Modeling a MEMS deformable mirror using non-parametric estimation techniques

Dani Guzmán, Francisco Javier de Cos Juez, Richard Myers, Andrés Guesalaga, and Fernando Sánchez Lasheras

1Pontificia Universidad Católica de Chile, Electrical Engineering Department, Vicuña Mackenna 4860, Santiago, Chile
2Physics Department, Durham University, South Road Laboratories, Durham, DH1 3LE, UK
3Mining Exploitation and Prospecting Department, C/Independencia nº13, University of Oviedo, 33004 Oviedo, Spain
4TecniProject S.L., Marqués de Pidal 7, 3004 Oviedo, Spain
*cdguzman@ing.puc.cl

Abstract: Using non-parametric estimation techniques, we have modeled an area of 126 actuators of a micro-electro-mechanical deformable mirror with 1024 actuators. These techniques produce models applicable to open-loop adaptive optics, where the turbulent wave front is measured before it hits the deformable mirror. The mirror’s input is the wavefront correction to apply to the mirror and its output is the set of voltages to shape the mirror. Our experiments have achieved positioning errors of 3.1% rms of the peak-to-peak wavefront excursion.

©2010 Optical Society of America

OCIS codes: (010.1080) Active or adaptive optics; (010.1285) Atmospheric correction; (010.1330) Atmospheric turbulence.

References and links


1. Introduction

Multi-Object Adaptive Optics (MOAO) is an astronomical Adaptive Optics (AO) concept [1] that would enable the simultaneous recording of spectra of multiple scientific targets at high spatial resolution in a field of view of 5 to 10 arcmin. MOAO has been proposed as an AO concept for current 8 meter telescopes [2] as well as for the next generation of extremely large telescopes [3]. In a MOAO system, the deformable mirrors operate in an optical open-loop with respect to the wavefront sensors, with the latter measuring raw incoming turbulence. Having a sufficiently accurate model of the behavior of the DM surface is key to achieving the level of correction required by the AO system.

Micro-Electro-Mechanical Systems (MEMS) deformable mirrors have come to alleviate the AO need of a deformable mirror capable of producing high orders of correction, with hysteresis-free actuators [4]. As established by Evans et al [5], a MEMS DM does not behave linearly, in the sense that the final position of the DM facesheet is not the linear combination of the individual action of the actuators. We recently modeled a deformable mirror with similar behavior, a Xinetics DM with electrostrictive actuators [6]. We applied a non-parametric regression technique called Multivariate Adaptive Regression Splines (MARS) to obtain positioning errors of the order of 1%. Other groups are actively researching this problem: there are previous and relevant results in the literature, both modeling MEMS DMs. Stewart et al [7] and Morzinski et al [8] have developed models based on the physics of the actuators and the DM membrane. Stewart achieved a residual error of 228 nm peak-to-valley and 41 nm rms. This result is for one focus term on the DM, with an excursion of 1.5 µm peak-to-valley. Morzinski experimented with excursions of around 500 nm, achieving rms errors of 16 nm. Expressing the residual error as a percentage of the peak-to-valley excursion (the figure of merit we introduced in [6]), Stewart achieved a residual error of 2.7% rms, for a single focus term and Morzinski obtained 3.3% rms for a Kolmogorov pattern. Blain et al [9], who recently published a MEMS DM model closer to our own approach, in the sense the model does not rely in the physics of the problem (as Stewart and Morzinski do), but it is built from a thorough characterization of the device, using the additivity of the influence functions. They obtain results simulating turbulent phase screens, which are assessed with a slightly different figure of merit: the rms residual error expressed as a percentage of the rms value of the requested phase screen. Blain obtained 11% rms residual error, for 100 phase screens produced by the DM. To establish a valid comparison with these previous works, we use both figures of merit throughout this paper.

