Particle Swarm Based Model Exploitation for Parameter Estimation of Wave Realizations

Kostas Hatalis, Basel Alnajjab, Shalinee Kishore, Rick S Blum
Dept. of Electrical and Computer Engineering
Lehigh University
Bethlehem, PA 18015
{kmh511,bra212,skishore,rb0f}@lehigh.edu

Alberto Lamadrid
Dept. of Economics
Lehigh University
Bethlehem, PA, 18015
all512@lehigh.edu

***Abstract*—Ocean wave energy farms are composed of several wave energy converter devices. The objective of each converter is to capture the potential and kinetic energy in rolling ocean waves and convert them into electricity. An important task in optimizing power delivery from ocean wave energy farms is short term prediction of incoming ocean waves. Accurate predictions enable predictive wave energy converter control, energy storage control, and condition monitoring. In a novel approach we estimate the parameters of regular ocean waves by exploiting the domain with particle swarm optimization. We estimate initial parameters, and use those as guides in the swarm to estimate better ones. Our approach yields highly accurate estimates of ocean wave parameters that are very close to the Cramer-Rao lower bound on estimating these parameters. Our approach of domain exploitation with particle swarm optimization is also superior to stand alone global optimization methods such as simulated annealing and genetic algorithm.**

I. Introduction

Although renewable energy sources, like wind, solar, and ocean waves, are often thought of as providing "free energy, the system benefits of integrating renewables may be lower than expected because of the uncertainty they introduce into the electricity grid, both in terms of dispatch and reliability. One approach to counter the uncertainty of renewable generation is to employ prediction methods to forecast availability renewable resources to optimize how the resources are converted to electricity and dispatched to the grid [1], [2], [3]. In this paper, we focus on delivering a highly accurate, short term forecasting approach that estimates parameters of incoming ocean waves using noisy sensor estimates in order to optimize operations of ocean wave energy farms.

An ocean wave energy farm contains an array of wave energy converters (WECs) that capture and convert the energy in rolling ocean waves to electricity, which then be delivered to a power system or the electricity grid. Predictive control mechanisms [4] can be used within individual WECs and across multiple WECs to optimize the output power of the wave farm. For example, these schemes can adapt the feedback force, relative motion, dampening, etc., within WECs based on incoming wave conditions to improve WEC power production. Additionally, WECs can have on-board energy storage and the farm may have centralized energy storage to offset any volatility in current and future power production. Thus, decisions regarding predictive control and energy storage rely heavily on accurate predictions of incoming ocean wave characteristics. For this reason, ocean wave farms will also require a network of sensors deployed around wave farms to provide (noisy) estimates of incoming ocean waves. These estimates must then be combined and used to extrapolate reliable estimates of incoming wave conditions at WEC sites. As noted above, the predictions will allow novel predictive control methods and storage decisions that increase the efficiency and consistency of ocean wave power production.

In this paper, we develop in particular an adaptive particle swarm optimization (PSO) approach to estimate parameters of regular ocean wave realizations. The main contributions of this paper are:

1) We design a strategy for PSO to exploit the model structure of our problem.
2) We propose two methods for emphasizing exploration with a new inertia weight and updating worst particles through local evolution to enhance the performance of PSO.
3) We design experiments to test if our method achieves the best possible estimates.
4) We implement PSO and other evolutionary methods without domain exploitation to compare their efficacy for parameter estimation.

We demonstrate the accuracy of our proposed method using simulated data for regular ocean waves, the model for simulating regular waves is presented in section III. The extension to irregular ocean waves is a direction for future research. In particular, we compare our estimation error to the theoretical lower bound offered by the Carmer-Rao Lower Bound (CRLB) for these ocean environments. Results show tight convergence between our results and the CRLB, indicating highly accurate estimates are possible.

Finally, we note that our proposed PSO approach may have wider applicability. For example, while we focus our attention here on the application of ocean wave parameter estimation for ocean wave energy farms, we note that our contributions can extend to any ocean scenario in which short term prediction of wave conditions are required using wave sensor data. In fact, our scheme would work for estimating parameters of waves in any regular wave field based on noisy measurements available from only a subset of locations (i.e., sensor measurements).

We start our discussion in Section II by providing background on ocean wave prediction. For further background on general ocean wave energy systems and the latest in this emerging area, we refer the reader to [5]. In Section III we present the ocean wave model and sensor measurement models used in our estimation approach. Section IV provides background on parameter estimation theory and outlines the CRLB for the ocean wave estimation problem. We provide background on PSO in Section V and our present our modified PSO in Section VI. Results and our discussion of the simulation results are presented in Section VII. We conclude the paper and present future research directions in Section VIII.

II. Background

Ocean waves have been modeled as plane waves consisting of a sum of sinusoids with different frequencies, amplitudes, wavelengths, directions, and phases [6]. Extensive ocean wave data sets are available from buoys operated by the National Oceanic and Atmospheric Administration, the Irish Marine Institute, or a variety of other entities around the globe. The buoys record the significant wave height $h_{1/3}$ (mean wave height of the highest third of the observed waves) and the wave period T_s by sampling over given time intervals. Together, $(h_{1/3}, T_s)$ can define the sea state and the parameters of an assumed wave spectrum $S(\omega)$, which gives the energy distribution of ocean waves among the different frequency components, ω. Several power spectra are commonly used in marine engineering, such as the Pierson-Moskowitz spectrum. Over time, the sea state changes stochastically in a manner that can be modeled and studied using historical data at any given buoy location. There is an extensive literature on such models of sea waves [7], [8]. These models however presume the availability of sampled-average values of $(h_{1/3}, T_s)$ and thus cannot be readily used for seconds ahead (short term) wave prediction.

