An Optimization Algorithm Based on Active and Instance-Based Learning

Olac Fuentes and Thamar Solorio

Instituto Nacional de Astrofísica, Óptica y Electrónica
Luis Enrique Erro 1
Santa María Tonantzintla, Puebla, México 72840

Abstract. We present an optimization algorithm that combines active learning and locally-weighted regression to find extreme points of noisy and complex functions. We apply our algorithm to the problem of interferogram analysis, an important problem in optical engineering that is not solvable using traditional optimization schemes and that has received recent attention in the research community. Experimental results show that our method is faster that others previously presented in the literature and that it is very accurate for the case of noiseless interferograms, as well as for the case of interferograms with two types of noise: white noise and intensity gradients, which are due to slight miss alignments in the system.

Key Words:
optimization, active learning, instance-based learning, locally-weighted regression

1 Introduction

Optimization in poorly modelled, noisy and complex domains is an important problem that is faced in many scientific and engineering areas. In many such domains, traditional optimizations algorithms such as the Simplex method, or the Levenberg Marquardt algorithm do not yield satisfactory results and are usually very sensitive to the starting search points provided by the user. For these reasons, non-traditional optimization algorithms, including simulated, genetic algorithms, evolution strategies, and hybrid evolutionary-classical algorithms have been proposed. While good results have been reported, the running times of these algorithms are often high, and they are not well suited to all domains. Thus, more efficient and complementary algorithms are desirable.

In this paper we propose an efficient algorithm to perform optimization in complex domains. Our algorithm is based on the observation that the candidate solutions generated by an optimization algorithm, which are normally discarded by both traditional and non-traditional optimization schemes, can be used as a training set for a learning algorithm, which in turn can predict the parameters of an optimal solution to the problem. An advantage of this approach is that, if we want to find the solutions to several similar problems, we can process them concurrently and integrate all the candidate solutions in a single training set. Since the training set is continuously changed, we require a learning algorithm that requires a small training time. Instance-based learning algorithms, whose training consists of simply storing the training data, fulfill this requirement. In our work we used the locally-weighted regression algorithm, an instance-based learning algorithm that has been found to yield similar accuracy as neural networks in many application domains while preserving the short training times inherent to this class of methods.

We illustrate out method with an application to the problem of interferogram analysis, which as received recent attention in the literature. This is an interesting domain, as there are published attempted solutions using both traditional optimization schemes and evolutionary algorithms.

The organization of the remainder of this paper is as follows: In Section 2 we describe the procedure we used to generate a set of noisy simulated interferograms. Section 3 presents the methods we used, including principal component analysis and locally-weighted regression. Section 4 gives a detailed description of our algorithm. In Section 5 the main results are shown, and Section 6 presents conclusions and suggest directions for future work.

2 The Optimization Algorithm

Normally, machine learning algorithms are passive in the sense that they just use the training data that are provided. Alternatively, an algorithm can be active, selecting the examples that are deemed most useful to be added to the training set. Moreover, the algorithm itself can produce those examples that are going to augment the training set. This can be more easily explained if we consider how instance based learning algorithms perform prediction tasks. What these algorithms do to predict the target function value for a new example is to select from the training set the most similar points to the new example and then, by some average metric, output the target value for the query. In the case of interferogram analysis, we can reverse the process and obtain a new instance based on the algorithm's output and measure how similar this new instance is to the query one (that is, we can easily obtain a new interfergram from the predicted value of the vector of aberration coefficients $v = [A, B, C, D, E, F]$) and the vector of intensity gradients $\gamma = [\gamma_1, \gamma_2, \gamma_3]$. If the difference between the newly obtained interferogram and the one under test is high, we can add this new instance to the training set and allow the query instance to remain in the test set until the algorithm's output is considered similar enough. By doing so, the algorithm is able to use new instances that are presumably closer neighbors to the query point than the original training examples. See Table 1 for an outline of our algorithm.