In general, non-parametric estimation are those techniques that can adapt their functionality to the data they are modeling, not assuming any predefined structure [10]. They are very appropriate to model non-linear systems, but they do need access to a large database of experimental data, so their parameters can be adjusted through exposure to the data, a process usually called ‘training’. We are using two different techniques for modeling our MEMS DM: MARS and Artificial Neural Networks (ANN). The former is explained in detail...
in Guzmán et al [6], so it is only briefly described in this article. ANN is explained in detail in the next section.

We have implemented non-parametric models for an area of a large MEMS DM. The models are then tested with a focus term, similar to Stewart’s paper as well as with random positions, similar to Blain’s paper. We believe these models produce comparable or better results with respect to previous works, with the benefit of having only one stage of training before usage.

2. Non-parametric estimation techniques

In this section, we provide some theoretical background for the non-parametric estimation techniques we used to produce our DM models.

2.1 Multivariate Adaptive Regression Splines (MARS)

MARS is a multivariate non-parametric regression technique introduced by Friedman [11] in 1991. The MARS method is a local regression method that uses a series of local so-called basis functions to model complex (non-linear) relationships [12]. The space of the predictors is split into several (overlapping) regions in which so-called spline functions are fit. The global MARS model then consists of the weighted sum of the local models, i.e.:

\[
\hat{y} = a_0 + \sum_{m=1}^{M} a_m B_m(x)
\]

where \(\hat{y}\) is the predicted response, \(a_0\) the coefficient of the constant basis function, \(B_m(x)\) the \(m^{th}\) basis function, which may be a single spline function or a product (interaction) of two (or more) spline functions, \(a_m\) the coefficient of the \(m^{th}\) basis function and \(M\) the number of basis functions included into the model.

In general, the MARS methodology consists of three steps: first a constructive phase, in which basis functions are introduced in several regions of the predictors and are combined in a weighted sum to define the global model [Eq. (1)]. This model often contains too many basis functions, which leads to overfitting. Therefore, the constructive phase is followed by a pruning phase, in which some basis functions of the overfitted model are deleted. This leads to a sequence of consecutively smaller MARS models, from which the optimal one is selected in a third step.

In the first step, a model is created by stepwise adding basis functions. Each basis function covers a given domain of the response variable. The basis functions in MARS consist either of one single spline function or the product of two (or more) spline functions for different predictors. The spline functions in MARS are piecewise polynomials, i.e. left-sided [Eq. (2)] and right-sided [Eq. (3)] truncated functions

\[
b^+(x-t) = \begin{cases} 
-((x-t)^q) & \text{if } x < t \\
0 & \text{otherwise}
\end{cases}
\]

\[
b^-(x-t) = \begin{cases} 
((x-t)^q) & \text{if } x > t \\
0 & \text{otherwise}
\end{cases}
\]

where \(t\) is called the knot location; \(b^+(x-t)\) and \(b^-(x-t)\) are spline functions describing the regions to the left and the right of the given \(t\). \(q\) indicates the power (>0) to which the spline is raised; the subscript “+” indicates that the function has been forced to zero for negative arguments.
2.2 Artificial Neural Networks (ANN)

The ANNs are mathematical modeling tools that are particularly useful in the field of prediction and forecasting in complex settings. Historically, these were meant to operate through simulating, at a simplified level, the activity of the human brain. The ANN accomplishes this through a large number of highly interconnected processing elements (neurons), working in unison to solve specific problems, such as forecasting and pattern recognition. Each neuron is connected to some of its neighbors with varying coefficients or weights that represent the relative influence of the different neuron inputs to other neurons.

The feed forward multi-layer perceptron back-propagation (MLP-BP) network is one of the most popular techniques in the field of ANN. The common topology of a MLP-BP neural network model is illustrated in Fig. 1. The source nodes in the input layer of the network supply respective elements of the activation pattern or input vector, which constitute the input signals applied to the neurons in the hidden layer. The output signals of the hidden layer are used as inputs to the output layer. The output signals of the neurons in the output layer of the network constitute the overall response of the network to the activation patterns applied by the input layer neurons.

