Naval Research Laboratory

Washington, DC 20375-5320

NRL/MR/5341--16-9680

A Parameterized Pattern-Error Objective for Large-Scale Phase Only Array Pattern Design

DAN P. SCHOLNIK

Surveillance Technology Branch
Radar Division

March 21, 2016

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
21-03-2016	Memorandum Report	1 October 2012 – 31 March 2015

4. TITLE AND SUBTITLE

A Parameterized Pattern-Error Objective for Large-Scale Phase-Only Array Pattern Design

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Dan P. Scholnik

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
4895

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/5341--16-9680

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5320

10. SPONSOR / MONITOR'S ACRONYM(S)

NRL

11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Modern phased-array radar systems increasingly require customized transmit patterns, implemented using only phase shifts, in order to illuminate or suppress interference from specific regions. Multiple custom patterns may be required at every beam position, requiring either precomputation of a large library of phase weights or real-time computation. Presented here is a weighted pattern-error formulation that offers efficient FFT-based calculation of both the objective and its gradient vector and is well suited for solving with efficient quasi-Newton methods. Although deliberately simple, the objective has several heuristic parameters for flexibility and can be used to simultaneously customize the mainbeam shape and selectively suppress the sidelobes. Examples demonstrate the feasibility of 1000+ element 2D array designs, as well as a characterization of point-nulls that commonly occur in the mainbeam of suboptimal local minima.

15. SUBJECT TERMS

Phase only array pattern
Transmit array pattern
Phase-only optimization
Array synthesis
Array pattern optimization

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dan P. Scholnik
Unclassified Unlimited	Unclassified Unlimited	Unclassified Unlimited	Unclassified Unlimited	24	**19b. TELEPHONE NUMBER** *(include area code)* (202) 404-1943

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

CONTENTS

EXECUTIVE SUMMARY

Modern phased-array radar systems increasingly require customized transmit patterns, implemented using only phase shifts, in order to illuminate or suppress interference from specific regions. Multiple custom patterns may be required at every beam position, requiring either precomputation of a large library of phase weights or real-time computation. Presented here is a weighted pattern-error formulation that offers efficient FFT-based calculation of both the objective and its gradient vector and is well suited for solving with efficient quasi-Newton methods. Although deliberately simple, the objective has several heuristic parameters for flexibility and can be used to simultaneously customize the mainbeam shape and selectively suppress the sidelobes. Examples demonstrate the feasibility of 1000+ element 2D array designs, as well as a characterization of point-nulls that commonly occur in the mainbeam of suboptimal local minima.

A PARAMETERIZED PATTERN-ERROR OBJECTIVE FOR LARGE-SCALE PHASE-ONLY ARRAY PATTERN DESIGN

1. Introduction

Traditionally, radar transmitters are run in saturation to maximize sensitivity and power efficiency. Although research into efficient amplitude-modulated transmitters is accelerating [1–3], in the near term high-power radar transmitters will continue to offer only phase control. This considerably complicates the design of custom array patterns for transmit arrays, as all but the most trivial phase-only optimizations are nonconvex and thus may have large numbers of local optima. In most fielded radar systems only the uniformly illuminated transmit beam is used, which provides maximum sensitivity but a narrow mainlobe and relatively poor sidelobes. In modern multifunction/multibeam arrays, however, it is often desirable to trade sensitivity for a broader mainbeam, or to selectively null directions or sectors of interest, or both. This may require patterns customized to each beam position, for example to keep a deep null around the horizon as the main beam is scanned. This in turn requires a highly efficient approach that can be used to precompute large libraries of coefficients, or to compute on the fly.

The existing approaches to phase-only array design can be roughly divided into heuristic, nonlocal optimization, and local optimization methods. The first two of these are primarily of interest here in that they can be used to augment or initialize local minimization. Many heuristic techniques are descended from a well-known approach to nonlinear FM waveform synthesis [4] and only return reliably good results for impractically large arrays [5]. Examples of 1D heuristic design approaches can be found in [6–11]. Reference [12], which proposes separate beams on interlaced subarrays, represents a 2D heuristic method. Nonlocal optimization methods (often referred to as "global" methods) such as simulated annealing and genetic algorithms have also been used in 1D and 2D [13–18], although they generally do not guarantee a global solution in a reasonable time and indeed can be impractically slow when applied directly to large problems.

The drawbacks of heuristic and nonlocal methods suggest that local optimization should be a part of practical large-scale approaches. Local optimization approaches are presented in [19–24] for 1D. The reflectarray and conformal literature is particularly rich with 2D/3D approaches [25–29], many based on the alternating-projection method of [30, 31]. Many of these approaches combine a core local minimization step with heuristic or nonlocal methods in order to deal with globally suboptimal local minima. Further discussion on this topic is deferred until 3.2. An approach that seems to be globally optimal is [32], but it is largely restricted to custom sidelobe nulling rather than mainlobe shaping. (An extension is presented in [33], which introduces mainlobe shaping but also local minima.)

Of the above approaches, local optimization combined with some non-local or heuristic methods to avoid or discard "bad" local minima seems to be the preferred approach. To that end, this work presents a pattern-error objective function that mirrors those used in complex-weight (non-phase-only) pattern design, is flexible, can be efficiently computed along with its gradient, and is suitable for solution with modern tools

Manuscript approved January 14, 2016.

that can handle designs with thousands of elements. Problems of this class often have local minima with characteristic point-nulls in the mainbeam, which some simple heuristics are used to identify and avoid.

2. Weighted Pattern Error

2.1 Notation

Throughout the sequel, scalar-valued variables (x,ϕ) will be represented with a medium weight italic font, while vector and matrix values will be represented in bold upright ($\mathbf{x}$) or Greek ($\boldsymbol{\phi}$) characters. The nth element of a vector $\mathbf{x}$ is represented either explicitly as $[\mathbf{x}]_n$ or implicitly as x_n.

2.2 Array Factor

We assume in the sequel that we have identical isotropic elements in a planar array, so that the array pattern reduces to the 2D *array factor*

$$A(\mathbf{u}) = \sum_{n=0}^{N-1} \mathrm{a}(\mathbf{x}_n) e^{j\frac{2\pi}{\lambda}\mathbf{u}^T\mathbf{x}_n}, \tag{1}$$

where $\mathrm{a}(\mathbf{x}_n)$ is the complex array weight applied to the element at array-plane location $\mathbf{x}_n$. Here, $\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix}$ represents two-dimensional spatial frequency normalized so that the unit circle $\|\mathbf{u}\| \leq 1$ corresponds to propagating radiation (so-called "visible" space). The array factor has the form of a discrete-space Fourier transform but with an inverse-transform exponent to describe plane-wave propagation.