What we are trying to do with this approach is to overcome one disadvantage of instance based-learning algorithms. When the query point lies too far from the training examples, it's difficult for the algorithm to output a target value

I_s is a matrix whose rows are vectors of interferogram images
A_s is the vector of aberration coefficients
γ_s is the vector of intensity gradients
S is the training set, given by the tuple $[I_s, A_s, \gamma_s]$
T is the test set of interferogram images

1. While $T \neq \emptyset$ do:
   - Build C, the classifier, using S and LWLR
   - Use C to predict the aberration coefficients vector V and the intensity gradients vector γ for the test interferograms
   - Compute I_t, the interferograms that generated the predicted aberrations and gradients in V and measure similarity with T
     - Let N be the interferograms for which the similarity was greater than some threshold, do:
       $T = T - N$
       $I_t = I_t - N$
     - Add I_t together with their corresponding aberrations and gradients vectors to the training set:
       $S = S \cup [I_t, V_t]$
2. End

Table 1. The optimization algorithm

accurate enough. However, it is likely that whatever the predicted value is, it will lie in the vicinity of the query point, presumably nearer than any of the training examples. By adding the new instance that was created based on the algorithms output, and repeating this step several times, eventually the algorithm will be provided with the needed examples to output a very accurate value.

In the problem we are trying to solve here, interferogram analysis, the algorithm tries to predict the intensity gradients and the aberration coefficients for a given interferogram. Once the algorithm has predicted these parameters we can easily compute the corresponding interferogram. By comparing the images of the two interferograms we can determine if this prediction is good enough, if this is not the case, then the new generated interferogram is added to the training set while the test one remains in the test set.

3 Application: Interferogram Analysis

Interferometry is a laboratory technique very commonly used to test the quality of optical systems. To perform interferometry, two beams, one passing through a reference surface and the other passing through the test surface, are combined an made to interfere, which results in a pattern, called interferogram, that characterizes the quality of the test surface. A schematic diagram of a simple

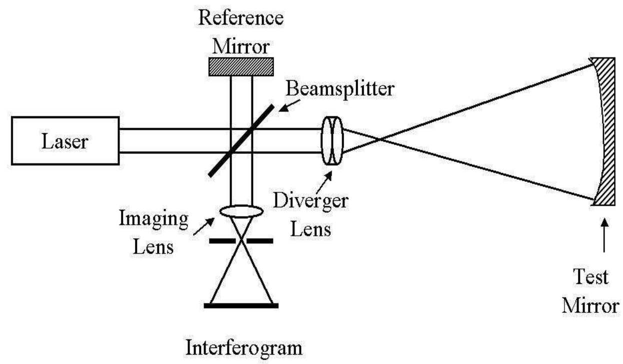


Fig. 1. A simple interferometer

interferometer is shown in figure 1. Experienced technicians can diagnose the flaws of the test surface by careful analysis of the interferogram, however, this is a time consuming task, and when there is a need to analyze more than a few interferograms, it becomes impractical. Thus, there is a need for techniques to automate this process.

The problem of automatically characterizing an interferogram has received recent attention in the literature. This is a difficult problem, and traditional optimization schemes based on the least-squares method often provide inconclusive results, specially in the presence of noisy data [1, 2]. For this reason, non-traditional optimization schemes, such as evolutionary algorithms [3, 4], have been proposed to solve this problem [5]. While evolutionary algorithms provide very accurate results in the case of both noiseless and noisy data, their running time is high, taking several minutes to analyze a single interferogram. Clearly, if we need to analyze a large number of interferograms, this approach becomes unfeasible.

3.1 Interferogram Simulation

To obtain the simulated interferograms we use Kingslake's formulation, where the intensity in the interferogrametric image is given by

$$I(x, y) = A + B\, cos(\frac{2\pi}{\lambda}\, W(x, y)) + N(x, y)$$

where λ is the wavelength of the light source, N is the noise which we represented as a random number obtained from a normal distribution with zero mean, and $W(x, y)$ is the optical path difference (OPD) between the reference and test surfaces, and it is represented by a polynomial using the Seidel aberration formulation:

$$W(x,y) = A(x^2+y^2)^2 + By(x^2+y^2) + C(x^2+3y^2)$$
$$+D(x^2+y^2) + Ey + Fx$$

where A is the spherical aberration coefficient, B, C and D are the comma coefficient, astigmatism and defocusing coefficients, respectively, E is the tilt about the y axis, and F is the tilt about the x axis.