![Fig. 1. Topology of feed forward multi-layer perceptron back-propagation ANN.](image)

With $n$ input neurons, $m$ hidden neurons, and one output neuron, the training process of MLP-BP network can be described as follows:

1. Calculate the outputs of all hidden layer nodes, using Eqs. (4) and (5):

$$\text{net}_j = \sum_{i=0}^{n} w_{ij} y_i \quad (i = 0, 1, \ldots, n; \ j = 1, \ldots, m) \quad (4)$$

$$z_j = f_H (\text{net}_j) \quad (j = 1, 2, \ldots, m) \quad (5)$$

where $\text{net}_j$ is the activation value of the $j^{th}$ node, $w_{ij}$ is the connection weight from input node $i$ to hidden node $j$, $y_i$ is the $i^{th}$ input with $y_0$ being the bias $b_{IH}$ (with weight $w_{0j} = 1$), $z_j$ is the corresponding output of the $j^{th}$ node in the hidden layer, and $f_H$ is called the activation function of a node, which is usually a sigmoid function, written in Eq. (6):

$$f_H (x) = \frac{1}{1 + e^{-x}} \quad (6)$$

2. Calculate the outputs of all output layer neurons using Eq. (7):
where $f_0$ is the activation function, usually a line function, $w_{j,k}$ is the connection weight from hidden node $j$ to output node $k$ (here $k = 1$), $z_j$ is the corresponding output of the $j^{th}$ node in the hidden layer with $z_0$ being the bias $b_{h0}$ (with weight $w_{0,k} = 1$). All the connection weights and bias values are assigned with random values initially, and then modified according to the results of MLP-BP training process.

(3) Minimize the global error $E$ via the training algorithm of Eq. (8):

$$E = \frac{1}{2} \sum (O - y)^2$$

2.3 ANN models implemented

When building a neural network model, a number of decisions must be made, including the neural network type, network structure, methods of pre- and post-processing of input/output data, the training algorithm and training stop criteria.

The feed-forward multi-layer perceptron (MLP) ANN is the most widely used type of ANN in industrial applications, and is also adopted in this study. Network architecture mainly denotes the number of input/output variables, the number of hidden layers and the number of neurons in each hidden layer. It determines the number of connection weights and the way information flows through the network. The major concern of the designer of an ANN structure is to determine the appropriate number of hidden layers and the number of neurons in each layer. There is no systematic way to establish a suitable architecture, and the selection of the appropriate number of neurons is basically problem specific. Hornik et al [13] proved that a single hidden layer network containing a sufficiently large number of neurons can be used to approximate any measurable functional relationship between the input data and the output variable to any desired accuracy. De Villars et al [14] showed that an ANN comprised of two hidden layers tends to be less robust and converges with less accuracy than its single hidden layer counterpart. Furthermore, some studies indicate that the benefits of using a second hidden layer are marginal to the rainfall-runoff modeling problem [15,16]. Taking recognizance of the above studies, a single hidden layer is used in this study.

There are some algorithms, including pruning and constructive algorithms, to determine an ‘optimum’ number of neurons in the hidden layer(s) during training. However, a trial and error procedure using different number of neurons is still the preferred choice of most users [16–18] and is the method used also in this research.

The number of ANN input/output variables is comparatively easy to determine. According to the area of the DM pupil modeled discussed in Section 3, we use the known surface measurements in the 14 x 9 actuators matrix, as the inputs. The output of the ANN is the predicted actuator values in the same actuators. Before fitting an ANN model, the data should be preprocessed. There are basically two reasons for preprocessing. Firstly, preprocessing can ensure that all variables receive equal attention during the training process. Otherwise, input variables measured on different scales will dominate training to a greater or lesser extent because initial weights within a network are randomized to the same finite range [19]. Secondly, preprocessing is important for the efficiency of training algorithms. For example, the gradient descent algorithm (error backpropagation) used to train the MLP is particularly sensitive to the scale of data used. Due to the nature of this algorithm, large values drag training because the gradient of the sigmoid function at extreme values approximates zero [19]. In general, there are fundamentally two types of preprocessing methods. The first one is to rescale the data to a small interval (referred to as rescaling), such as $[-1, 1]$ or $[0, 1]$, depending on the transfer (activation) function used in the neurons, because some transfer functions are bounded (e.g. logistic and hyperbolic tangent function). The second method is to
standardize the data by subtracting the mean and dividing by the standard deviation to make the data have a mean of 0 and variance 1 (referred to as standardization).