The identical-element assumption will not always apply, and it is straightforward to incorporate unique element patterns into the array pattern. When it does apply, however, the benefits are substantial: the array factor can be computed using FFT techniques. This enables efficient computation of key quantities used in the optimization of the array pattern.

2.3 Pattern Error Objective

A very common approach to the design of FIR filters, phased array weights, and related structures is to minimize some weighted error between the response being designed and some ideal desired response. When coefficient amplitudes are not fixed, this often results in a convex optimization [34, 35]. A similar approach can be used for phase-only responses, but will not be convex. Consider first the weighted L_p pattern error metric

$$f(\mathbf{a}) = \left(\int W^{pq}(\mathbf{u}) \left| |A(\mathbf{u})|^q - D^q(\mathbf{u}) \right|^p d\mathbf{u} \right)^{\frac{1}{p}}, \; \begin{array}{l} p \geq 1 \\ q = \{1,2\} \end{array} \tag{2}$$

where $\mathbf{a}$ is the vector of array weights with $[\mathbf{a}]_n = \mathrm{a}(\mathbf{x}_n)$, W is a nonnegative weighting function, and D is the desired magnitude pattern. The support of W determines the region of integration, and its value defines how much each direction adds to the overall error. It can be zero for don't-care regions. The parameters q and p permit some additional control over the nature of the resulting approximation, with q choosing whether

to approximate either the magnitude or power pattern[1] and real-valued p controlling the relative importance of large and small errors. In particular, larger p emphasizes peak approximation errors (thus providing more of an equiripple error) while smaller p emphasizes the average errors. (The least-squares special case of $p = 2$ is already widely used for phase-only synthesis, for example [26, 36, 39].) One key difference from traditional convex weighted error metrics is that only the magnitude of A is considered; the phase response will be determined by the optimization. In complex-weight design the absolute value is usually avoided by imposing conjugate symmetry on the weights, but conjugate symmetry is precluded in all but the most trivial many phase-only designs. (Details are in Appendix A.)

Although (2) is written in terms of the array factor only, it is straightforward to incorporate a common nonisotropic element pattern A_{el} by defining $W(\mathbf{u}) = W'(\mathbf{u})|A_{\mathrm{el}}(\mathbf{u})|$ and $D(\mathbf{u}) = D'(\mathbf{u})|A_{\mathrm{el}}(\mathbf{u})|^{-1}$. Thus the simplifying isotropic element assumption does not lead to any significant loss of generality.

2.4 Unconstrained vs. Constrained vs. Feasibility Formulations

The focus here is on unconstrained minimization of (2) for absolute efficiency. This is important for phased-array applications as many thousands of different beams might need to be precomputed, or custom beams may be required to be computed on the fly. Another advantage of unconstrained minimization is that it (by definition) guarantees a feasible solution. A third advantage is that it guarantees (locally) Pareto-optimal solutions; that is, no part of the pattern can be improved without degrading some other part. This ensures that all available degrees of freedom are used. The primary drawback of using (2) for unconstrained formulation is that it relies on the heuristic choice of a weighting function W in order to control the relative errors in different parts of of the pattern. Thus without some trial-and-error or additional information one cannot guarantee specific error bounds.

A common alternative is the family of projection-based *feasibility methods* based on [31] which have no objective function as such but seek to directly meet a set of constraints. These methods have their own drawbacks. First, the constraints may be infeasible. Alternately, the constraints may be too loose, in which case the first feasible iterate found is returned even though additional improvement is possible. In general feasibility methods do not produce Pareto-optimal solutions. These methods are often employed for reflectarrays, where one or a small number of fixed beams are to be designed and computational speed is less of a factor.

Finally, there exist several approaches to extend the unconstrained optimization presented here to full constrained optimization [40]. This can be done by adapting the weighting function W to achieve the desired levels, which is effectively a penalty method. The *augmented Lagrangian* method is a related alternative that can improve conditioning by estimating the Lagrange multipliers (dual variables). The drawback of these methods is primarily increased computational complexity.

2.5 Derivation of the Gradient

We now proceed to derive the gradient of the objective f, which we need for most optimization solvers. The objective is a real-valued function of the complex vector $\mathbf{a}$, which we decompose as magnitude/phase

[1] It has been claimed that $q = 2$ reduces the problem of undesirable local minima, based on results from related analysis problems [36–38]. However, no significant difference in local-minima performance between $q = 1$ and $q = 2$ was observed here. This may be attributable to the differences between the analysis and synthesis problems (including a lack of a phase-only constraint in the former) or to the nature of the local minima most often encountered in this work.

$\boldsymbol{\rho} \circ e^{j\boldsymbol{\phi}}$ where $\circ$ is the Hadamard (pointwise) product and exponentiation by a vector is taken pointwise. The gradient with respect to both magnitude and phase can be written as the $2N \times 1$ vector

$$\nabla f(\boldsymbol{\rho}, \boldsymbol{\phi}) \triangleq \begin{pmatrix} \nabla_{\boldsymbol{\rho}} \\ \nabla_{\boldsymbol{\phi}} \end{pmatrix} f(\boldsymbol{\rho}, \boldsymbol{\phi})$$

and straightforward application of the chain rule to (2) yields

$$\nabla f \;=\; f^{1-p} \int W^{pq}(\mathbf{u}) \Big| |A(\mathbf{u})|^q \;-\; D^q(\mathbf{u}) \Big|^{p-1} \operatorname{sgn}\big(|A(\mathbf{u})|^q \;-\; D^q(\mathbf{u})\big) \frac{q}{2} |A(\mathbf{u})|^{q-2} \quad \nabla |A(\mathbf{u})|^2 \, d\mathbf{u}. \tag{3}$$

It remains to compute the gradient of $|A(\mathbf{u})|^2$:

$$\begin{aligned} \nabla |A(\mathbf{u})|^2 &= \nabla \Big(\operatorname{Re}\{A(\mathbf{u})\}^2 + \operatorname{Im}\{A(\mathbf{u})\}^2 \Big) \\ &= 2 \operatorname{Re}\{A(\mathbf{u})\} \nabla \operatorname{Re}\{A(\mathbf{u})\} + 2 \operatorname{Im}\{A(\mathbf{u})\} \nabla \operatorname{Im}\{A(\mathbf{u})\}. \end{aligned} \tag{4}$$

