Prognostic Optimization of Phased Array Antenna for Self-Healing

David Allen[1]

[1]*HRL Laboratories, LLC, Malibu, CA, 90265, USA*
dlallen@hrl.com

ABSTRACT

Phased array antennas are widely used in many applications and consist of many antennas coupled together to enable digital beam-forming. As transmit/receive elements begin to degrade and eventually fail the antenna's beam will distort from the desired pattern. We propose a novel optimization algorithm which takes into account not only the current state-of-health of the system, but potential future states-of-health from prognostic observations. The approach can be run entirely off-line (before the start of a mission), so requires no additional computational resources or sensors be added to the system and does not require the system to be able to detect the degradation/failures during a mission. Our main objective is to trade some current optimization flexibility for improved system robustness under future failures.

1. INTRODUCTION

Phased array antennas (Hansen, 2009) are used in many domains such as radars, communications, satellites, and weather research and many deployed systems exist across airborne, ground, maritime and space domains. They are composed of many individual elements and the radiation pattern depends on each element's location, excitation magnitude and phase. As elements degrade and eventually fail this affects the ability of the array antenna to produce the desired radiation pattern.

Typically the location of the elements is fixed, however by adjusting the excitation magnitude and phase through digital beam-forming the radiation pattern (known as the beam) can be steered, made broader or narrower, regions of enhanced or nulled coverage can be created etc. without any mechanical rotation of the antenna. Array optimization or reconfiguration is the process of generating the parameters for the excitation magnitude and phase of each element to adapt the overall beam to the desired pattern. Most existing approaches are designed for offline use prior to start of a mission or task. They analyze the current state-of-health of the system, such as which elements are fully functional and which are failed, and then performs the optimization. In instances where failures can be detected during the mission these techniques can be rerun to compensate for failures.

The approach presented here assumes that we do not have a way of reliably detecting degradation or failures while in operation, however we may have the ability to detect which elements are at risk of failing in the near future (e.g. maybe they have already begun to degrade or are being heavily stressed by current usage). Additionally some new array materials such as GaN may provide prognostic observables prior to failures. We go beyond current techniques by not only optimizing over the current state-of-health, but also performing a preemptive optimization over potential future states-of-health.

This preemptive optimization is much more robust because it allows us to maintain mission specifications of our system even in the presence of undetected future failures which might occur during the mission. Overall this will help improve the system's affordability and survivability as repairs can be delayed or shifted to more convenient times, such as delaying them till access to external test/repair equipment is available. This graceful degradation or self-healing can lead to important performance improvements.

2. ARRAY OPTIMIZATION

An array's radiation pattern is a function of each element's location, excitation magnitude, and phase. An example beam pattern from a 32 element linear array is depicted in Figure 1. Many techniques exist for determining the excitation magnitude and phase parameters for each element to control the beam.

Some beam control may involve steering the beam or trying to optimize a cost criteria such as maximum side-lobe level (SLL), average SLL, or cumulative difference. Many techniques have been developed over the years to optimize the desired beam pattern. Some of the most common include genetic algorithms (Yeo & Lu, 1999), stochastic optimization (such as Particle Swarm Optimization - PSO)