Prediction of ocean waves can be done using both time series based methods and physics based models. Time series-based prediction of ocean waves uses techniques such as regression [9], neural networks [10], and genetic algorithms [11] that employ statistics from existing buoy datasets to predict future wave conditions. For example, such methods have been previously studied for ocean wave elevation forecasting using non-linear autoregressive neural networks trained using backpropagation to conduct multistep forecasts of wave power directly from past observed wave heights [12]. Physics-based wave prediction models have been around for a while and have improved in their performance [13]. The state-of-the-art physics-based forecasting tool is the WAVEWATCH III model [14], which uses action balance equations in conjunction with predicted wind speed and air-sea temperature differences to generate wave predictions.

Comparisons of the two techniques [15], [16], [17] show that statistical approaches provide higher accuracy for the short term while physics-based models are more reliable for long-term prediction. The definition of short and long term in these cases depends on the time resolution of the datasets. For example, [16] studies hourly data and finds that statistical approaches are superior for predicting 1 to 10 hours out while physics-based models are accurate up to 80 hours ahead. However, as noted above, since existing times-series based models build off estimates of $(h_{1/3}, T_s)$ and predict statistical parameters of waves, they cannot be used for estimating wave characteristics seconds ahead. Specifically they can not provide exact wave realizations that will be seen seconds ahead at a location in the ocean. In such cases, wave sensors are needed that make real-time measurements of wave elevations, velocities, accelerations, etc., within or near the location of interest. As such both physical and time series models fall short for our prediction task providing motivation for our proposed method.

If we know the wave parameters (amplitude, phase, direction, and frequency) for all components of the ocean wave, then we can write an equation that gives the exact waveform $H(x, y, t)$ that would be observed at a desired location (x, y) on the ocean surface at time t. Of course, we do not know these parameters explicitly, so they must be estimated from noisy sensor data obtained from sensors deployed at locations around (x, y) at some other time $t' < t$. The focus of this paper is to provide highly accurate estimates of amplitude, phase, direction and frequency of incoming regular waves based on noisy sensor data using our proposed PSO scheme.

III. Ocean Wave Model

Since there is a lack of data for different sampling frequencies or sampling rates of ocean wave elevations we simulate our own wave data with additive noise. We use the standard wave equation model with several assumptions [18]. First we assume the ocean is an ideal incompressible fluid with no loss of mechanical energy. Second, we assume the monitoring area in the ocean has sufficient depth such that finite depth effects, other than dispersion, are small. Lastly we assume waves are created by forcing functions that were applied at adequate distances away, such as a distant storm, resulting in the observation of fully developed ocean waves. Under the given assumptions, the form of an ocean waves can be seen as plane waves consisting of a sum of sinusoids with different amplitudes, frequencies, directions, wavelengths and phases.

For simulating of ocean waves we focus on vertical sensors for predicting regular wave formations (due to their simplicity). Under generally well accepted assumptions [6], and under regular wave conditions, the wave elevation for all sensor locations (x, y) on the ocean surface for all times t the exact time waveform which would be observed at a particular point in the ocean can be described by

$$H(x,y,t) = A\cos\left(\frac{\omega^2}{g}\big(x\cos(\beta)+y\sin(\beta)\big)-\omega t+\phi\right), \quad (1)$$

which has the parameters A for the amplitude, ω for the frequency measured in radians per second $(rads/s)$, β for the wave angular direction in radians measured relative to the x-axis, and ϕ for the phase in radians. These parameters, collected together in the parameter vector $\boldsymbol{\theta} = (A, w, \beta, \phi)^\top$, are assumed to be unknown and thus we must provide estimates for their values based on collected sensor measurements. Specifically, we consider the case where we collect a noisy measurement H_m from N sensors with given coordinates $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ collected at times $t = 0, T_s, \ldots, (K-1)T_s$ where T_s is the sampling period and where each sensor provides us with K observations. A measurement made by a sensor located at (x_n, y_n), where $n \in \{1, \ldots, N\}$ at time $t = kT_s$ where $k \in \{0, 1, \ldots, (K-1)\}$, is assumed to be given by

$$H(x_n, y_n, t) = H'(x_n, y_n, t) + \mathcal{V}_n(t) \tag{2}$$

where $\mathcal{V}_n(t)$ is Gaussian white noise for sensor n at time t and H' is the true measurement. The observations from all sensors and all times are collected into the vector

$$\begin{aligned}\mathbf{H} = [&H(x_1, y_1, 0), \ldots, H(x_1, y_1, (K-1)T_s), \ldots \\ &H(x_2, y_2, 0), \ldots, H(x_N, y_N, (K-1)T_s)]^T.\end{aligned} \tag{3}$$

Once we collected sensor data and estimated the parameters accurately, they can replaced back into Eq. (1) to forecast the wave realization at some short time ahead $t + j$.

IV. Parameter Estimation Theory

In parameter estimation, we are given sample data to estimate parameters of the selected distribution. Generally, given sample sensor data $\mathbf{H}$ containing all the K measurements from each of the N sensors, and $t = kT_s$, we assume the noise vector

$$[\mathcal{V}_1(t_0), ..., \mathcal{V}_N(t_0), ..., \mathcal{V}_1(t_K), ..., \mathcal{V}_N(t_K)]^\top \tag{4}$$

is a jointly Gaussian with a zero mean vector and a covariance matrix with $(\sigma_1^2, ..., \sigma_{NK}^2)$ along the diagonal. For simplicity we assume $\sigma_1 = ... = \sigma_{NK} = \sigma$. Under this assumption the joint probability density function of the observations conditioned on θ, called the likelihood function, is $f_{\mathbf{H}}(\mathbf{H}|\boldsymbol{\theta})$. A possible estimate of the unknown $\boldsymbol{\theta}$ could be computed by the well-known Maximum Likelihood (ML) estimate $\boldsymbol{\theta}_{ML}$ which is defined as

$$\boldsymbol{\theta}_{ML} = \arg\max_{\boldsymbol{\theta}} f_{\mathbf{H}}(\mathbf{H}|\boldsymbol{\theta}). \tag{5}$$

For simple convex problems, the ML estimate could be found using derivatives of the objective function. Unfortunately, the considered problem of wave estimation is non-linear and has many local minima and maxima which makes using such methods to successfully estimate $\boldsymbol{\theta}$ a very hard task. As such we provide a way to use particle swarm optimization, a global optimization method that does not require the calculation of derivatives, to solve for $\boldsymbol{\theta}$ by exploiting the domain structure of regular waves in Eq. (1).