Since A is a holomorphic (analytic) function, it obeys the polar form of the Cauchy-Riemann equations:

$$\begin{aligned} \boldsymbol{\rho} \circ \nabla_{\boldsymbol{\rho}} \operatorname{Re}\{A(\mathbf{u})\} &= \quad \nabla_{\boldsymbol{\phi}} \operatorname{Im}\{A(\mathbf{u})\} \\ \boldsymbol{\rho} \circ \nabla_{\boldsymbol{\rho}} \operatorname{Im}\{A(\mathbf{u})\} &= -\nabla_{\boldsymbol{\phi}} \operatorname{Re}\{A(\mathbf{u})\}. \end{aligned}$$

Thus the gradient is completely defined by the real and imaginary parts of

$$\nabla_{\boldsymbol{\rho}} A(\mathbf{u}) = e^{j\boldsymbol{\phi}} \circ e^{j\frac{2\pi}{\lambda}\mathbf{X}^T\mathbf{u}}$$

where $\mathbf{X}$ is the matrix whose columns contain the in-plane element locations $\{\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}\}$. Substituting back into (4) and applying the identities $\operatorname{Re}\{xy^*\} \equiv \operatorname{Re}\{x\}\operatorname{Re}\{y\} + \operatorname{Im}\{x\}\operatorname{Im}\{y\}$ and $\operatorname{Im}\{xy^*\} \equiv \operatorname{Im}\{x\}\operatorname{Re}\{y\} - \operatorname{Re}\{x\}\operatorname{Im}\{y\}$ we find

$$\begin{pmatrix} \nabla_{\boldsymbol{\rho}} \\ \nabla_{\boldsymbol{\phi}} \end{pmatrix} |A(\mathbf{u})|^2 = 2 \begin{pmatrix} \operatorname{Re}\Big\{ A(\mathbf{u}) e^{-j\boldsymbol{\phi}} \circ e^{-j\frac{2\pi}{\lambda}\mathbf{X}^T\mathbf{u}} \Big\} \\ \operatorname{Im}\Big\{ A(\mathbf{u}) \boldsymbol{\rho} \circ e^{-j\boldsymbol{\phi}} \circ e^{-j\frac{2\pi}{\lambda}\mathbf{X}^T\mathbf{u}} \Big\} \end{pmatrix}.$$

If we now fix the weight magnitudes $\boldsymbol{\rho}$ we are left with the $N \times 1$ gradient $\nabla_{\boldsymbol{\phi}} |A(\mathbf{u})|^2$. Substituting back into (3), we find the nth gradient component as

$$\begin{aligned} \frac{\partial f}{\partial \phi_n} &= \operatorname{Im}\Big\{ \int f^{1-p} W^{pq}(\mathbf{u}) \Big| |A(\mathbf{u})|^q - D^q(\mathbf{u}) \Big|^{p-1} \times \qquad (5a) \\ &\qquad \operatorname{sgn}\big(|A(\mathbf{u})|^q - D^q(\mathbf{u})\big) \frac{q A(\mathbf{u})}{|A(\mathbf{u})|^{2-q}} e^{-j\frac{2\pi}{\lambda}\mathbf{u}^T\mathbf{x}_n} d\mathbf{u} \; \rho_n e^{-j\phi_n} \Big\} \\ &= \operatorname{Im}\Big\{ \int B(\mathbf{u}) \, e^{-j\frac{2\pi}{\lambda}\mathbf{u}^T\mathbf{x}_n} d\mathbf{u} \; \rho_n e^{-j\phi_n} \Big\} \qquad (5b) \end{aligned}$$

where in (5b) we combine all the weighting factors into a single function B (which has the same support as W) and see that the integral has the form of a Fourier transform. Although this expression is a little messy, it is straightforward to compute. In practice the integrals in the objective and gradient will be approximated by a sum over lattice points. As shown in the next section, choosing the approximating lattice properly in relation to the element lattice allows the forward and inverse transforms to be efficiently computed using an FFT/IFFT pair.

A couple of concluding notes: Although the restriction $\boldsymbol{\rho} = \mathbf{1}$ is most common, representing an array with fixed uniform amplitude illumination, phase-only optimization can also find application to arrays with fixed but nonuniform tapers. Such fixed tapers might be designed over the ensemble of desired beams, as in [41], or imposed by a legacy system. Also, (2) can be somewhat trivially extended to a sum of weighted L_p norms with different values of p. This would allow designs with different characteristics in different regions: an equiripple mainbeam and a L_2 sidelobe region, for example. This would still require only a single FFT/IFFT pair to compute.

2.6 FFT Computation of Objective and Gradient

Because the objective and gradient will be repeatedly evaluated inside of an optimization routine, we want to find an efficient way to compute them. The majority of the computation lies in the sum of (1) and the integral (to be approximated with a sum) of (5b). The identical-element assumption ensures that these are both Fourier transforms, and thus fast algorithms are available. If the element locations are restricted to a lattice (which they are in most large linear and planar arrays), then the computation can be made using any of a number of multidimensional FFT/IFFT algorithms [42]. For arbitrary element locations nonuniform FFT algorithms exist [43–45] that have the same asymptotic complexity as the conventional FFT, although the overhead and scale factor are greater. These find application for conformal [26] and otherwise irregular [46] arrays, where a uniform lattice is not available.

We focus attention here on planar arrays with a uniform element lattice; the restriction to 1D or extension to 3D is mathematically straightforward although shadowing effects will generally violate the ideal element pattern assumption in 3D. The approach outlined below maps samples on an arbitrary lattice such that an ordinary rectangular FFT can be used, since such routines tend to be widely available and are the mostly highly optimized for computation. Similar approaches can be found in [42].

Let 2×2 matrix $\boldsymbol{\Lambda}$ with linearly independent columns be a lattice basis for the wavelength-normalized in-plane array element positions, such that $\{\mathbf{x}_n, \ldots, \mathbf{x}_{N-1}\} \subset \lambda\boldsymbol{\Lambda}\mathbb{Z}^2$ where $\mathbb{Z}^2$ is the set of all integer 2-vectors. This lets us re-enumerate the element positions in terms of index $\mathbf{n} \in \mathbb{Z}^2$ as $\mathbf{x}_\mathbf{n} = \lambda\boldsymbol{\Lambda}\mathbf{n}$. The columns of the dual lattice basis $\boldsymbol{\Lambda}^{-T}$ define two adjacent sides of a parallelogram that encloses one period of the array factor. We wish to sample the array factor on a uniform grid within this period of sufficient density to approximate the integral of (2). We do this by defining a superlattice of the dual lattice with basis $\boldsymbol{\Lambda}^{-T}\mathbf{R}^{-1}$, where $\mathbf{R} = \mathrm{diag}(R_1, R_2)$ is a diagonal integer resampling matrix. This then defines $|\mathbf{R}| = R_1 R_2$ spatial frequency points within the period of the form $\mathbf{u}_\mathbf{k} = \boldsymbol{\Lambda}^{-T}\mathbf{R}^{-1}\mathbf{k}$, with $\mathbf{k}$ ranging over the set

$$\mathcal{K} = \{\mathbf{k} = (k_1, k_2)^T : k_1 = 0, \ldots, R_1 - 1; \quad k_2 = 0, \ldots, R_2 - 1\}.$$