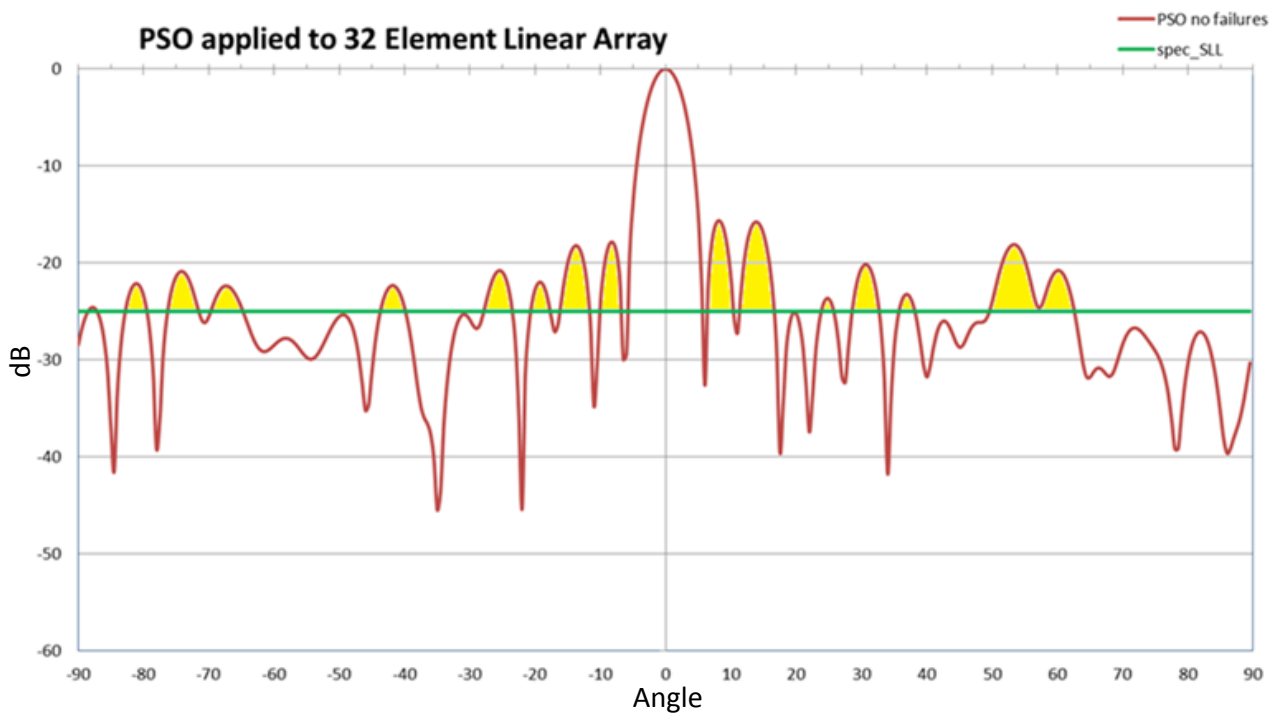

Figure 1. Radiation pattern generated from a 32 element linear array.

(Yeo & Lu, 2009)(Boeringer & Werner, 2004)(Khodier & Al-Aqeel, 2009), and hybrid approaches (Yeo & Lu, 2005).

Some approaches have also been developed to handle re-optimization after element failures (Joler, 2012)(Keizer, 2007). These are mostly performed off-line prior to a mission. There have been some attempts at detecting failures while the array is in use and doing very efficient heuristic compensation (Levitas et al., 1999).

The radiation pattern can be generated from the array factor (AF) given by (Boeringer & Werner, 2004):

$$AF(\Theta) = \sum_{n=1}^{N} A_n e^{j2\Pi n(d/\lambda)\sin(\Theta)} \quad (1)$$

where N is the number of radiating elements, A_n are the complex element weights for excitation magnitude and phase, and $d/\lambda = ½$ is the spacing between elements normalized by the wavelength.

Particle Swarm Optimization

Particle swarm optimization (PSO) is a generic optimization approach to iteratively improve the current best solution with regard to a given metric and has been used extensively for optimizing phased array antennas. The basic concept is that there is a swarm of particles where each is a possible solution (i.e. a setting of all elements' excitation magnitude and phase parameters). These particles move through the solution space based on their own local observations and also the best known position of the swarm in the overall search-space. This allows it to be guided to regions of known good quality while still allowing particles to explore unknown regions in search of better solutions. In practice, as the algorithm progresses the particles will move toward near-optimal solutions.

Algorithm 1 presents the pseudo code for PSO. After initialization it iteratively updates each particle's velocity and position; then it computes a cost function to determine if the position is better than previously observed positions. PSO is therefore general enough that it can optimize over various different cost functions.

The results of PSO are not guaranteed to be optimal, however in practice the optimization converges to near optimal results fairly quickly and in many instances have been shown to outperform other approaches such as genetic algorithms (Yeo & Lu, 2009)(Boeringer & Werner, 2004).

Some typical cost functions used for phased array optimization include:

Maximum side-lobe level (or Peak side-lobe level): the largest side peak of the beam pattern relative to the main beam

Average side-lobe level: the average of the side-lobe peaks relative to the main beam

Cumulative Difference: the area under the beam pattern but above a specified threshold (ignoring the main beam) (see portion shaded yellow in Figure 1).