In order to reduce the dimensionality of the problem, we designed a simple strategy to ‘clone’ the ANN models, which relies on the isotropic behavior of MEMS DMs. We experimented with two ANN models of different sizes:

- **ANNs** (small ANN): MLP with one hidden layer and ANN structure: 12 x 16 x 12 neurons
- **ANNb** (big ANN): MLP with one hidden layer and ANN structure: 30 x 40 x 30 neurons

As an example, the cloning strategy for ANNb is illustrated in Figs. 2 and 3. We can see the DM modeled area of \(14 \times 9 = 126\) actuators (the modeled area is fully described in the next section), which has been divided in 7, \(6 \times 5\) actuators sectors, illustrated in different colors (except purple, used to denote main overlapping regions). Each sector is modeled by an ANNb model (of \(6 \times 5 = 30\) neurons in the first layer). The different sectors must overlap in order to give continuity to the model. The ANNb model is trained using data from the central sector solely (in white in Fig. 2).

![Fig. 2. Topology of DM actuators and the cloning structure of ANNb, with actuators named from 1 to 126. The actuators in purple represent overlapping actuators.](image)

The cloning consists of copying a trained ANN to a different sector. This is the case for all colors different than white in Figs. 2 and 3. The \(6 \times 5\) box is positioned at the different sectors to define the inputs for the ANN, which produces the outputs to apply at that particular sector. In overlapping actuators, a decision is taken on using a particular one ANN for modeling those actuators, or taking the average of all overlapping outputs. Figure 3 illustrates the input actuators for each clone ANN, using the same colors as Fig. 2. A similar strategy was followed for the ANN small, although it is more complex to illustrate, so it is not included here.
When training a model, DM surface data as well as actuator voltages are generated 'estimated' or 'predicted' actuator voltages. The training and estimation processes (illustrated in Fig. 4) are fundamentally different in terms of data flow. When training a model, DM surface data as well as actuator voltages are fed to the model's training algorithm. Once trained, the model is fed with DM surface data, to generate 'estimated' or 'predicted' actuator voltages.

![Fig. 3. The actuators structure is illustrated for 6 cases, with a different 6 x 5 DM actuators sector in each one. The seventh actuator is the one at the middle of the white box in Fig. 2.](image1)

### 2.4 Training

All modeling efforts for this article were undertaken using the R programming language [10]. The training and estimation processes (illustrated in Fig. 4) are fundamentally different in terms of data flow. When training a model, DM surface data as well as actuator voltages are fed to the model's training algorithm. Once trained, the model is fed with DM surface data, to generate 'estimated' or 'predicted' actuator voltages.

![Fig. 4. Training and estimation processes.](image2)

The ANN training is fundamentally a problem of nonlinear optimization, which minimizes the error between the network output and the target output by repeatedly changing the values of ANN's connection weights according to a predetermined algorithm. Error backpropagation...
[20] is a well-accepted algorithm for optimizing feedforward ANNs term. In this study, training was implemented using the AMORE function in R, which uses the error backpropagation algorithm by updating the weight and bias values according to gradient descent with momentum. Networks trained with the backpropagation algorithm are sensitive to initial conditions and susceptible to local minima in the error surface. On the other hand, there may be many parameter sets within a model structure that are equally acceptable as simulators of a dynamical process of interest. Consequently, instead of attempting to find a best single ANN model, we may make predictions based on an ensemble of neural networks trained for the comparable task (see e.g [21]). In this present study, the idea of ensemble prediction is adopted, and the simple average ensemble method is used. Each ANN model is trained five times so as to get 5 networks, and the best one is chosen according to their training performance.