Evaluating (1) at these points and substituting for $\mathbf{x_n}$ in terms of the lattice basis in the exponential yields

$$\begin{aligned} A(\mathbf{u_k}) &= \sum_{\mathbf{n}\in\mathbb{Z}^2} \mathrm{a}(\mathbf{x_n}) e^{j2\pi \mathbf{k}^T \mathbf{R}^{-1}\mathbf{n}} \\ &= \sum_{\mathbf{n}\in\mathcal{K}} \mathrm{a}'(\mathbf{x_n}) e^{j2\pi \mathbf{k}^T \mathbf{R}^{-1}\mathbf{n}} \end{aligned} \tag{6}$$

where

$$\mathrm{a}'(\mathbf{x_n}) = \sum_{\mathbf{m}\in\mathbb{Z}^2} \mathrm{a}\big(\mathbf{x_{(n-Rm)}}\big) \tag{7}$$

is the periodic "wrapping" of the array coefficient function a. For any practical $\mathbf{R}$ the sum of (7) will have at most one nonzero term, and so the operation (sometimes also referred to as *data turning*) represents only zero-padding and data rearrangement. The expression in (6) is, to within a scale factor, the standard separable 2D IDFT, which can be made more explicit by breaking it out as

$$A\big(\mathbf{u}_{(k_1,k_2)}\big) = \sum_{n_1=0}^{R_1-1} e^{j2\pi \frac{n_1 k_1}{R_1}} \sum_{n_2=0}^{R_2-1} e^{j2\pi \frac{n_2 k_2}{R_2}} \mathrm{a}'(\mathbf{x}_{(n1,n2)}).$$

This, to within a possible normalizing scale factor of $1/|\mathbf{R}|$, is the form computed by most standard 2D IFFT routines.

Having computed the array factor, we now approximate the integral of (2) as a Riemann sum over points in the lattice $\mathbf{\Lambda}^{-T}\mathbf{R}^{-1}\mathbb{Z}^2$:

$$f(\mathbf{a}) \approx \frac{1}{|\mathbf{\Lambda}||\mathbf{R}|} \sum_{\mathbf{k}\in\mathbb{Z}^2} W(\mathbf{u_k}) \Big| |A(\mathbf{u_k})|^q - D^q(\mathbf{u_k}) \Big|^p$$

where the leading factor is the area of the cell occupied by each point $\mathbf{u_k}$. Although in the most-common case W and D would have support over exactly one period of the array factor, it is not required. In cases of wide element spacing, for example, the visible space $\|\mathbf{u}\| \le 1$ is not contained within a single period of the array factor. In such cases it may make sense to define the support of W and D over the full visible part of the array factor.

Similarly approximating the integral of (5b) with a sum yields

$$\begin{aligned} \frac{\partial f(\boldsymbol{\phi})}{\partial \phi_\mathbf{n}} &\approx \mathrm{Im}\left\{ \frac{1}{|\mathbf{\Lambda}||\mathbf{R}|} \sum_{\mathbf{k}\in\mathbb{Z}^2} B(\mathbf{u_k}) e^{-j2\pi \mathbf{k}^T \mathbf{R}^{-1}\mathbf{n}} \rho_\mathbf{n} e^{-j\phi_\mathbf{n}} \right\} \\ &= \frac{1}{|\mathbf{\Lambda}||\mathbf{R}|} \mathrm{Im}\left\{ \rho_\mathbf{n} e^{-j\phi_\mathbf{n}} \left(\sum_{\mathbf{k}\in\mathcal{K}} B'(\mathbf{u_k}) e^{-j2\pi \mathbf{k}^T \mathbf{R}^{-1}\mathbf{n}} \right) \right\} \end{aligned} \tag{8}$$

where

$$B'(\mathbf{u_k}) = \sum_{\mathbf{m}\in\mathbb{Z}^2} B\big(\mathbf{u_{(k-Rm)}}\big) \tag{9}$$

is the periodic wrapping of B. (In this case the sum of (9) may contain more than one nonzero term if W is defined beyond one period.) The parenthesized expression in (8) is the standard form for the DFT. The gradient can therefore be found by first computing A via IFFT per (6), then computing B, and then (8) via FFT.

2.7 Computational Complexity

A straightforward (non-FFT) discretization of the gradient computation of (5) is essentially a matrix-vector multiply, and would have complexity $O(NM)$, where M is the number of points used to approximate the integral. Using the FFT as outlined above has complexity $O(|\mathbf{R}| \log_2(|\mathbf{R}|))$. For most cases of interest the FFT approach will be considerably faster, the exceptions being cases of small N or M that will still be quite fast regardless of the method. We can make the comparison more directly by noting that the number of sample points $|\mathbf{R}|$ is generally chosen proportional to the number of elements N (multipliers of 100–400 are typical), so that the asymptotic complexity can also be written simply as $O(N \log_2(N))$.

3. Pattern Error Minimization

3.1 Local Solvers

A large number of solvers exist to find local minima of an objective function given its gradient. The so-called *quasi-Newton* methods are often preferred, as they offer much better convergence properties than simple steepest-descent algorithms but do not require the overhead of Hessian matrix computation or inversion as do true Newton methods. Quasi-Newton methods instead update an estimate of the inverse-Hessian or its equivalent based only on the objective and its gradient. The original approaches, such as the eponymous BFGS method of Broyden, Fletcher, Goldfarb, and Shanno [47–50] still require storage of, and multiplication by, a full $N \times N$ matrix, with a per-iteration complexity of $O(N^2)$. These algorithms have previously been used for phase-only array synthesis [26, 36]. More recently limited-memory versions (L-BFGS) [51] have been developed that store only a much smaller $N \times L$ matrix to represent the Hessian, with per-iteration complexity of $O(NL)$. The penalty vs. full BFGS is typically a small increase in the number of required iterations to reach convergence.