Algorithm 1. Particle Swarm Optimization (PSO) Pseudocode

1. Initialization
   a. For all particles
      i. Set position uniformly distributed $x_i = U(b_{low}, b_{up})$;
         where b defines the search space and low is the lower bound and up is the upper bound
      ii. Set velocity $v_i=U(-|b_{up}-b_{low}|, |b_{up}-b_{low}|)$
      iii. Initialize the particle's best known position (p_i)
   b. Initialize swarm's best known position (g)
2. Repeat until termination criteria met
   a. For each particle i
      i. Update the velocity (v_i,d) for each dimension d based on PSO update function
      ii. Update the position $x_i = x_i+v_i$
      iii. Compute cost function: $f(x_i)$
      iv. If($f(x_i)$ better than $f(p_i)$)
         1. $p_i = x_i$; Update the particle's best known position
      v. If($f(x_i)$ better than f(g))
         1. $g = x_i$; Update the swarm's best known position
3. Return g, the best solution found

3. PREEMPTIVE OPTIMIZATION ALGORITHM

As elements of the phased array antenna fail the radiation pattern will get distorted. For example the main beam may broaden out or the side-lobes may increase above the desired threshold. If you can detect the failure while the array is in the field then you can re-optimize the pattern to compensate for the distortion or degradation (Keizer, 2007). In many systems the engineering cost to add additional sensors to reliably detect the failures is prohibitive and therefore failures cannot be detected while system is in use and external test equipment unavailable. However in some instances we may be able to detect potential future failures, such as elements that have not completely failed but have partially degraded, or elements which have been heavily stressed in the past, or those where we have prognostic observations predicting failure onset.

In this work we propose a new optimization approach which not only leverages the system's current state-of-health, but its potential future states. Current algorithms monitor the current state-of-health and assume it is fixed, however in the real-world those elements will begin to degrade and eventually fail. If left uncorrected, these can significantly affect the performance of the array. Our novel optimization approach works by adapting the cost function used by the PSO algorithm.

For simplicity we will assume either an element is failed or not (the algorithm can be extended to handle the case of degradation).

Let F be a list of currently failed elements, (e.g. $F = \{3,4,6,7\}$).

Let P be a list of potential future failures, (e.g. $P = \{5,20,30\}$).

The standard approach to optimization would compute PSO(F). It takes the current state-of-health as input and a previously defined cost metric. The optimization generates a set of element parameters optimizing the beam with respect to the cost metric. Failures of elements in F can be modeled by setting the excitation magnitude of those elements to 0.

Our approach, PSO_Robust(F, P), takes as input both the current state-of-health and a list of potential future element failures. In Algorithm 1, on line (2.a.iii) a cost function $f(x_i)$ is computed. The input to this function is a current instantiation of all the elements' parameters (hence with it you can compute the radiation pattern such as in Figure 1 and compute the cost functions previously described).

We will replace the cost function $f(x_i)$ with the following:

$$f\_robust(x_i, P) = \sqrt[|P|+1]{f(x_i) * \prod_{p \in P} f(x_i, x_p = 0)} \quad (2)$$

where |P| is the cardinality of P.

What the above cost function does is compute the cost under the current-state-of-health, $f(x_i)$, and under each potential future state $f(x_i, x_p=0)$ under the assumption of single future failure. This function then combines these results using the geometric mean. Other approaches could be used to combine the results (e.g. arithmetic mean [average] of the costs, weighted combination of current and average of potential states, etc.). We chose to use geometric mean because it more heavily weights bad instances than the others. For example under the above scenario where our potential future failures are 5, 20, and 30, if all degrade evenly it would not matter which cost function we chose, but let us assume a failure at 20 or a failure at 30 would degrade performance by a small amount but a failure at 5 would severely impact performance since we already have 3,4,6&7 failed and losing 5 creates a large clustered failure (i.e. no radiation from five consecutive array elements). If x_n handles the future case of a failure at 5 better than x_m then we would like the cost function to measure that, and in general we are more worried about worst case single failures (as they may potentially happen) more than average case results. This ensures that if that failure happens we will maintain our mission specifications under that specific condition rather than the average of all future states.