A. Cramer Rao Lower Bound

In the following section we briefly introduce the Cramer Rao Lower Bound [19] and provide the general equations for calculating the Fisher information matrix under Gaussian white noise. It is important in deriving the CRLB as it provides us with a theoretical lower bound on the estimation of a wave parameter and thus acts as our main way to asses estimates. To calculate the CRLB we assume a sensor measurement H_m is collected in the vector $\mathbf{H}$. We formulate the CRLB for the form of a wave given in Eq. (1) by collecting the unknown parameters in the $\boldsymbol{\theta}$ vector. We let $\hat{H}(x_p, y_p, t)$ be an unbiased estimator of the quantity we are trying to predict $H(x_p, y_p, t_p)$, which comes from Eq. (1) at a particular location (x_p, y_p) and time t_p. Unbiased estimators are those satisfying the equation

$$\mathbb{E}_{\boldsymbol{\theta}}\left[\hat{H}(x_p, y_p, t)\right] = H(x_p, y_p, t). \tag{6}$$

Then the mean-square error (MSE) of any unbiased estimate must satisfy

$$MSE_{\hat{H}(x_p,y_p,t_p)} \geq qJ(\boldsymbol{\theta})^{-1}q^\top = CRLB_{\hat{H}(x_p,y_p,t_p)} \tag{7}$$

where the quantity on the right hand side is called the CRLB of the estimate $\hat{H}(x_p, y_p, t_p)$. From (7), the CRLB is defined in terms of the row vector

$$q = \left[\frac{\partial H(x_p, y_p, t_p)}{\partial \theta_1}, ..., \frac{\partial H(x_p, y_p, t_p)}{\partial \boldsymbol{\theta}_n}\right] \tag{8}$$

which depends only on the wave characterization we wish to estimate, and the Fisher information matrix $J(\boldsymbol{\theta})$, which depends only on the sensors used to obtain the measurements. We consider Gaussian noise, and the $l - n^{th}$ entry of the Fisher information matrix has a general form given by

$$\begin{aligned}J_{l,n}(\boldsymbol{\theta}) = \sum_{n=1}^{N}\sum_{k=1}^{K}\frac{1}{\sigma_{n,k}^2}\cdots \\ \left(\frac{\partial}{\partial \boldsymbol{\theta}_l}H(x_n, y_n, t)\right)\left(\frac{\partial}{\partial \boldsymbol{\theta}_n}H(x_n, y_n, t)\right)\end{aligned} \tag{9}$$

where we see that calculating each entry will involve the product of two derivatives taken with respect to $\boldsymbol{\theta}$. We have obtained expressions for the derivatives relating to a regular wave characterization described by Eq. (1) in a previous work [20]. These expressions can now be used to find the CRLB for any ocean wave environment described by Eq. (9).

V. Particle swarm optimization

Optimization methods can be categorized into two classes. The first is composed of local optimization methods. This class of methods, used in maximum likelihood estimation, usually requires the evaluation of derivatives of the objective function. The other class of optimization methods is called global optimization. These methods can be roughly classified as deterministic or stochastic. Stochastic global optimization methods tend to converge well to the global optima although there is a lack of strong theoretical guarantees of this. The basic approach is that in each iteration a set of the trials that are thought to be close to being optimal is taken to generate new trials on the next iteration. For deterministic techniques, they are also able to generally achieve a level of confidence that the global optimum will be reached and include methods such as branch and bound, and interval analysis. Overall, there is no algorithm that can solve all global optimization problems with certainty in finite time. The computational resources needed to achieve convergence increases very quickly with the problem size. In general, convergence is also hard to prove for global optimization methods.

Particle swarm optimization is a swarm-based evolutionary computation technique that was developed developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking [21]. PSO is a well known stochastic global optimization method which can be used to find an approximate solution to a problem and shares many similarities with evolutionary computation techniques such as genetic algorithms. The problem is initialized with a population of random solutions and searches for optima by updating generations.

However, unlike genetic algorithms, PSO has no evolution operators such as crossover and mutation. PSO iteratively tries to search candidate solutions called particles in a search space with regard to a given measure of quality around a global optimum. We have a total of P number of particles in the swarm which are initialized with a population of random solutions that will move through a D-dimension space to find new potentially better solutions. A fitness function F is then calculated as a certain measure of quality in reaching a target value, and is typically the mean squared error. Each particle i, where $i \in (1,..,P)$, has a single solution set of parameters we're trying to estimate with each parameter $d \in (1,...,D)$. Every particle is associated with two components, a velocity $v_{i,d}$ and a position $x_{i,d}$. These components are updated on each iteration $\lambda \in (1,...,\Lambda)$ as follows

$$\begin{aligned} v_{i,d}(\lambda+1) = v_{i,d}(\lambda) + c_1 \cdot \epsilon_U \cdot (p_b(i) - x_{i,d}(\lambda)) + \\ c_2 \cdot \epsilon_U \cdot (p_g - x_{i,d}(\lambda)), \end{aligned} \tag{10}$$