Several quasi-Newton solvers with Matlab interfaces were tested on the example designs in the next section. Several L-BFGS implementations are freely available [52–57]. Of these, [54, 57] also provide conjugate-gradient methods, an alternative class of algorithms for gradient-based minimization with similar computational complexity. The `fminunc` function of the commercial Matlab optimization toolbox provides a full-memory BFGS implementation, although it was found to be substantially slower than the L-BFGS solvers. All solvers showed similar convergence performance, with the L-BFGS implementations generally requiring the least total time.

3.2 Initialization

As will be seen in the examples, the pattern-error objective has some (indeed many) local minima that are globally suboptimal. A common feature of these is characterized by example in Sec. 4.3. Since the various solvers find a local minima "close" to the starting point, choosing the initial element phases becomes key. Even for simple designs, experimentation quickly shows that choosing completely random phases rarely produces good solutions.

3.2.1 Approaches Used in the Examples

Although solving the local minima problem is not the primary goal of this paper, several heuristic methods were used to improve local minima quality. These include:

Symmetry

Imposing even symmetry on the initial phases can often improve local solutions, even if the ideal pattern is not itself symmetric. Circular symmetry can also prove useful, reducing a 2D initialization choice to 1D.

Phase Continuity

Starting with completely random phases implies starting with a random array factor which bears no resemblance to the desired pattern. Often a better choice is to sample a smooth phase function, such as a low-order polynomial. A second-order bivariate polynomial will produce a linear-FM pattern of the form considered in [5], with the polynomial coefficients controlling the starting beam position and beamwidth. (See Section 4.1 for an example.) Higher-order polynomials provide a richer set of initial phases but are less smooth.

Partial-Convergence Search

Often it is apparent that the current local solution is poor after relatively few iterations, well before full convergence. One way to exploit this is to try many initializations, optimize for a fixed number of iterations or until a trigger is activated (one such trigger is discussed in Section 4.3), and then fully optimize the best partial result. These test optimizations can be further sped up by tweaking parameters in the solvers (for example, the memory depth L in a L-BFGS algorithm) as well as reducing the number of sample points $|\mathbf{R}|$ that are used to approximate the integrals in the objective and gradient.

3.2.2 Other approaches

Although not considered further here, more-elaborate approaches have been suggested to mitigate the pattern-synthesis local minima problem, many coming from the reflectarray community. Some forego local optimization altogether, choosing instead metaheuristic approaches such as simulated annealing, particle-swarm optimization, and genetic algorithms [13–17]. The primary problem with these approaches is speed; [17], for example, describes a single example that took 44 hours to compute on a single CPU core. An approach in [16] that should be applicable to local minimization as well is to slowly walk the design parameters out from a uniform-illumination starting point, re-optimizing at each step.

Local optimization approaches generally involve a series of modified optimizations. One such approach is to first reduce the problem order by deriving the array weights from a polynomial basis with optimized coefficients [24, 58], followed by full optimization. Others include adaptively changing the pattern-error weighting [39], starting with a significant amplitude taper which is gradually reduced [28], or designing a series of increasing-size arrays [29]. Both global and local approaches are used in [26], with the global search applied to low-order polynomial models and the result used to seed the final full optimization.

4. Examples

In this section a series of examples is presented, with progressive complexity. A common array geometry is used for all examples. The array plane is inclined so that boresight points 15° above the horizon. The basis matrix for the elements in the plane is $\mathbf{\Lambda} = \begin{pmatrix} 1/\sqrt{3} & 1/(2\sqrt{3}) \\ 0 & 1/2 \end{pmatrix}$, which defines a hexagonal lattice with an interelement spacing of $\lambda/\sqrt{3}$. The array elements are centered on the 1075 points of the lattice that fall within a radius of 10λ from the origin. The 1 dB and 3 dB beamwidths of the array with uniform (identical) drive phases are about 1.7° and 3.0°, respectively.

For all examples, the resampling matrix of section 2.6 was set to $\mathbf{R} = \mathrm{diag}(512, 512)$, which results in just fewer than 244 spatial-frequency sample points per array element. Under Matlab on a dual-core Intel Core i7 laptop, the resulting objective computation took an average of 13 ms, while simultaneous computation of the objective and the gradient took 25 ms. FFT and IFFT computation contributed just under half of the total time.

4.1 Example 1: Circular Beam

For a first example, suppose we wish to form a broad flat-top circular boresight beam of radius 10°, which is approximately $r_{\mathbf{u}} = 0.17$ in units of normalized spatial frequency. The desired response D is thus constant over the circle and zero elsewhere. By definition the total energy in the array weights is N, and applying Parseval's relation reveals that the energy in one period $\Pi = \{\mathbf{\Lambda}^{-T}\mathbf{v} : \mathbf{v} \in ([0,1) \times [0,1))\}$ of the array factor is

$$\int_{\Pi} |A(\mathbf{u})|^2 \, d\mathbf{u} = |\mathbf{\Lambda}|^{-1} N. \tag{10}$$

If this energy were distributed evenly across the circular beam, the resulting array-factor height would be $D_0 = \sqrt{\frac{N}{|\mathbf{\Lambda}|\pi r_{\mathbf{u}}^2}}$. This is physically unrealizable, of course, as it implies zero sidelobes everywhere. Because we know that some of the total energy must lie in the sidelobes, we will define the relaxed mainlobe level $D_1 = \alpha D_0$, where $0 \le \alpha \le 1$ is the relaxation parameter. (When α is given in dB units the operation $20\log_{10}(\alpha)$ is implied.) This relaxed level will then be used in the optimization objective.

In conventional complex array weight optimization we would choose the error weight function W to be nonzero on both mainlobe and sidelobe regions, weighting each to get the desired balance between the errors. Here, however, the choice of D_1 and the implicit energy constraint (10) combine to effectively constrain the sidelobe region, so that we can set W to unity within the mainlobe and zero elsewhere. We argue this for two different cases. If we assume that the optimization successfully approximates the desired level D_1 in the mainlobe region, then the energy in the mainlobe is approximately $\pi r_{\mathbf{u}}^2 D_1^2 = \alpha^2 |\mathbf{\Lambda}|^{-1} N$. As a result of (10), this means that the sidelobe energy is approximately $(1 - \alpha^2)|\mathbf{\Lambda}|^{-1} N$. So in this case we have implicitly set the total sidelobe energy and allowed the optimization to allocate it so as to minimize mainlobe error. On the other hand, if D_1 is set too high for the optimization to approximate in the mainlobe, this implies that the design has run up against the fundamental Fourier limitations of finite arrays. In this case the optimization is essentially pushing upward on the mainlobe at optimality, and thus implicitly is simultaneously trying to minimize sidelobe energy. We can augment the implicit energy constraint with explicit sidelobe weighting in W, but will defer that to the second example.