Table 1. Results showing the max peak SLL and cumulative difference cost function for standard PSO (top) and our robust extension (bottom) under the two cases of no failures and element 5 failing.

PSO	**Max Peak SLL**	**Cumulative Difference**
No Failures	-16.47	49.63
Failures: #5	-14.16	69.53

Robust {5}	**Max Peak SLL**	**Cumulative Difference**
No Failures	-16.46	59.92
Failures: #5	-16.95	40.86

(SLL threshold = -20dB, swarm=3000, epochs=100)

This approach does incur a penalty for this improved robustness. The generated radiation pattern under the current state-of-health will not be quite as good with respect to the cost metric, however if any of the potential failures occurs it will maintain a more desirable beam pattern. Additionally the metrics can be analyzed a priori to determine if failures would result in performance below mission specifications. This improved robustness, at the expense of a reduced performance, can be very desirable in many application domains such as where online detection of failures is infeasible either technically or due to cost of additional sensors.

4. EXPERIMENTAL RESULTS

We have implemented the above algorithm and performed experiments on a linear array, however the approach is completely general and could be used for other array configurations such as two dimensional arrays. The results we present are using the cumulative difference cost function, but we have experimented with other various cost functions with similar results.

Table 1 shows results where the current health of the system is fully functional and the only potential future failure is element 5. Running PSO results in a beam pattern where the cumulative difference is 49.63, whereas the robust version's difference is 59.92 (lower is better). These correspond with the Max Peak SLL shown on the left, where the difference is only .01 dB (-16.46 vs. -16.47). However if element 5 does fail the cost metric for the PSO optimized beam shoots up to 69.53 compared with only 40.86 for the robust version. Similarly this results in a peak side-lobe level which is 2.5 dB better.

Figure **2** depicts the beam patterns of both the standard PSO and our robust extension under the case of a fully health array and Figure **3** depicts them in the case of an undetected or uncompensated failure at element 5. As can be seen in Figure **2** both algorithms have relatively similar main beams and peak side-lobe levels, however under future failure of element 5 (Figure **3**) the side-lobe closest to the main lobe jumps dramatically under standard PSO, while the robust PSO can still maintain similar performance.

For a slightly more complex case, we look at Table 2, where the current state-of-health has elements 3, 4, 6, & 7 all failed and 5, 20, & 30 are potential future failures. In this case our initial penalty for incorporating robustness is only 5.39 (34.57 vs. 29.19), but under any failure the benefit is fairly substantial (119.72, 118.11, and 63.19). Figures 4-7 show the patterns for the current state-of-health of the array as well as for each of the 3 potential states of the array. Similar to the previous example under the different future failures the robust algorithm does in face maintain not only a better cumulative cost function (which is what it optimized over) but the peak side-lobes also are maintained. If we directly optimize peak SLL we might see even further improvements, however our goal was to maintain the entire pattern, hence the choice of the cumulative difference metric.

Table 2. Results showing the cumulative cost function from the array pattern computed under the failed elements 3, 4, 6, & 7. The first row shows the penalty paid by incorporating the robustness, but in the instances when either 5, 20, or 30 failed there is a substantial benefit.

Failures	**PSO**	**Robust[3,4,6,7]+{5,20,30}**	**Difference**
3, 4, 6, 7	29.19	34.57	5.39
3, 4, 6, 7, 5	262.04	142.32	-119.72
3, 4, 6, 7, 20	352.31	234.20	-118.11
3, 4, 6, 7, 30	165.95	102.76	-63.19