$$x_{i,d}(\lambda+1) = x_{i,d}(\lambda) + v_{i,d}(\lambda+1), \tag{11}$$

where Λ is the maximum number of PSO iterations that acts as stopping criteria. The new velocity $v_{i,d}(\lambda+1)$ depends on three terms. The first term is $v_{i,d}(\lambda)$ which is the current velocity. The second part is $c_1 \cdot \epsilon_U \cdot (p_b(i) - x_{i,d}(\lambda))$. The c_1 term is a positive constant called the coefficient of the self-recognition component. The term ϵ_U is a uniformly distributed random number in [0,1]. The $p_b(i)$ vector value is the particle i's best position found so far. The $x_{i,d}(\lambda)$ value is the particle's current position. The third term in the velocity update equation is $c_2 \cdot \epsilon_U \cdot (p_g - x_{i,d}(\lambda))$. The c_2 factor is a constant called the coefficient of the social component. The p_g term is called the global best position and is the best known position found by any particle in the whole swarm so far. Once the new velocity, $v_{i,d}(\lambda+1)$, has been determined, it's used to update the particles position $x_{i,d}(\lambda+1)$. The personal best position of a particle is calculated as such

$$p_b(i) \leftarrow \begin{cases} p_b(i) & \text{if } F(x_{i,D}(\lambda+1)) \geq F(p_b(i)). \\ x_{i,D}(\lambda+1) & \text{if } F(x_{i,D}(\lambda+1)) < F(p_b(i)). \end{cases} \tag{12}$$

And the global best particle position is updated by

$$p_g \leftarrow \begin{cases} p_g & \text{if } F(x_{i,D}(\lambda+1)) \geq F(p_g). \\ x_{i,D}(\lambda+1) & \text{if } F(x_{i,D}(\lambda+1)) < F(p_g). \end{cases} \tag{13}$$

In Eqs. (12,13) the personal and global best particles are updated by checking their fitness solution to the fitness of the updated particle $x_{i,D}(\lambda+1)$. If the new particle yields a better fitness than the global or best particle is updated.

PSO does not require a large number of parameters to be initialized. But the choice of PSO parameters can have a large impact on optimization performance. For most of the practical applications an example good choice of the number of particles P is typically in the range 20 to 40. In the case of very difficult problems the choice can be increased to the range of 100 to 200 particles, however this can significantly decrease runtime. Fine-tuning of the particle acceleration constants c_1 and c_2 can also aid in faster convergence and alleviation of local minima. Usually the choice for these parameters is, $c_1 = c_2 = 2$. We propose an alternative is to choose a larger social component, c_2, and a smaller self-recognition component, c_1, such that it satisfies conditions such as $c_1 + c_2 = 3$. This we believe can further aid in convergence as it emphasizes more on converging on the global best position.

We provide two extensions to PSO to improve convergence. The first is a modification of an inertia weight which we call the stochastic inertia weight, and the second is a process we call local evolution by which we update the worst particles in the swarm to match closer to the best one.

A. Stochastic Inertia Weight

A factor w called the inertia weight [22] is sometimes introduced to improve performance. This adaptive version of PSO updates the velocity of a particle as follows

$$\begin{aligned} v_{i,d}(\lambda+1) = w \cdot v_{i,d}(\lambda) + c_1 \cdot \epsilon_U \cdot (p_b(i) - x_{i,d}(\lambda)) + \\ c_2 \cdot \epsilon_U \cdot (p_g - x_{i,d}(\lambda)), \end{aligned} \tag{14}$$

The inertia weight can play an important role in convergence behavior. It is employed to control the impact of the previous history of velocities on the current one. Accordingly, the parameter w regulates the trade-off between the global (wide-ranging) and local (nearby) exploration abilities of the swarm. A large inertia weight aids in global exploration (searching wide ranging areas), while a small inertia weight aids in local exploration by searching within the nearby areas. To obtain a balance between global and local exploration the number of iterations required to locate the optimum solution is reduced. The inertia weight is usually set as a constant or can be set to decrease over time. There are also several variants of using a dynamic inertia weight [23]. We propose our own version which we call the stochastic inertia weight update as follows

$$w = w_{min} - \epsilon_N (w_{max} - w_{min}) \frac{\lambda}{\Lambda} \tag{15}$$

where λ is the current iteration index, Λ is the predefined maximum number of iterations, and w_{max} and w_{min} are the maximal and minimal weights, and ϵ_N is a normally distributed random number at each iteration. This causes the weight to fluctuate randomly from small to large values over iterations depending on ϵ_N. This is contradictory to the idea of having the inertia weight decrease over time. We believe this way we get increased exploration that allows the swarm to search more broadly over time for new solutions in order to minimize the fitness function.

B. Local Evolution

We also present a method to potentially speed up convergence called local evolution. The core idea behind our heuristic is to evolve "bad" particles closer to the good ones. In every iteration, we record the best-so-far particle in the swarm denoted x^{best}. This is done by finding the global best particle with the highest fitness $x^{best} = p_g$. For any "bad" particle x^{bad}, which has the lowest personal best score, we shift it toward x^{best} using the following update

$$x^{bad} \leftarrow \gamma \times x^{bad} + (1-\gamma) \times x^{best} \tag{16}$$

where we choose γ as a constant between 0 and 1. We apply the process of local evolution to the worst particle but it can also be applied to update a subset of the bottom worst particles.

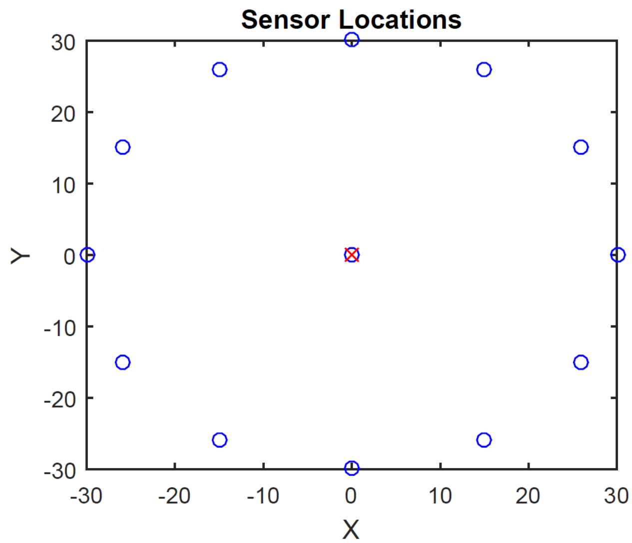


Fig. 1: Twelve example elevation sensors surrounding a WEC at a radius of 30 meters. The WEC is labeled as (0,0) has its own on-board elevation sensor.