Clearly, it is easy to obtain an interferogram I given the vector of aberration coefficients $v = [A, B, C, D, E, F]$. However, we are interested in the inverse problem, that is, obtaining the vector of aberration coefficients from the corresponding interferogram, which, as mentioned before, is a very difficult optimization problem. Section 4 will describe our proposed solution to this problem.

4 Experimental Results

In this paper we propose an efficient algorithm to find the phase of a set of interferograms. We use a version of the well-known locally weighted regression algorithm [6], modified to automatically incorporate new available data in regions of the parameter space where the available data are too sparse, or the target function is too complex, to generate accurate predictions. To further reduce the running time, we use a principal component analysis preprocessing stage to compress the high-dimensional interferograms into a more manageable size with minimal loss of information.

4.1 Principal Component Analysis

The formulation of standard PCA is as follows. Consider a set of M objects $O_1, O_2, \ldots, O_M$, where the mean object of the set is defined by

$$X = \frac{1}{M}\sum_{n=1}^{M} O_n \tag{1}$$

Each object differs from the mean by the vector

$$\theta_i = O_i - X \tag{2}$$

Principal component analysis seeks a set of M orthogonal vectors v and their associated eigenvalues k which best describes the distribution of the data. The vectors v and scalars k are the eigenvectors and eigenvalues, respectively, of the covariance matrix

$$C = \sum_{i=1}^{M}\sum_{n=1}^{M} \theta_n \theta_n^T = AA^T \tag{3}$$

where the matrix $A = [\theta_1, \theta_2, \ldots, \theta_M]$

The associated eigenvalues allow us to rank the eigenvectors according their usefulness in characterizing the variation among the objects.

Then, this module takes as input the set of interferograms (or images), and finds its principal components (PCs). Then the interferograms are projected onto the space defined by the first 30 principal components, which were found to account for about 90% of the variance in the set, and the magnitudes of these projections are used as attributes for the next stage in the system. That is, the projection *Proj* is given by

$$Proj = PCs^T (T - M) \tag{4}$$

where T is the training set of images and M is the mean image.

4.2 Locally-Weighted Regression

Locally-weighted regression belongs to the family of instance-based learning algorithms. In contrast to most other learning algorithms, which use their training examples to construct explicit global representations of the target function, instance-based learning algorithms simply store some or all of the training examples and postpone any generalization effort until a new instance must be classified. They can thus build query-specific local models, which attempt to fit the training examples only in a region around the query point. In this work we use a linear model around the query point to approximate the target function.

Given a query point $\mathbf{x}_q$, to predict its output parameters $\mathbf{y}_q$, we assign to each example in the training set a weight given by the inverse of the distance from the training point to the query point: $w_i = \frac{1}{|\mathbf{x}_q - \mathbf{x}_i|}$. Let W, the weight matrix, be a diagonal matrix with entries $w_1, \ldots, w_n$. Let X be a matrix whose rows are the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the input parameters of the examples in the training set, with the addition of a "1" in the last column. Let Y be a matrix whose rows are the vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$, the output parameters of the examples in the training set. Then the weighted training data are given by $Z = WX$ and the weighted target function is $V = WY$. Then we use the estimator for the target function $\mathbf{y}_q = \mathbf{x}_q^T (Z^T Z)^{-1} Z^T V$.

Thus, locally weighted linear regression (LWLR) is very similar to least-squares linear regression, except that the error terms used to derive the best linear approximation are weighted by the inverse of their distance to the query point. Intuitively, this yields much more accurate results than standard linear regression because the assumption that the target function is linear will not hold in general, but is a very good approximation when only a small neighborhood is considered.

5 Experimental Results

In this section we describe the experiments performed with our optimization algorithm applied to the problem of predicting the aberration coefficients vector and the intensity gradients vector. First, we generated a thousand aberration

Coefficients	Mean Absolute Error	Standard Deviation
A	3.9286e-003	3.5283e-004
B	4.5149e-003	6.8457e-004
C	1.5848e-003	2.0088e-004
D	2.6554e-003	2.3317e-004
E	5.4078e-004	1.2607e-004
F	1.0868e-003	9.6457e-005
γ_1	0	0
γ_2	0	0
γ_3	0	0