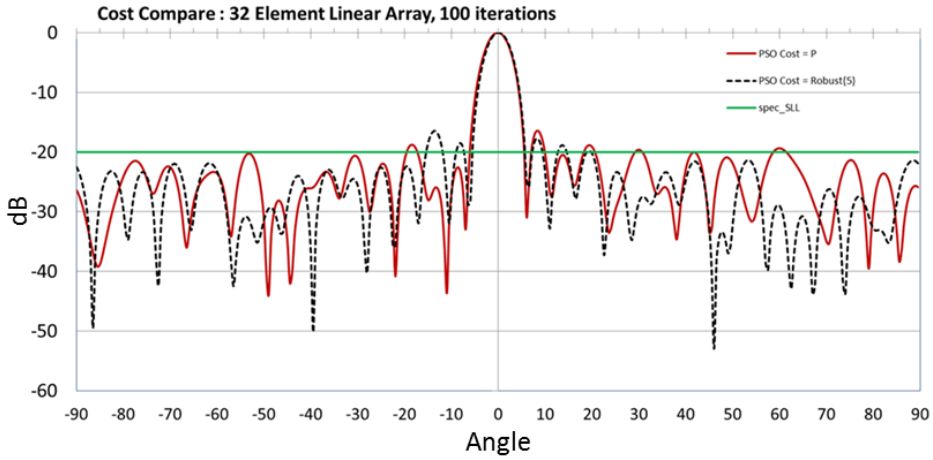

Figure 2. Beam pattern for standard PSO (red) and robust PSO (black-dashed), where there were originally no failed elements and the only potential failure used by the robust version was element 5.

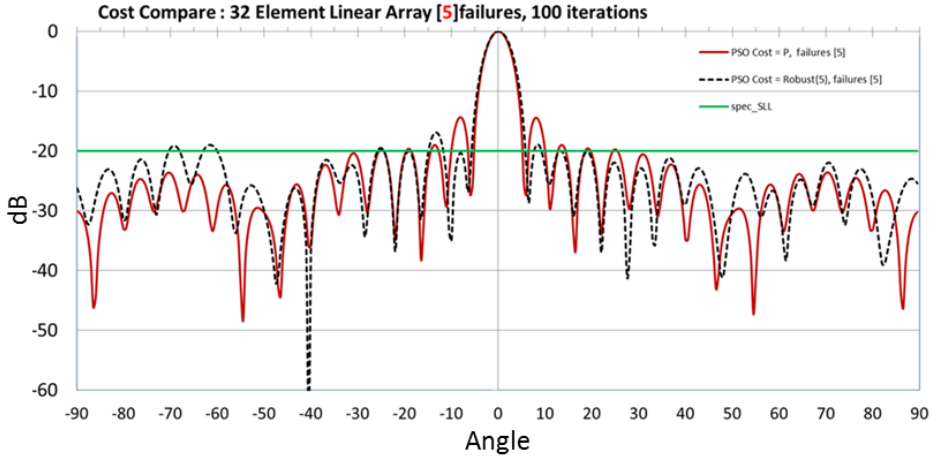

Figure 3. Beam pattern for standard PSO (red) and robust PSO (black-dashed), where they were optimized with no failed elements, but then element 5 did fail. Our robust extension to PSO was able to maintain lower peak side-lobe levels than standard PSO.

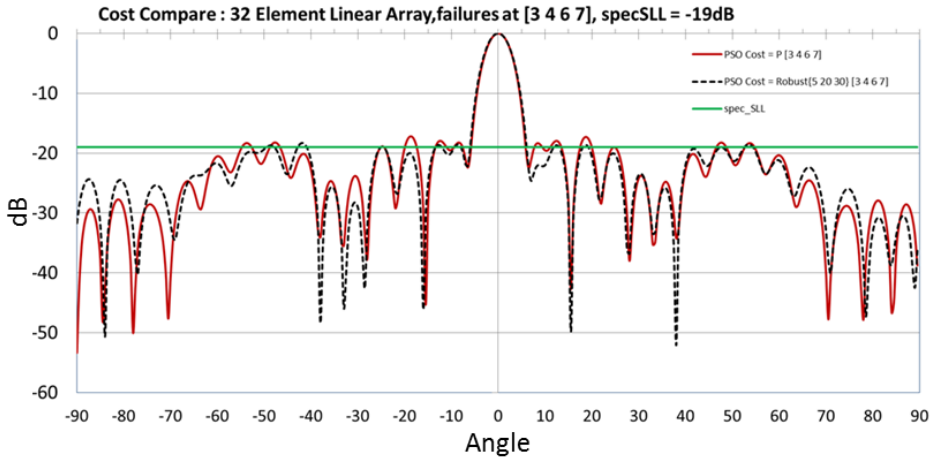

Figure 4. Beam pattern for standard PSO (red) and robust PSO (black-dashed) where elements 3, 4, 6, and 7 were originally failed and elements 5, 20, and 30 were potential future failures.