VI. Swarm Based Parameter Estimation

In tackling the wave parameter estimation problem we choose PSO due to its simplicity and success in solving similar parameter estimation [24], [25] and signal estimation problems [26], [27]. For testing we first simulate wave elevation with additive white noise. Simulations were done by choosing fixed values for amplitude A, phase ϕ, frequency ω, and direction β. We choose N number of sensors which are positioned on a circle topology with one of the sensors located at (0,0) as shown in Fig. 1. In applying PSO each particle has a dimension of 4 representing the four parameters of a regular wave.

We initially attempted to solve for $\boldsymbol{\theta}$ using PSO presented in section V, but results proved to be poor as their estimates were not near their corresponding CRLB values (results shown as a comparative method in section VII). To improve estimates we set out to exploit the problem domain. We first notice that solving for parameters of a wave form at the sensor located at (0,0), we can ignore estimates for β. This reduces the search space for solving the other parameters. Using measurements from the single elevation sensor at the WEC we can solve for initial estimates of A, ω, ϕ. Then using those estimates we solve for β using all sensor measurements while keeping initial estimates of A, ω, ϕ in Eq. (1) constant. Finally with a good estimate of β we solve again for the other parameters over all measurements. So in estimating parameters of H we call our modified version of PSO, using the stochastic inertia weight and local evolution, three times summarized below:

1) We call Algorithm 1 to get an initial estimate of A, ω, ϕ using measurements from the sensor at $(0,0)$.
2) We call Algorithm 2 to get an estimate of β over all sensor measurements.
3) We call Algorithm 3 to solve again for A, ω, ϕ over all sensor measurements.

In the next four subsections and corresponding algorithms, we show how to estimate $\boldsymbol{\theta}$ in more detail. Overtime wave realizations can change thus their model parameters will change too. However in the very short term on the scale of seconds these parameters will not change providing the ability to forecast seconds ahead. Once parameters are estimated they can plugged back into Eq. (1) to predict the wave elevation at some short time ahead $t + j$.

A. Particle Initialization

For each particle, there may be a certain range within which value of the parameter should lie for better search results. At the very beginning of a PSO run at $\lambda = 1$, the D number of parameters of a particle are initialized randomly somewhere in their feasible numerical range. Therefore, if the d^{th} parameter of the given particle has its lower and upper bound as L_d and U_d, respectively, then we may initialize the d^{th} component of the i^{th} particle as

$$x_{i,d} = L_d + \epsilon_U \cdot (U_d - L_d) \tag{17}$$

Additionally, during the PSO algorithm, if particles are also reinitialized if they outside the bounds of their constraints during an iteration.

B. Step 1: Solving For Initial Estimates At (0,0)

The whole procedure is shown in Algorithm 1. In our first step we only take into consideration measurements from a single sensor which we arbitrarily choose to be the origin. By using only measurements from the origin we can ignore estimating β since $(x, y) = (0, 0)$ which would reduce Eq. (1) to the following realization:

$$\hat{H}(0,0,t) = \hat{A}\cos(-\hat{\omega}t + \hat{\phi}) \tag{18}$$

By ignoring the direction β we are able to get initial estimates for $\hat{A}, \hat{\omega}$, and $\hat{\phi}$. These initial estimates should be close to the true values and thus provide us with a good starting point for future PSO iterations to solve for β and eventually resolve the parameters in step 3.

The algorithm begins by first randomly initializing particles, corresponding velocities and setting the fitness to some large number initially. The first for-loop goes through each PSO iteration. In line 4 the fitness is checked iteratively to see if it's below a threshold (chosen close to 0), if not a second for-loop is executed which first checks if one of the four parameters of a particle do not exceed its threshold which is stored in δ_j. If one of them does then we repeatedly re-initialize that parameter until a new value is given satisfying the constraint. Next estimates for $\hat{A}, \hat{\omega}$, and $\hat{\phi}$ are updated with a candidate solution provided by a given particle. An estimate of wave elevation $\hat{H}_e$ is then made which is used to calculate the fitness for each time step $t \in T$. The goal of PSO is to minimize the fitness function by taking the mean squared error of the estimate $\hat{H}_e$ with the measured value $H_e(0,0,t)$ as follows

$$F(x_i) = \sum_{k=1}^{K} (H(0,0,t) - \hat{A}\cos(-\hat{\omega}t + \hat{\phi}))^2. \tag{19}$$

With the fitness we then update the personal and global best scores $p_b(i)$ and p_g. The stochastic inertia weight is next

Algorithm 1 Solve for initial A, ω, ϕ estimates.

Initialize $\mathbf{x}$ and $\mathbf{v}$, position and velocity vectors.
for $\lambda = 1$ to Λ **do**
  for $i = 1$ to P **do**
    if $F(x_i) > \tau$ **then**
      for $d = 1$ to 3 **do**
        Check δ_d constraint of $x_{i,d}(\lambda)$.
      end for
      $\hat{A} = x_{i,1}(\lambda)$, $\hat{\omega} = x_{i,2}(\lambda)$, $\hat{\phi} = x_{i,3}(\lambda)$
      for $k = 1$ to K **do**
        $\hat{H}(k) = \hat{A}\cos(-\hat{\omega}k + \hat{\phi})$
      end for
      $F(x_i) = \frac{1}{K}\sum_{k=0}^{K}(H(k) - \hat{H}(k))^2$
      Update $p_b(i)$ and p_g.
      Update stochastic inertia weight w.
      Update particle i's velocity and position.
      Update the worst particle with local evolution.
    end if
  end for
end for

Algorithm 2 Solve for β estimate.

