteration number, m is the number of optimisation variables, N is the size of the swarm and $rand_{N_{res}}$ is a random number, sampled from a uniform discrete distribution defined in the interval $[1, N_{res}]$. M is a scaling parameter that determines how coarse or fine the search strategy is.

Each fruit fly is assigned values DI_{ij} based on how close each fruit fly parameter $(X_{ij}[k], Y_{ij}[k])$ is to the origin of the coordinate system:

$$D_{ij}[k] = \sqrt{X_{ij}^2[k] + Y_{ij}^2[k]} \quad \text{(A1.2)}$$

$$DI_{ij}[k] = \frac{1}{D_{ij}[k]} \quad \text{(A1.3)}$$

For each fruit fly at $\mathbf{d_i}[k]$ an objective function value $Dm_i[k]$ is assigned, $Dm_i[k] = f(\mathbf{d}_i[k])$.

Then, we rank the fruit flies based on their objective function values, and the fruit fly $\mathbf{d}^*[k]$ that achieves the lowest objective function value $Dm_i^*[k]$ at position $(X_i^*[k], Y_i^*[k])$ is identified. In case the objective function value $Dm_i^*[k]$ is lower than the previous centre of attraction $D_0[k]$, then $Dm_i^*[k]$ becomes the new centre of attraction $\mathbf{d}_0[k]$ $(X_0[k], Y_0[k])$.

$$\begin{aligned} &\boldsymbol{if}\ \boldsymbol{Dm_i^* < D_{m,k0}} \\ &\qquad then\ X_0[k] = X_i^*[k]\ and\ Y_0[k] = Y_i^*[k] \end{aligned} \quad \text{(A1.4)}$$

The algorithm can change their search strategy only every κ iterations. This resembles the delay that real fruit flies present in changing search strategy. If the best objective function value $\mathbf{d}^*[k]$ improves over the last κ iterations the swarm enters the "surging" phase, during which the fruit flies surge towards the attraction point $\mathbf{d}_0[k]$:

$$\begin{aligned} &if\,(Dm_i^*[k] < Dm_i^*[k-\kappa]) \\ &\boldsymbol{M[k+1] = c \cdot M[k]} \end{aligned} \tag{A1.5}$$

In case the best objective function value does not change over the last κ iterations the swarm enters the "visual contrast" phase, during which the fruit flies are attracted by the point $\mathbf{s}_i^{\cdot\cdot}[k]$ which achieves the largest objective function value $\max(Sm_i[k]) = Sm_i^{\cdot\cdot}[k]$:

$$\begin{aligned} &if\,(Sm_i[k] = Sm_0[k-\kappa] \\ &X_0[k] = X_i^{\cdot\cdot}[k]\ and\ Y_0[k] = Y_i^{\cdot\cdot}[k] \end{aligned} \tag{A1.6}$$

where k is the current iteration.

When a fruit fly does not improve its performance, then it returns to its previous position:

$$\begin{aligned} &\boldsymbol{if\ Dm_i[k] > Dm_i[k-1]} \\ &\qquad then\ X_i[k] = X_i[k-1]\ and\ Y_i[k] = Y_i[k-1] \end{aligned} \tag{A1.7}$$

Group policy

To enable the coordinated movement, the maximum X_{max}, Y_{max} and minimum X_{min}, Y_{min} coordinate values are used:

$$\begin{aligned} X_{min} &= \min\big(X_{ij}[k-1]\big) \\ X_{max} &= \max\big(X_{ij}[k-1]\big) \\ Y_{min} &= \min\big(Y_{ij}[k-1]\big) \\ Y_{max} &= \max\big(Y_{ij}[k-1]\big) \end{aligned} \tag{A1.8}$$

to form the increments $X_{inc,k}, Y_{inc,k}$:

$$X_{inc,k} = \frac{X_{max} - X_{min}}{N_g}$$
$$Y_{inc,k} = \frac{Y_{max} - Y_{min}}{N_g} \quad \text{(A1.9)}$$

And the groups X_g, Y_g are defined:

$$X_g = \left[X_{min} + (g-1) \cdot X_{inc,k}, X_{min} + g \cdot X_{inc,k}\right]$$
$$Y_g = \left[Y_{min} + (g-1) \cdot Y_{inc,k}, Y_{min} + g \cdot Y_{inc,k}\right] \quad \text{(A1.10)}$$

where g=1,…, N_{g}. Parameter N_{g} is problem-dependent.

Each fruit fly is mapped to two groups X_g, Y_g, based on its coordinates $X_{ij}[k-1]$, $Y_{ij}[k-1]$. At each iteration, the fruit flies are positioned according to Eq. (A1.12):

$$X_{i,j}[k] = X_{bas}[k] + X_{0j}[k] \cdot \left(1 + M \cdot (2 \cdot rand_{N_{res,xg}} - 1\right), jg=1,2,\ldots,m$$
$$Y_{i,j}[k] = Y_{bas}[k] + Y_{0j}[k] \cdot \left(1 + M \cdot (2 \cdot rand_{N_{res,yg}} - 1\right), jg=1,2,\ldots,m \quad \text{(A1.11)}$$

where $X_{bas}[k]$ and $Y_{bas}[k]$ are the coordinates of one of the four best performing fruit flies, $\mathbf{d_{bas}}[k] \in [\mathbf{d_0}[\mathrm{k}], \mathbf{d_1}[\mathrm{k}], \mathbf{d_2}[\mathrm{k}], \mathbf{d_3}[\mathrm{k}]]$, where $\mathrm{f}(\mathbf{d_0}[\mathrm{k}]) < \mathrm{f}(\mathbf{d_1}[\mathrm{k}]) < \mathrm{f}(\mathbf{d_2}[\mathrm{k}] < \mathrm{f}(\mathbf{d_3}[\mathrm{k}]) < \mathrm{f}(\mathbf{d_i}[\mathrm{k}])$, $k = 1, 3, 5, \ldots, K_{max}$ and $rand_{N_{res,xg}}$, $rand_{N_{res,yg}}$ are unique for each group X_g, Y_g.
For $k = 2, 4, 6, \ldots, K_{max}$ the fruit flies take positions according to Eq. (A1.1).

The algorithm terminates when the maximum number K_{max} of iterations is reached.