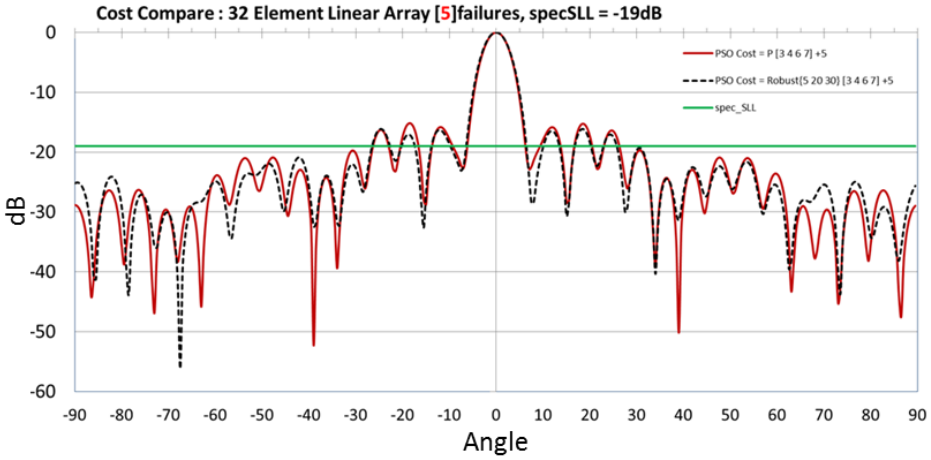

Figure 5. Beam pattern for standard PSO (red) and robust PSO (black-dashed) where elements 3, 4, 6, and 7 were originally failed prior to the optimization and element 5 later failed.

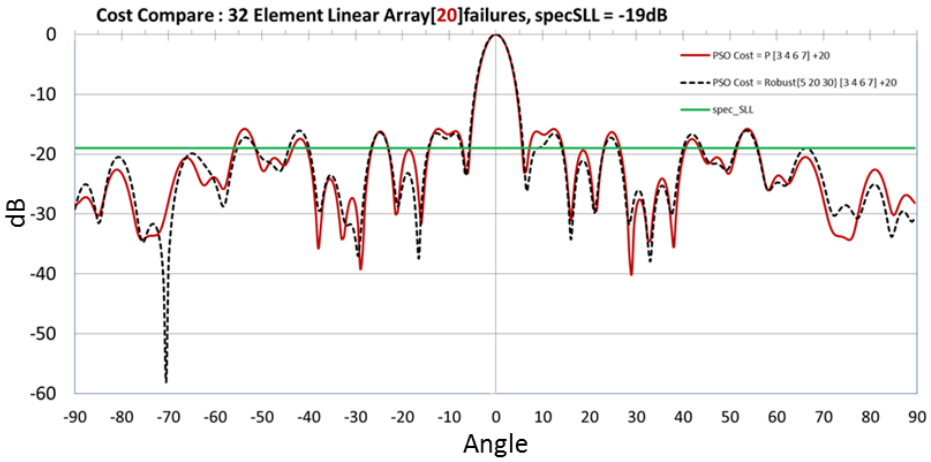

Figure 6. Beam pattern for standard PSO (red) and robust PSO (black-dashed) where elements 3, 4, 6, and 7 were originally failed prior to the optimization and element 20 later failed.