Initialize $\mathbf{x}$ and $\mathbf{v}$, position and velocity vectors.
for $\lambda = 1$ to Λ **do**
  for $i = 1$ to P **do**
    if $F(x_i) > \tau$ **then**
      Check δ constraint of $x_i(\lambda)$.
      $\hat{\beta} = x_i(\lambda)$
      for $k = 1$ to K **do**
        $\hat{H}(k) = \ldots$
        $A\cos((\frac{\omega^2}{g})(x\cos(\hat{\beta}) + y\sin(\hat{\beta})) - \omega k + \phi)$
      end for
      $F(x_i) = \frac{1}{K}\sum_{k=0}^{K}(H(k) - \hat{H}(k))^2$
      Update scores, $p_b(i)$, and p_g.
      Update stochastic inertia weight w .
      Update particle i velocity and position.
      Update worst particle with local evolution.
    end if
  end for
end for

calculated which is used to update the velocity and position of each particle. Last we use the principal of local evolution to update the worst particle. The whole process repeats until the fitness falls below the threshold τ.

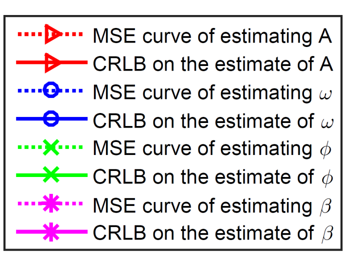


Fig. 2: Legend for Figs. 3 and 4.

Algorithm 3 Resolve for A, ω, ϕ estimates.

Initialize $\mathbf{x}$ and $\mathbf{v}$, position and velocity vectors.
for $\lambda = 1$ to Λ **do**
  for $i = 1$ to P **do**
    if $F(x_i) > \tau$ **then**
      for $d = 1$ to 3 **do**
        Check δ_d constraint of $x_{i,d}(\lambda)$.
      end for
      $\hat{A} = x_{i,1}(\lambda)$, $\hat{\omega} = x_{i,2}(\lambda)$, $\hat{\phi} = x_{i,3}(\lambda)$
      for $k = 1$ to K **do**
        $\hat{H}(k) = \ldots$
        $\hat{A}\cos((\frac{\hat{\omega}^2}{g})(x\cos(\beta) + y\sin(\beta)) - \hat{\omega}t + \hat{\phi})$
      end for
      $F(x_i) = \frac{1}{K}\sum_{k=0}^{K}(H(k) - \hat{H}(k))^2$
      Update scores, $p_b(i)$, and p_g.
      Update stochastic inertia weight w .
      Update particle i velocity and position.
      Update worst particle with local evolution.
    end if
  end for
end for

C. Step 2: Solving For β

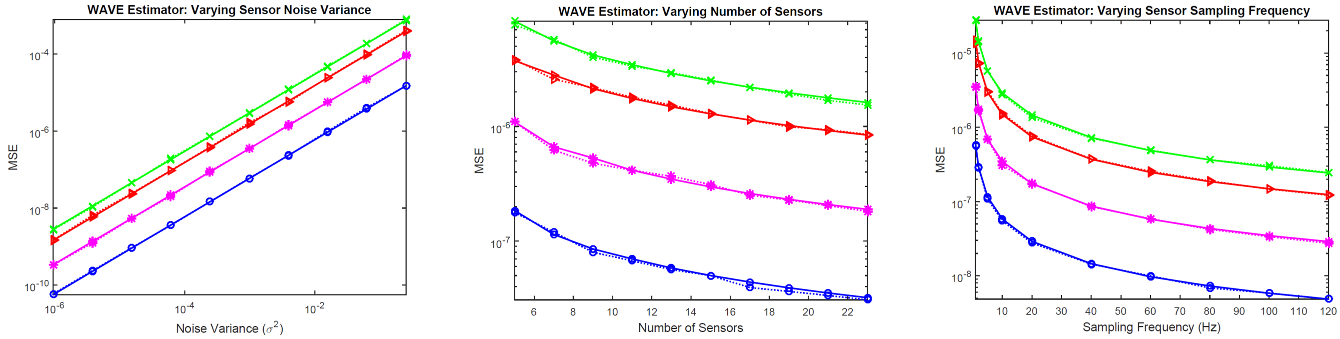
In the second step we calculate the estimate for direction β while keeping A, ω and ϕ fixed to their estimates from step one. The whole procedure is shown in Algorithm 2. We again initialize PSO particles randomly but here they only have a dimension equal to 1 since they only estimate β. Unlike step one we utilize the full function form of $H(x, y, t)$ given in Eq. (1) in the fitness when calculating the MSE. During this run, we use more data from all N sensors in their different locations. Then we proceed to run PSO as in step one by checking constraints, updating $p_b(i)$ and p_g, updating the position and velocity vectors, followed by local evolution on the worst particle.

D. Step 3: Solving For A, ω, ϕ

In step 3, shown in algorithm 3, our estimate for β will be fairly robust from step 2 so we input it as a constant in the function form of $H(x, y, t)$ when calculating the fitness. Since β is constant we then re-estimate $\hat{A}, \hat{\omega}$, and $\hat{\phi}$. We do note that the estimates from step 1 are used as candidate solutions in one of the particles in the swarm. The original estimates are assumed to be good but they were calculated from using a single sensor. We assert that using more data to estimate $\hat{A}, \hat{\omega}$, and $\hat{\phi}$ will increase their accuracy to their true values. The algorithm carries similarly to step 1 and 2 where we initialize particles, calculate the fitness and update velocity and position vectors.