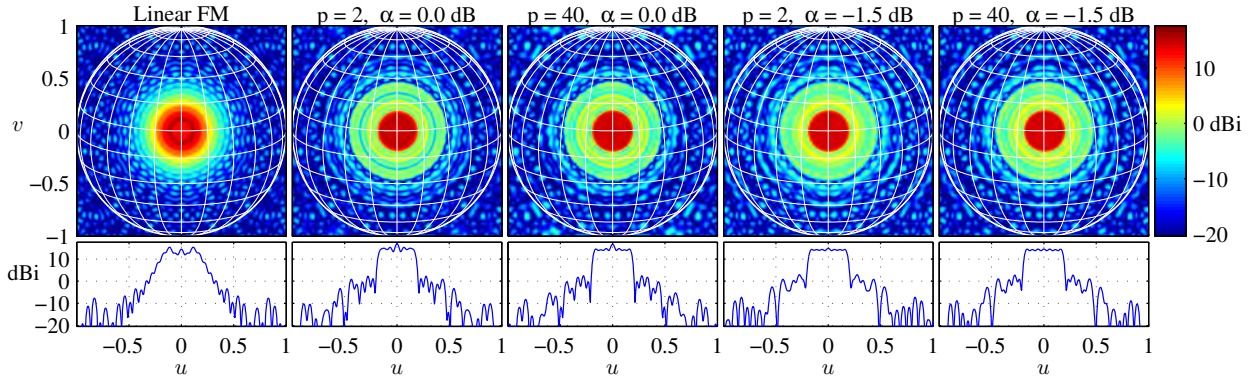
Because the desired pattern is circularly symmetric, the initial phases were chosen to be circularly symmetric as well. A second-order polynomial in the element distance from array center was used, with the polynomial coefficients chosen from a uniform distribution. Ten such polynomials were drawn and used to generate ten prospective sets of starting phase values. Then, for each of the four combinations of $p \in \{2, 40\}$ and $\alpha \in \{0\,\text{dB}, -1.5\,\text{dB}\}$ with D and W as described above and $q = 1$, the objective (2) was minimized for a maximum of 20 iterations per initial phase set. The best result of the ten for each subdesign was then fully optimized. The time to test all ten initializations using [52] was about 2 s per subdesign, while the time to fully optimize the best candidate ranged from 0.1 s to 0.6 s. Convergence was typically achieve in less than 50 iterations.

A special case of the second-order polynomial used is a pure quadratic in the element distance. In [5] this was shown to amount to 2D linear FM modulation, which in the limit of an infinite circular array approaches an ideal flat-top circular beam. The single polynomial coefficient sets the beamwidth, and for practical array sizes also affects the mainbeam ripple characteristic. For comparison to the L_p-optimized designs, this single parameter was optimized to maximize the minimum gain within the mainlobe.

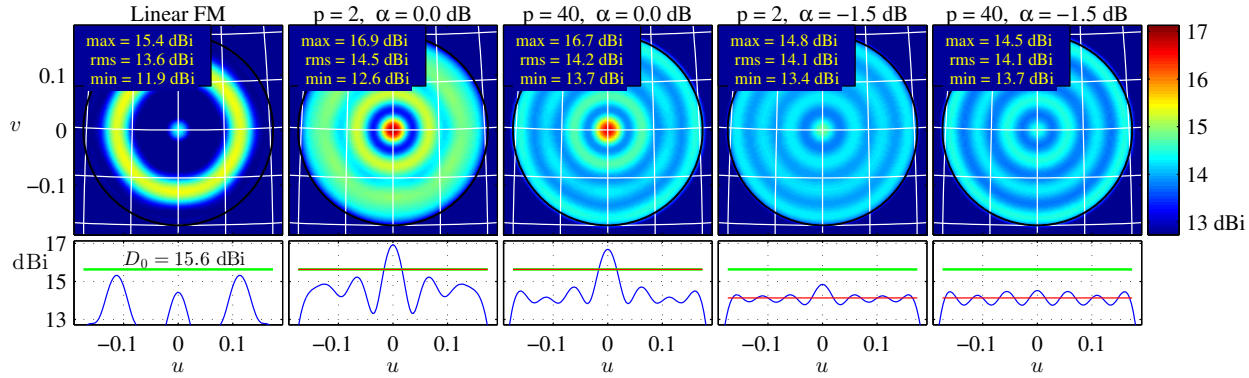
The resulting array factors are shown in Fig. 1a. They are scaled relative to the pattern of an isotropic antenna radiating the same power; for unit-magnitude coefficients this is equal to $A(\mathbf{u})/\sqrt{N}$. Visually all four L_p designs look very similar in Fig. 1a, which shows the patterns over all visible spatial frequencies, with differences primarily in the mainlobe. The linear FM pattern, by comparison, has significantly more ripple and wider transition bands. The mainlobe detail is shown in Fig. 1b, and here we can most clearly see the effect of the values of p and α. When $\alpha = 1$ (0 dB), we see that the optimized mainlobe falls well below the target D_0 level except at the center. When we relax the desired mainlobe by 1.5 dB, then we see the optimized mainlobe ripple about the desired level as in classic digital filter designs. As we might expect, the $p = 2$, $\alpha = 0\,\text{dB}$ case provides the greatest rms value in the mainlobe, only 1.1 dB below the ideal D_0 value, and thus concentrates the most energy in the mainlobe. However, it has a worse minimum value in the mainlobe than either $p = 40$ case, which are essentially equiripple owing to the large p. In many broad-beam applications it is this maximin pattern value, representing worst-case gain, that is most important. (Despite being optimized specifically for this metric, the linear FM weighting is significantly worse than all of the L_p designs on this score.) Adding -1.5 dB relaxation to the $p = 40$ case has minimal effect on either rms or minimum mainlobe values and simply causes the energy from the large central ripple to be distributed to the sidelobes. Further relaxing the mainbeam height does result in an increasingly flat mainlobe at the expense of greater sidelobe energy.