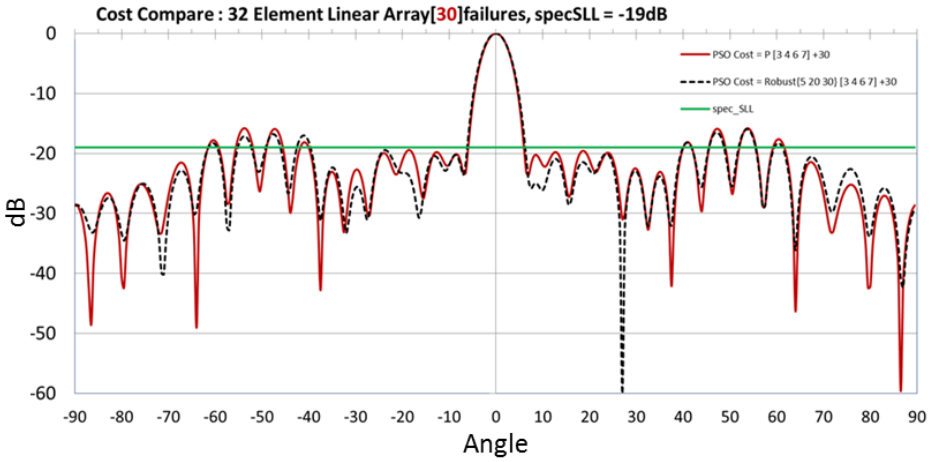

Figure 7. Beam pattern for standard PSO (red) and robust PSO (black-dashed) where elements 3, 4, 6, and 7 were originally failed prior to the optimization and element 30 later failed.

5. CONCLUSIONS

In this work we propose a novel prognostic approach to do preemptive optimization of phased array antennas. When determining element parameters for excitation magnitude and phase during digital beam-forming we not only optimize over the current state-of-health but consider potential future states-of-health. This allows the algorithm to trade some current optimization flexibility for improved system robustness under future failures which might occur during a mission. This improves the overall system's affordability and survivability as it is more robust to failures and repairs can be performed at more optimal times. This technique does assume that potential future failures can be determined, however there is evidence that in many systems this is true. Additionally this approach does not require additional sensors or engineering to reliably detect failures during a mission and does not require systems resources while online, as it is performed prior to the start of a mission but then has the most effect when failures do occur. It also allows a user to determine whether the system will be able to maintain minimum mission specifications even under potential failures a priori, allowing them to make a decision whether to go ahead with the mission.

ACKNOWLEDGEMENT

The author would like to thank Boeing for their support of this project and the technical discussions and implementation work of Gavin Holland, Edward Sabatka, and Marian Kis.

REFERENCES

Boeringer, D.W., & Werner, D.H., (2004). Particle Swarm Optimization Versus Genetic Algorithms for Phased Array Synthesis. *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3.

Hansen, R.C., (2009). *Phased Array Antennas*, Second Edition, John Wiley & Sons, Inc.

Joler, M., (2012). Self-recoverable antenna arrays. *IET Microwaves, Antennas & Propagation*.

Keizer, W., (2007). Element Failure Correction for a Large Monopulse Phased Array Antenna with Active Amplitude Weighting. *IEEE Transactions on Antennas and Propagation*, Vol. 55, No. 8.

Khodier, M., & Al-Aqeel, M., (2009). Linear and Circular Array Optimization: A Study Using Particle Swarm Intelligence. *Progress in Electromagnetics Research B*, Vol. 15.

Levitas, M., Horton, D.A., & Cheston, T.C., (1999). Practical Failure Compensation in Active Phased Arrays. *IEEE Transactions on Antennas and Propagation*, Vol. 47, No. 3.

Yeo, B-K., & Lu, Y., (1999). Array Failure Correction with a Genetic Algorithm. *IEEE Transactions on Antennas and Propagation*, Vol. 47, No. 5.

Yeo, B-K., & Lu, Y., (2005). Adaptive array digital beamforming using complex-coded particle swarm optimization-genetic algorithm. *Microwave Conference*.

Yeo, B-K., & Lu, Y., (2009). Fast array failure correction using improved particle swarm optimization. *Microwave Conference*.

BIOGRAPHIES

David Allen received his M.S. and Ph.D. degrees in Computer Science from the University of California, Los Angeles (UCLA) in 2001 and 2005 respectively. He joined HRL Laboratories, LLC, in 2006 and currently holds the position of Research Staff Scientist. His research interests include artificial intelligence, probabilistic reasoning, data science, decision making under uncertainty, complex systems analysis, and network science; he has applied them to many domains including integrated system health management, cybersecurity, social network analysis, and behavior modeling. In 2010 he received HRL's Distinguished Inventor Award and in 2011 received a GM Contribution Award.