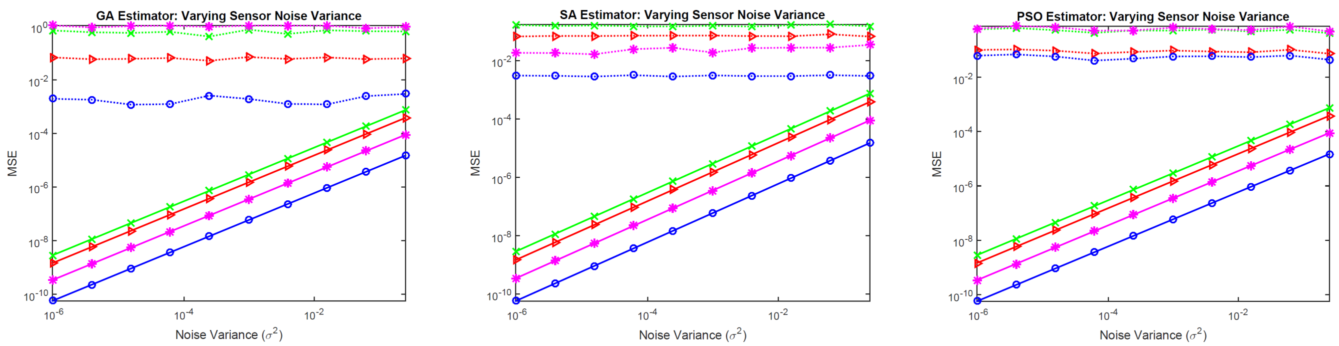
VII. Results and Discussion

Experiments were carried out to estimate the parameters of a regular wave. Keeping everything constant three experiments were carried out by varying the variance of the additive noise to the sensor data, varying the number of sensors surrounding a WEC, and by varying the sensor sampling frequencies. We believe that these three types of experiments are able to properly showcase that our estimation method will be well



(a) (b) (c)

Fig. 3: (a) Testing different noise variances of sensor measurements; (b) testing different number of sensors; (c) testing different values of sensor sampling frequencies.



(a) (b) (c)

Fig. 4: Testing different noise variances of sensor measurements where estimation was done by (a) genetic algorithm, (b) simulated annealing, and (c) vanilla particle swarm optimization.

suited to estimate waves under different conditions. For all estimation experiments we set the constraints of the parameters to have a minimum of 0 and a maximum of 2 for A and ω and 2π for ϕ and β. For simulating the elevations of a wave we set the parameters to their mid constraint points: $A = 1$, $\omega = 1$, $\phi = \pi$, and $\beta = \pi$, and then add Gaussian white noise.

We set the maximum number of iterations $\Lambda = 100$ for our PSO based estimator. The total number of particles in a swarm was set to $P = 40$. The self recognition and social components were set to $c_1 = 1$ and $c_2 = 2$ were we gave more emphasis on global exploration. We let PSO run for the full iterations Λ by setting $\tau = 0$ in order to study the results of a full run. One could set τ to a minimum error to reduce runtime (but this will come with potential expense of accuracy). For the stochastic inertia weight we set the minimum and maximum weights to $w_{min} = 0$ and $w_{max} = 1$ and we choose the local evolution constant to be $\gamma = 0.2$ to evolve the worst particles in the swarm closer to the solution of the global best particles. We tested our estimator against genetic algorithm, simulated annealing, and vanilla PSO without model exploitation. Results are obtained by doing a Monte Carlo analysis of 1000 runs for each method.

In Fig. 3 we present results from our estimator, Fig 2 shows its legend. Keeping everything constant, in Fig. 3.a we varied the variance of additive noise to simulated wave elevation sensor measurements. The plot includes the theoretical CRLB curves for the four parameters of a regular wave, and the MSE curves of the four parameters our estimator found. As one can see the MSE curves touch the CRLB curves which shows we've achieved the best possible estimates. Similar in Figs. 3.b we varied the number of sensors surrounding a WEC and in 3.c, we varied the sampling frequency (in Hz) of the sensors. Again we see in both plots that the MSE curves touch the CRLB which shows that for any experimental variation we always achieve the best estimates.

In Fig. 4 we show results from three different comparative methods which were obtained from the Matlab R2015b Global Optimization Toolbox. We used these methods to solve the wave parameter estimation problem which is a difficult problem due to the complexity of the objective function shown in Eq. (1). Methods were chosen to solve the estimation problem as-is, in other words these methods do not exploit the wave model structure. The three main methods we choose for comparison are adaptive PSO as described in section V, genetic algorithm, and simulated annealing. Fig 4, shows these methods applied when varying the variance of additive noise. As seen in all three plots, none of these methods are able to successfully estimate the parameters yielding MSE curves significantly above the lower bound curves. Experiments were also carried out with these three methods when varying the number of sensors and the sampling frequencies. All of which also had MSE curves significantly above the CRLB curves and hence not shown here. Additionally, we did not carry out experiments with simulated annealing or genetic algorithm of exploiting the domain since they generally have a longer runtime than PSO, and our estimator using PSO already produces adequate results.

VIII. Conclusion

We present a modified version of particle swarm optimization using local evolution and a stochastic inertia weight that is applied to estimate parameters of simulated regular wave realizations by strategically exploiting the sinusoidal model behavior of ocean waves. Estimation of wave parameters provides a way to do short term forecasting that can be used for control operations of a wave farm. Our approach works very well as estimates touch the Cramer-Rao lower bound giving the best possible results. Our method of exploiting the application domain for parameter estimation can be also be extended to other areas such as signal parameter estimation.