Table 2. Mean Absolute Errors for Simulated Interferograms

Coefficients	Mean Absolute Error	Standard Deviation
A	3.9652e-002	1.4999e-003
B	4.2934e-002	1.8648e-003
C	1.7704e-002	5.2013e-004
D	3.3045e-002	9.5258e-004
E	7.0914e-003	4.1191e-004
F	1.6100e-002	7.3746e-004
γ_1	0	0
γ_2	0	0
γ_3	0	0

Table 3. Mean Absolute Errors for Noisy Simulated Interferograms

vectors, a thousand intensity gradients vectors and their corresponding interferograms. Then we randomly divided the data into ten equally sized subgroups, one group was used for testing and the remainder nine where considered the training set. Ten different experiments were performed, each one using a different group for testing. We repeated this procedure ten times, and the overall average are the results presented here. Table 2 shows averaged mean absolute errors and standard deviations for each aberration coefficient as well as for each intensity gradient. Figure 5 shows three pairs of interferograms, column a) corresponds to the original simulated interferograms, while column b) shows the interferograms that generated the aberration coefficients predicted by our optimization algorithm. As can be seen, the interferograms are practically identical.

As the experimental results show, our method is very accurate with the simulated interferograms. However, real data always pose the challenge of managing noise. In order to evaluate the noise sensibility of our method, we performed experiments on interferograms with simulated noise. In this experiment we also performed ten runs of the procedure described previously, the only difference being that we added noise to the test interferograms. The noise is simulated as

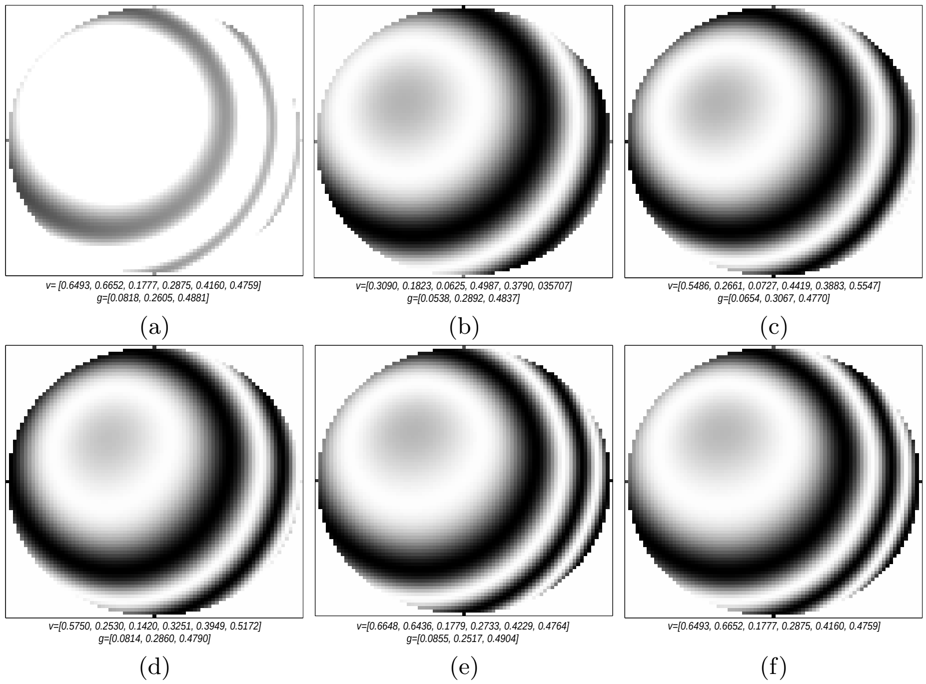


Fig. 2. A detailed trace of the optimization algorithm. The original test interferogram is the figure on the top-left of the figure, the immediate left figure is the result of the traditional LWLR. The following figures to the original test interferogram. The last figure (bottom-right) is how the test interferogram will look if we eliminate the gradient, it is very difficult to distinguish between this image and the one obtained by our optimization technique

a Gaussian error, depending only on a mean and a standard deviation. Table 3 shows errors in aberration coefficients. In Figure 4 we can see a visual comparison between noisy interferograms and the interferograms obtained from the predicted aberrations. It can be seen that our method's performance was not damaged by the noise in the test data.