4.2 Example 2: Circular Beam w/ Sidelobe Suppression

Building on the previous example, we now enforce a zone of increased sidelobe suppression from 2° below to 2° above the horizon. Such a zone could be useful to mitigate multipath, for example, or to reduce interference with other systems. The extra suppression is achieved by setting D to zero in this region, and setting W to a nonzero constant. Adjusting the relative value of W on the mainlobe and sidelobe regions allows us to tradeoff the residual errors in the two regions. As before, the rest of the sidelobes are allowed to be determined by the optimization. The initialization is handled as in the previous example, with parameters $p = 2$ and $\alpha = -1.5\,\text{dB}$. Although the computational complexity of each iteration is identical to the previous example, more iterations (100 was typical) are required for convergence, resulting in an average total solution time of 3 s. In general, such closely spaced mainlobe and sidelobe specifications appear to produce error surfaces that are not locally well approximated as a quadratic form and result in slower convergence.



(a) Full visible space array factors. The lines on the top plot indicate constant azimuth and elevation at 15° intervals, with array boresight at 15° elevation. The bottom row shows azimuth slices through the center of the mainlobe.



(b) Mainlobe detail. The lines on the top plot indicate constant azimuth and elevation at 5° intervals. The bottom row shows azimuth slices through the center of the mainlobe. The green (upper) line segment indicates the theoretical ideal array-factor height D_0, while the red line indicates the relaxed height D_1.

Fig. 1: Optimized array factors for the Example 1 designs.

The optimized array factor is shown in Fig 2, with azimuth and elevation cuts shown above and to the right. The deep suppression around the horizon is clearly visible, with the peak and rms horizon sidelobe levels indicated on the plot. The max, min, and rms values in the mainlobe are also shown on the plot, and reveal that the "cost" of the extra sidelobe suppression is a loss of 0.2 dB of mainlobe energy and 0.7 dB of worst-case gain. Visually, we can see that the sidelobes at elevations above the mainlobe have increased relative to the previous design.

Figure 3 is a different view of the same optimized array factor, to illustrate the relationship with the functions D and W. The parallelogram bounds one period of the array factor, over which W and D are defined. The two sides meeting at the origin represent the columns of the dual lattice basis $\mathbf{\Lambda}^{-T} = \begin{pmatrix} \sqrt{3} & 0 \\ -1 & 2 \end{pmatrix}$. Superimposed on the plot are (tiny) white dots at every nonzero sample of W.

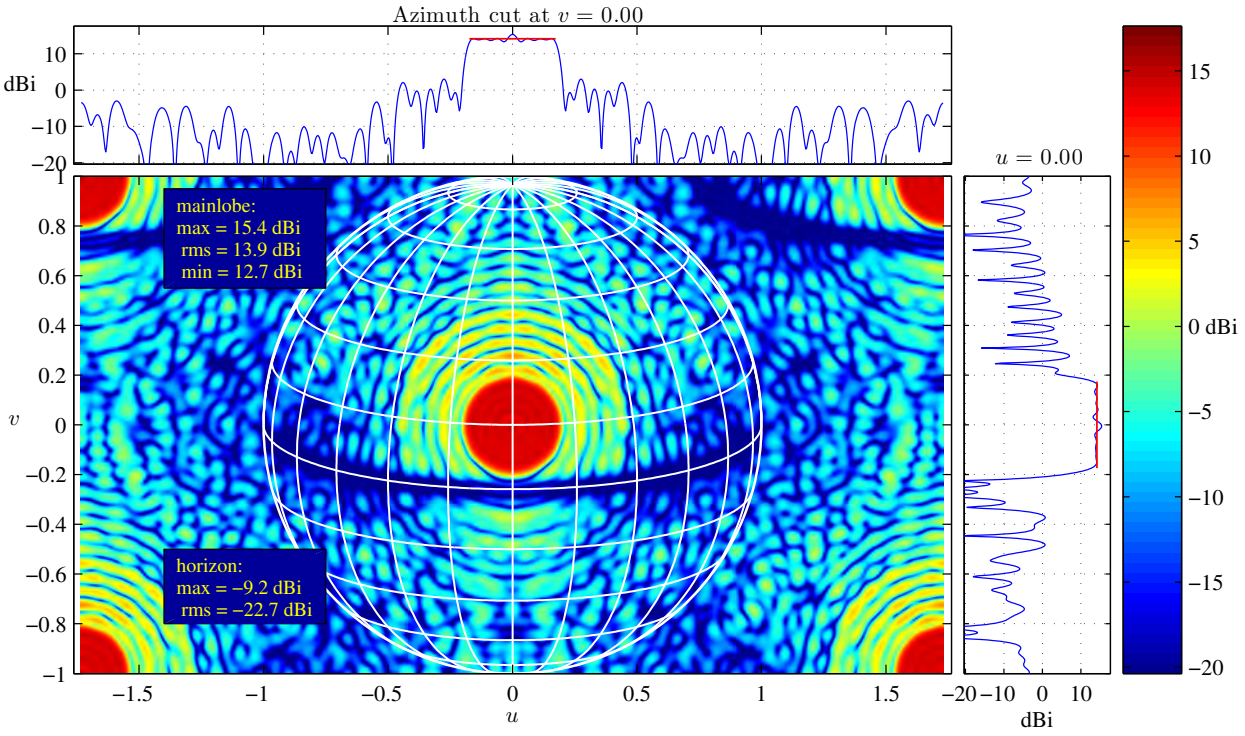


Fig. 2: Optimized array factor for the second example. The plots above and to the right are azimuth and elevation slices through the beam center. The red line segments indicate the desired array factor level D_1.

4.3 Local Minima Characterization

The complicated error surface for the objective (2) means that numerous (globally) suboptimal local minima exist for most designs. Local solvers will find the "closest" (in some sense) minimum to the point used to initialize the optimization. Thus the results can be very sensitive to a change in the initial phases. In the examples presented so far, the combination of a smooth and circularly symmetric phase initialization function and running 10 partial trials was almost always sufficient to produce a good local (and possibly global) solution. With more-complicated desired geometries it can be hard to avoid "bad" local minima.

Here we define "bad" local minima specifically as those with point nulls within the mainbeam. Reference [29] mentions, somewhat in passing, a local minima characteristic consisting of "holes" or point nulls in the mainbeam of a pattern designed using a least-squares approach. The same nulls were seen both with and without the phase-only constraint, and thus are due to optimizing only the magnitude (rather than complex) pattern. Point-null artifacts have also been described as the Fourier-domain counterpart to the image-domain "stripes" observed and characterized in phase-retrieval problems [59].