A key issue about ANN training is how to decide when to stop the training because ANN are prone to either underfitting or overfitting if their trainings are not stopped appropriately. The cross-validated early stopping technique is most commonly used [22] to avoid stopping too late (i.e. resulting in overfitting). However, Amari et al [23] showed that overfitting would not occur if the ratio of the number of training data sets to the number of the weights in the network exceeds 30. In such cases, training can be stopped when the training error has reached a sufficiently small value or when changes in the training error remain small. In fact, some experiments [24] with simulated time series show that, even when the ratio is as high as 50, very slight overfitting still can be observed in some cases. But cross-validation generally does not help to improve, in many cases even degrades, the generality of ANNs when the ratio is larger than 20. In our study, the ratio of the number of input data sets and the number of network weights is far larger than 50, therefore, the use of cross-validation data was not considered necessary. The training is stopped after 10000 epochs. For all of our training, we have used the R language and environment for statistical computing [10], running on a 58 GFLOPS Cray XD-1 supercomputer. When the machine was mostly available to us, a MARS model training takes 3 hours, while a ANN model training takes around 24 hours to run.

3. Experimental setup and methodology

Our MEMS DM is a Boston Micromachines ‘Kilo DM’, with 32 x 32 actuators (1020 actuators in total – the corner actuators are not present). We measured the DM surface with a ‘Fisba µPhase 2 HR’ Twymann-Green interferometer, coupled to a 10 mm telecentric lens. The light source used by the interferometer is a temperature-stabilized 632 nm laser. As shown in Fig. 5, the setup is in single reflection. We measured the repeatability of our interferometer to be 6 nm in single-pass configuration, which is the one used in our experiments.

We settled for modeling an area of the DM pupil, consisting of 14 x 9 actuators (total: 126). There are various reasons to model this sector:

• There were a few unresponsive actuators in other areas of the DM, which we did not want to incorporate in the modeled area

• The modeled area is similar in size with respect to smaller DMS (Boston’s Multi-DM with 12 x 12 actuators), so its size is appropriate

• The duration period for training would stay under control, since the experiments performed for this article are intended to be a proof-of-concept of non-parametric estimation techniques

• We can have a static ring of the DM surface around the modeled area, where we can sample long-term drift effects we have seen with our interferometer. This is described in detail in Guzmán et al [6]
Figure 6 shows a typical phase map from the interferometer, where we over plotted the position of the actuators as well as the ring of 8 points which we used to sample the long-term drifts. These 8 points are used to fit a plane which is subtracted from the original phase maps, in order to compensate for the drifts we observed throughout our experiments.

We used the actuator positions to sample the DM surface for our model. Each actuator coordinate (in phase map space) was found by fitting a 2D Gaussian profile to each influence functions [6]. The positions found are plotted with a ‘+’ mark in Fig. 6.

The typical deformation we applied to the DM was around ± 700 nm. The maximum excursion we can measure is limited by the ability of our interferometer to unwrap the phase information from the interference fringes, which in turn is limited by the spatial resolution of the fringe sampling. We ‘raise’ the DM to half its range prior to starting a run and then we apply positive and negative commands around this plateau, in order to model the DM in accordance with AO system requirements. Our interferometer was calibrated for each run with this plateau position, which was subtracted automatically for all phase maps. This is why our data have a zero mean value, around which there are positive and negative excursions.

The non-parametric estimation techniques we used need to be exposed to as much information as possible about the process being modeled. In practice, there are infinite combinations of actuator positions, which would make data gathering and later training
impracticable; therefore we consider a statistical sample of this universe, commanding random shapes on the DM, using a purely random generator. We settled on this approach because we are interested in learning as much as possible about the DM behavior, in terms of spatial frequency response as well as voltage versus final position of the surface. It is certainly more stringent to use purely random voltages on the actuator than using Kolmogorov-behaved patterns, since the latter have a colored spectral pattern which limits exposing the model to all spatial characteristics. We trained our models with 12,000 random surface positions.