We are looking at several extensions for future work. Firstly we are interested to inspect the performance and runtime of using other state-of-the-art PSO version in place of ours. Secondly, we are planning to include in our optimization formulation regularization terms to prevent over-fitting the noise. And thirdly we will look into waves which may have irregular forms where they can be seen as a sum of regular waves, with each component having different parameter values. For short term forecasting of wave characteristics to be applied in wave energy farm control it may be necessary to then estimate an unknown number of A, ω, ϕ and β values. This problem is considered extremely hard and to tackle it we plan to significantly extend our PSO estimator with methods from estimation theory and machine learning.

Acknowledgments

This material is based upon work supported by the National Science Foundation under grant number CCF-1442858.

References

[1] T. Hong, P. Pinson, and S. Fan, "Global energy forecasting competition 2012," *International Journal of Forecasting*, vol. 30, no. 2, pp. 357–363, 2014.

[2] M. Lei, L. Shiyan, J. Chuanwen, L. Hongling, and Z. Yan, "A review on the forecasting of wind speed and generated power," *Renewable and Sustainable Energy Reviews*, vol. 13, no. 4, pp. 915–920, 2009.

[3] A. Costa, A. Crespo, J. Navarro, G. Lizcano, H. Madsen, and E. Feitosa, "A review on the young history of the wind power short-term prediction," *Renewable and Sustainable Energy Reviews*, vol. 12, no. 6, pp. 1725–1744, 2008.

[4] M. Richter, M. E. Magaña, O. Sawodny, and T. K. Brekken, "Nonlinear model predictive control of a point absorber wave energy converter," *Sustainable Energy, IEEE Transactions on*, vol. 4, no. 1, pp. 118–126, 2013.

[5] J. Cruz, *Ocean wave energy: current status and future prespectives*. Springer Science & Business Media, 2007.

[6] J. Falnes, *Ocean waves and oscillating systems: linear interactions including wave-energy extraction*. Cambridge university press, 2002.

[7] M. K. Ochi, *Ocean waves: the stochastic approach*. Cambridge University Press, 2005, vol. 6.

[8] R. H. Stewart, *Introduction to physical oceanography*. Texas A & M University, 2004.

[9] A. Malmberg, U. Holst, and J. Holst, "Forecasting near-surface ocean winds with kalman filter techniques," *Ocean Engineering*, vol. 32, no. 3, pp. 273–291, 2005.

[10] S. Mandal and N. Prabaharan, "Ocean wave forecasting using recurrent neural networks," *Ocean Engineering*, vol. 33, no. 10, pp. 1401–1410, 2006.

[11] S. Gaur and M. Deo, "Real-time wave forecasting using genetic programming," *Ocean Engineering*, vol. 35, no. 11, pp. 1166–1172, 2008.

[12] K. Hatalis, B. Alnajjab, S. Kishore, and A. Lamadrid, "Adaptive particle swarm optimization learning in a time delayed recurrent neural network for multi-step prediction," in *Foundations of Computational Intelligence (FOCI), 2014 IEEE Symposium on*. IEEE, 2014, pp. 84–91.

[13] K. Hasselmann, W. Sell, D. Ross, and P. Müller, "A parametric wave prediction model," *Journal of Physical Oceanography*, vol. 6, no. 2, pp. 200–228, 1976.

[14] H. L. Tolman *et al.*, "User manual and system documentation of wavewatch iii tm version 3.14," *Technical note, MMAB Contribution*, vol. 276, 2009.

[15] J.-R. Bidlot, D. J. Holmes, P. A. Wittmann, R. Lalbeharry, and H. S. Chen, "Intercomparison of the performance of operational ocean wave forecasting systems with buoy data," *Weather and Forecasting*, vol. 17, no. 2, pp. 287–310, 2002.

[16] G. Reikard, "Integrating wave energy into the power grid: Simulation and forecasting," *Ocean Engineering*, vol. 73, pp. 168–178, 2013.

[17] G. Reikard, P. Pinson, and J.-R. Bidlot, "Forecasting ocean wave energy: the ecmwf wave model and time series methods," *Ocean Engineering*, vol. 38, no. 10, pp. 1089–1099, 2011.

[18] B. Alnajjab and R. S. Blum, "Ocean wave prediction for noisy sensor measurements," in *Proc. Marine Energy Technology Symposium*, 2013.

[19] C. Radhakrishna Rao, "Information and accuracy attainable in the estimation of statistical parameters," *Bulletin of the Calcutta Mathematical Society*, vol. 37, no. 3, pp. 81–91, 1945.

[20] B. Alnajjab and R. S. Blum, "Optimal design of sensor networks for enhanced ocean wave energy conversion," in *Signals, Systems and Computers, 2013 Asilomar Conference on*. IEEE, 2013, pp. 384–388.

[21] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1. New York, NY, 1995, pp. 39–43.

[22] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1. IEEE, 2000, pp. 84–88.

[23] J. Bansal, P. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*. IEEE, 2011, pp. 633–640.

[24] M. Schwaab, E. C. Biscaia Jr, J. L. Monteiro, and J. C. Pinto, "Nonlinear parameter estimation through particle swarm optimization," *Chemical Engineering Science*, vol. 63, no. 6, pp. 1542–1552, 2008.

[25] A. Alireza, "Pso with adaptive mutation and inertia weight and its application in parameter estimation of dynamic systems," *Acta Automatica Sinica*, vol. 37, no. 5, pp. 541–549, 2011.

[26] E. Martinez-Ayala, V. Ayala-Ramirez, and R. E. Sanchez-Yanez, "Noisy signal parameter identification using particle swarm optimization," in *Electrical Communications and Computers (CONIELECOMP), 2011 21st International Conference on*. IEEE, 2011, pp. 142–146.

[27] H. Liu and J. Li, "A particle swarm optimization-based multiuser detection for receive-diversity-aided stbc systems," *Signal Processing Letters, IEEE*, vol. 15, pp. 29–32, 2008.