In Figure 2 we present a detailed trace of a randomly chosen test example. What we see in this figure is how the newly generated training examples are helping the algorithm of LWLR to better approximate the parameters. The original test interferogram is the figure on the top-left of the figure, the immediate left figure is the result of the traditional LWLR. While our optimization algorithm begins iterating we can see that the following images are moderately changing with each iteration until the termination criterion is met, the final figure is very similar to the original test interferogram. The last figure (bottom-right) is how the test interferogram will look if we eliminate the gradient, it is very difficult to distinguish between this image and the one obtained by our optimization tech-

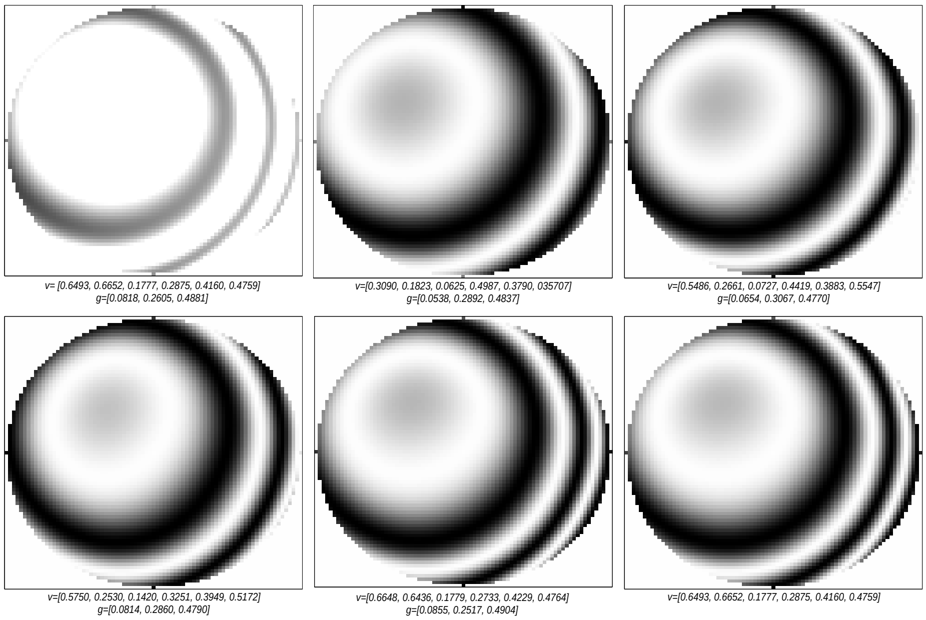


Fig. 3. Column (a) Shows three generated interferograms used for testing while column (b) shows the corresponding interferograms obtained from the predicted aberration coefficients.

nique. Figure 3 presents a similar trace, only this time we are using for testing interferograms that have the intensity gradient noise as well as white noise.

6 Conclusions

In this paper we have presented an optimization algorithm that has a very strong feature: the ability of extending the training set automatically in order to best fit the target function for the test data. There is no need for manual intervention, and if new test instances need to be classified the algorithm will generate as many training examples as needed.

We have shown experimental results of using our method to solve the problem, given a large set of interferograms, finding their corresponding vectors of aberration coefficients and intensity gradients. The method yields very accurate results, even in the presence of noise, and also, it is significantly faster than other methods introduced earlier.

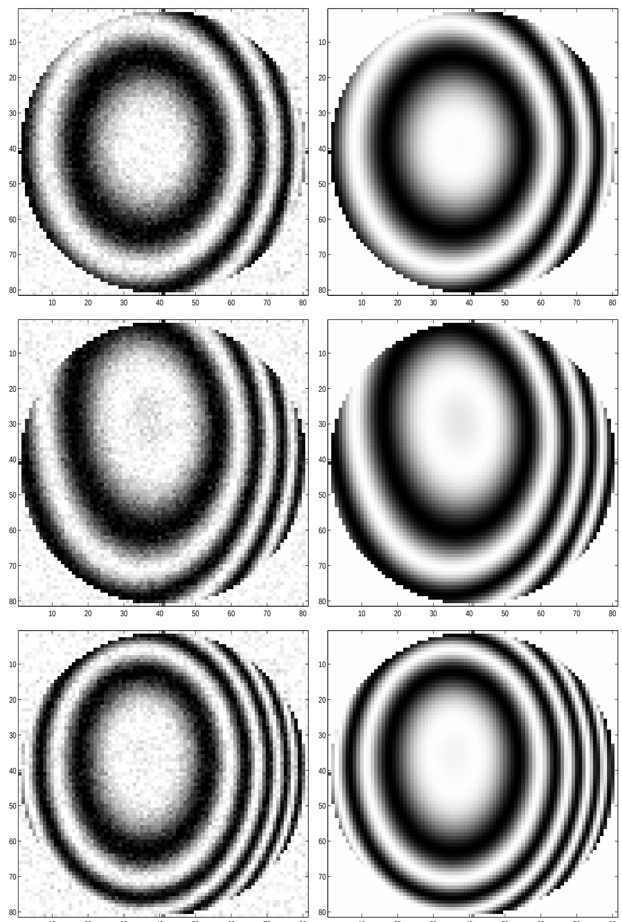
- Testing the method using real interferograms
- Extending the algorithm to handle higher-order aberrations
- Testing the applicability of the method to other optimization problems in optics, as well as in other areas of science.