4. Figures of merit

It is relevant to define the figures of merit which we will use to assess the quality of our results. The residual error is the difference in position between the commanded position for the DM surface – computed by the AO computer – and the real position achieved by the DM, given the set of voltages the model calculated and applied to the DM. A standard statistical figure to assess the level of error in the residual is the root-mean-square (rms).

In a previous article [7], the residual error produced by a DM open-loop model is quantified solely in terms of nanometer rms. We believe it is more appropriate to refer this error with respect to some other parameter in the experiment. For example, it is not equivalent to incur in 10 nm of error when commanding the DM surface for a 100 nm peak-to-valley excursion than it is for 1000 nm peak-to-valley excursion. We have defined the following two figures of merit to weigh the real effect of the residual error.

4.1 Ratio residual – desired peak-to-valley

This figure of merit relates the residual error with the peak-to-valley excursion of the desired wavefront correction:

\[
\frac{\text{Residual}_{\text{rms}}}{\text{Desired}_{\text{PV}}} \quad (9)
\]

4.2 Ratio residual – desired rms

This figure of merit relates the residual error to the rms value of the desired wavefront correction:

\[
\frac{\text{Residual}_{\text{rms}}}{\text{Desired}_{\text{rms}}} \quad (10)
\]

The first figure of merit was used in [6], but we have added the second index in order to compare our results to those in Blain et al [9].

5. Results

Once the models were trained, they were tested by requesting different DM shapes. In this section, we present the results.

5.1 Focus term

We sampled a Zernike focus term at the actuator positions and fed the models with this surface data. Each model drew a focus term on the mirror, which we recorded using the interferometer. Figure 7 and 8 present slices of the phase map for each of the models.
Fig. 7. Slice in the X axis for a Zernike focus term from each of the three models implemented. Actuator locations indicated by diamond marks.

We found minor amplitude differences between the models for the same request in surface position, which can be calibrated out in the final model. In order to do fair comparisons, we adjusted the theoretical focus term to the model output, so there are no systematic errors in the results. In order to obtain the residual errors when drawing the focus term – avoiding boundary effects – they were calculated in a sub-region of the modeled area.

Table 1 summarizes the residual errors for each of the models. We only used the index in Eq. (9), since we are interested in comparing our results to those in Stewart et al [7], yielding a marginally better performance (2.7% against our 3.1%).

The MARS model gives the best results for this test, which might be due to the smooth character of the requested wavefront. As we will see in the next set of results, MARS can produce lower residuals in general, but it does not necessarily mean it will produce lower errors when running a DM in a MOAO system.

Table 1. Residual errors when commanding a Zernike focus terms, for the three different models

<table>
<thead>
<tr>
<th>Model</th>
<th>Focus term peak-to-valley (nm)</th>
<th>Residual error peak-to-valley (nm)</th>
<th>Residual error rms (nm)</th>
<th>Residual\textsubscript{Desired, PV}</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARS</td>
<td>613.5</td>
<td>108.0</td>
<td>19.0</td>
<td>3.1%</td>
</tr>
<tr>
<td>ANNs</td>
<td>562.8</td>
<td>157.2</td>
<td>31.7</td>
<td>5.6%</td>
</tr>
<tr>
<td>ANNb</td>
<td>524.5</td>
<td>103.3</td>
<td>23.7</td>
<td>4.5%</td>
</tr>
</tbody>
</table>
5.2 Random phases

The final and more stringent experiment for testing the models was to feed independent random data for validation and let the model generate actuator voltages, which are then run in the DM to obtain new prediction surface data. The difference between validation and prediction runs is the residual error we discussed earlier. The first validation run is equivalent to the requested set of wavefront corrections from the AO computer. The second prediction run corresponds to the actual positions of the DM, using the model output voltages.

The validation run took 3,000 new random combinations, which we think is an adequate statistical base to draw conclusions from. Using both figures of merit defined above, we plot the results in Fig. 9, summarizing them in Table 2. All results in Fig. 9 were obtained using the same random sequence of demands to the DM.

Consistent with the focus term experiment, the MARS model produced the best results in terms of lowest ‘DC’ residual error. However, it also produced a higher number of outliers or inconsistent residuals as seen in the top panels of Fig. 9. In that sense, both ANN models have a lower ‘trial-to-trial’ error. Indeed, the standard deviation of the residuals in the case of the MARS model is 2.2%, while the ANNb produces a standard deviation of only 1%.

The larger trial-to-trial errors in the MARS model are caused by an inaccurate modeling of actuators placed at the frontier of the modeled area. Such behavior is expected because those actuators are surrounded by inactive actuators (unlike inner actuators) and the algorithm does not have information about those inactive actuators. ANN produced better results, learning this particularity with more accuracy, resulting in lower trial-to-trial errors. Nevertheless, the cloning strategy used for the ANN (described in Fig. 2) produces the higher DC error.
Fig. 9. Residual errors for each of the models, expressed in terms of both figures of merit. Left panels use Eq. (9) and right panels Eq. (10). Top panels come from MARS model; central panels ANNs model and lower panels ANNb model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Desired excursion PV (nm)</th>
<th>Desired excursion RMS (nm)</th>
<th>Residual RMS (nm)</th>
<th>Residual RMS (\text{PK-PK} )</th>
<th>Residual RMS (\text{RMS} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARS</td>
<td>1320.8</td>
<td>320.0</td>
<td>40.6</td>
<td>3.1%</td>
<td>12.7%</td>
</tr>
<tr>
<td>ANNs</td>
<td>1339.0</td>
<td>327.5</td>
<td>63.2</td>
<td>4.7%</td>
<td>19.3%</td>
</tr>
<tr>
<td>ANNb</td>
<td>1303.6</td>
<td>321.2</td>
<td>46.8</td>
<td>3.6%</td>
<td>14.6%</td>
</tr>
</tbody>
</table>

6. Conclusions

We have presented three non-parametric estimation models which behave adequately when commanding a MEMS DM. The models are capable of reproducing complex shapes on the facesheet, with similar or better results than those found in the literature.

These types of models do not need a theoretical development as physical models do, but only access to a large set of training data. The complexity of the model depends directly on the size of the DM to be commanded and it represents the main burden of this technique. Nevertheless, the cloning strategy presented for the artificial neural networks is reliable and it can be considered a solution, in particular for larger mirrors.
It is expected that the membrane of MEMS mirrors would suffer from stretching over long period of times, although this is not specified by the manufacturer. A re-training process may be required if this stretch is significantly.

For astronomical applications (such as MOAO), where open-loop AO is combined with integral field spectroscopy at a spatial sampling well below the diffraction limit, then a low DC error, such as that obtained with MARS, may be preferred. This is because the occasionally larger wavefront excursion will not substantially affect the coupling of light to spatial sampling elements (spaxels), which is the primary scientific performance metric. A non-astronomical example of such a case might be an open-loop laser correction system aiming to increase “power in the bucket” in a relatively low beam quality system (such as a multimode laser). A counter example might be a high contrast imager operating in a specific spatial frequency band, where a low trial-to-trial error would be more important.

DM calibration is an issue in a real MOAO system, since it works in an optical open-loop. A model such as any of the ones presented in this article can be easily incorporated into the MOAO computer, which has to quantify the correcting effect of the DM for each MOAO channel, as part of the calibration process.

Acknowledgments

We appreciate fruitful discussions with Prof. Ray Sharples and Dr. Tim Morris, from Durham University. We also thank the reviewers for very useful suggestions. D. Guzman appreciates support from the Science and Technology Facilities Council (STFC), through the 'Dorothy Hodgkin' postgraduate studentships program. This work was partially supported by the Chilean Research Council grant Fondecyt-1095153 and by the Science and Technology Facilities Council (STFC) grant PP/E007651/1.