Acknowledgements

We would like to thank CONACYT for partially supporting this work and Sergio Vázquez y Montiel and Jaime Sánchez Escobar for stimulating disscussions.

References

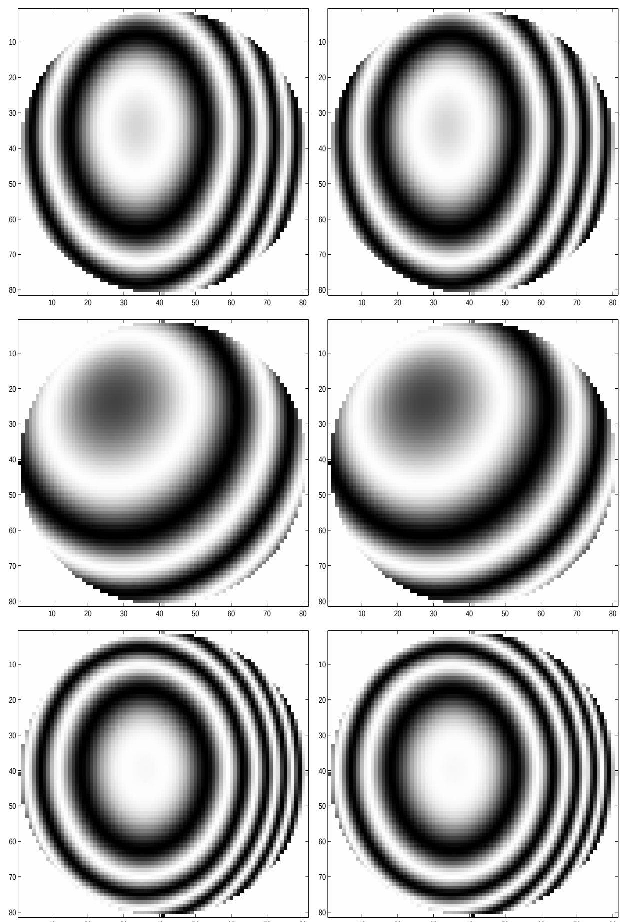
1. D. Dutton, A. Cornejo, and M. Latta. A semiautomatic method for interpreting shearing interferograms. *Applied Optics*, 7:125–131, 1968.
2. J. Y. Wang and D. E. Silva. Wave-front interpretation with zernike polynomials. *Applied Optics*, 19:1510–1518, 1980.
3. T. Bäck, F. Hoffmeister, and H. Schwefel. A survey of evolution strategies. In *Proceedings of the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers, Inc, 1991.
4. T. Bäck and H. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.
5. S. Vázquez y Montiel, J. Sánchez, and O. Fuentes. Obtaining the phase of an interferogram using an evolution strategy, part I. *Applied Optics*, 41(17):3448–3452, June 2002.
6. C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11:11–73, 1997.



a) Simulated noisy interferograms

b) Predicted interferograms

Fig. 4. Column (a) Shows three of the 3,000 noisy interferograms used for testing while column (b) shows the corresponding interferograms obtained from the predicted aberration coefficients.



a) Simulated interferograms

b) Predicted interferograms

Fig. 5. Column (a) Shows three generated interferograms used for testing while column (b) shows the corresponding interferograms obtained from the predicted aberration coefficients.