Consider Fig. 4a, which results from choosing the worst of the 10 initial phases rather than the best in Example 2. Eight point nulls are clearly visible. (The indicated minimum mainbeam value is an artifact of the grid used for plotting and computation; each point is a true null to working precision.) The mainlobe details

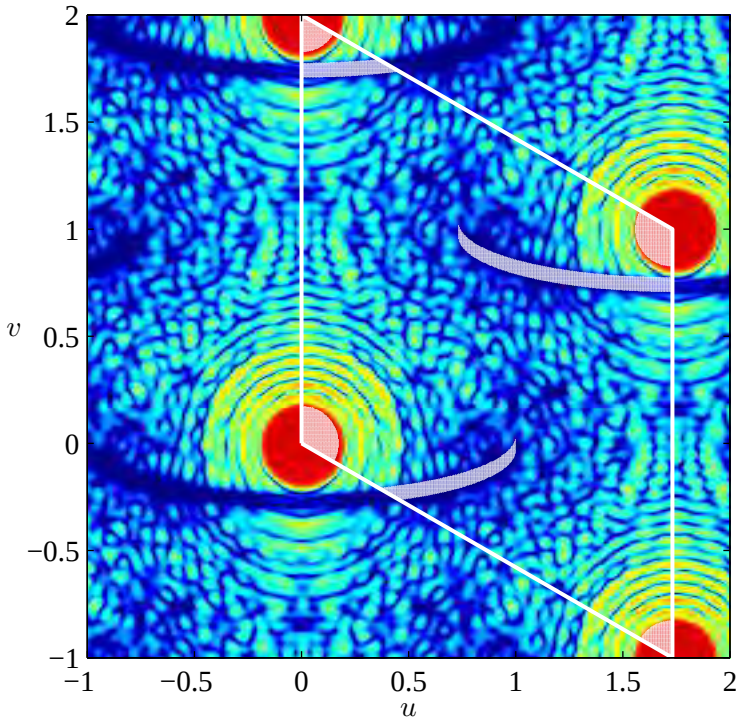
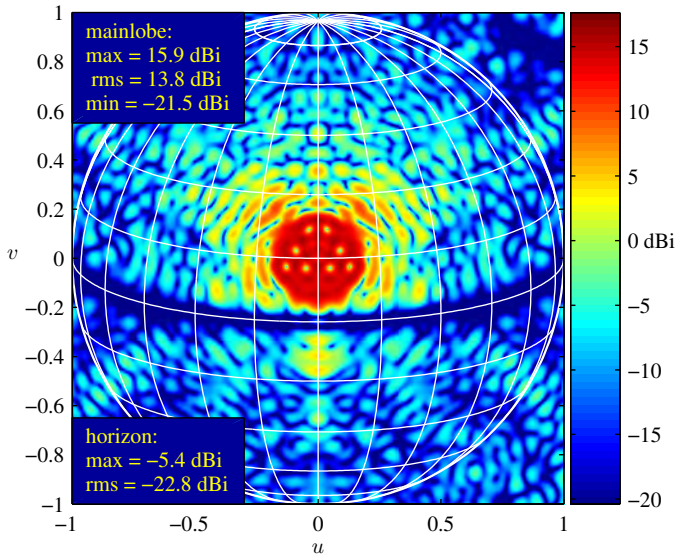


Fig. 3: Illustration of the geometry of the D and W functions. The parallelogram bounds one period of the array factor. Overlaid onto the period are (tiny) white dots for each nonzero sample of W, defining the mainlobe and horizon regions.

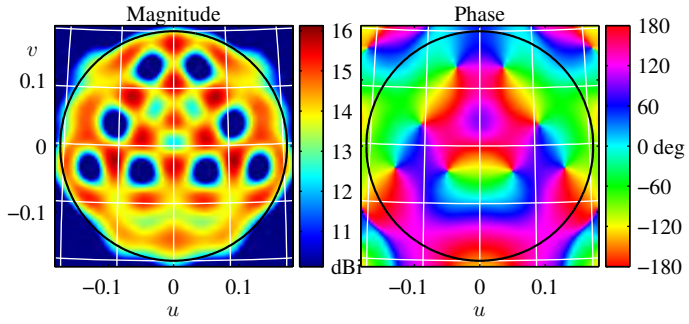
in magnitude and phase are shown in Fig. 4b. In the immediate orbit of each null, the phase of the array factor rotates a full 360 degrees [59]. Along any line drawn through a null location the phase is discontinuous (at the null), and since the complex array factor (as a finite discrete-space Fourier transform) is continuous, the magnitude must go to zero at that point.

As noted in [29, 59], the nulls are narrow enough that they do not greatly affect a squared-error objective. One might expect here that increasing p would directly mitigate the point nulls, since they will then increasingly dominate the objective. Experimentation has shown otherwise, with point nulls appearing for any p. Similar nulls arise when using alternating-projection approaches [36, 59]. Thus point nulls appear to be a common feature of magnitude/power pattern synthesis problems, and not specific to a given formulation.

There are (at least) three exploitable properties of such nulls: they tend to form early in the solver iteration, they are stable once formed, and they are easy to detect while computing (2). This can be used to trigger an early solver termination, in order to move on to another starting point. For example, the minimum value of the ratio $|A(\mathbf{u})|/D(\mathbf{u})$ over the mainbeam region after 20–60 iterations has proved to be a reliable indicator of point nulls [60].



(a) All visible space.



(b) Details of the magnitude and phase response in the mainlobe. The nulls in magnitude correspond to discontinuities in phase.

Fig. 4: Array factor with point nulls for a representative "bad" local minimum of the Example 2 design.

4.4 Example 3: Sector Beam w/ Nonuniform Amplitude

Finally, we consider an example with a noncircular mainbeam which is not centered at boresight. The mainbeam region is the sector between 0° and 30° in azimuth and between 5° and 60° in elevation. (A similar design can be found in [27].) Rather than a flat desired pattern, we seek a pattern that provides uniform radar detection sensitivity at the lesser of 100 km and the range at 40 km altitude. (The actual units are irrelevant, only the ratio matters.) The result is a beam that is flat for low elevation angles and falls off (as cosecant-squared in magnitude) for higher elevation angles as the altitude limit kicks in. As before we scale down the desired response from the ideal to account for sidelobe energy, in this case by 1 dB. We retain the horizon sidelobe suppression region from Example 2. Initialization is done similarly to the previous examples, even though now the desired mainbeam is far from circularly symmetric. Two changes to increase initialization diversity: the polynomial order of the phase initializing functions is increased to four, and 50

starting points are tested at a time. Even with these extra initial values, multiple runs are sometimes required to find a single null-free solution. Here initialization took about 9 s, and final solve took over 200 iterations and about 1.8 s.

The array factor for one solution is shown in Fig. 5. The $\csc^2$ profile is clearly visible in the elevation slice, and we achieve a deeper horizon depth than in Example 2 (at the expense of higher sidelobes) by increasing the weight on that region of W.

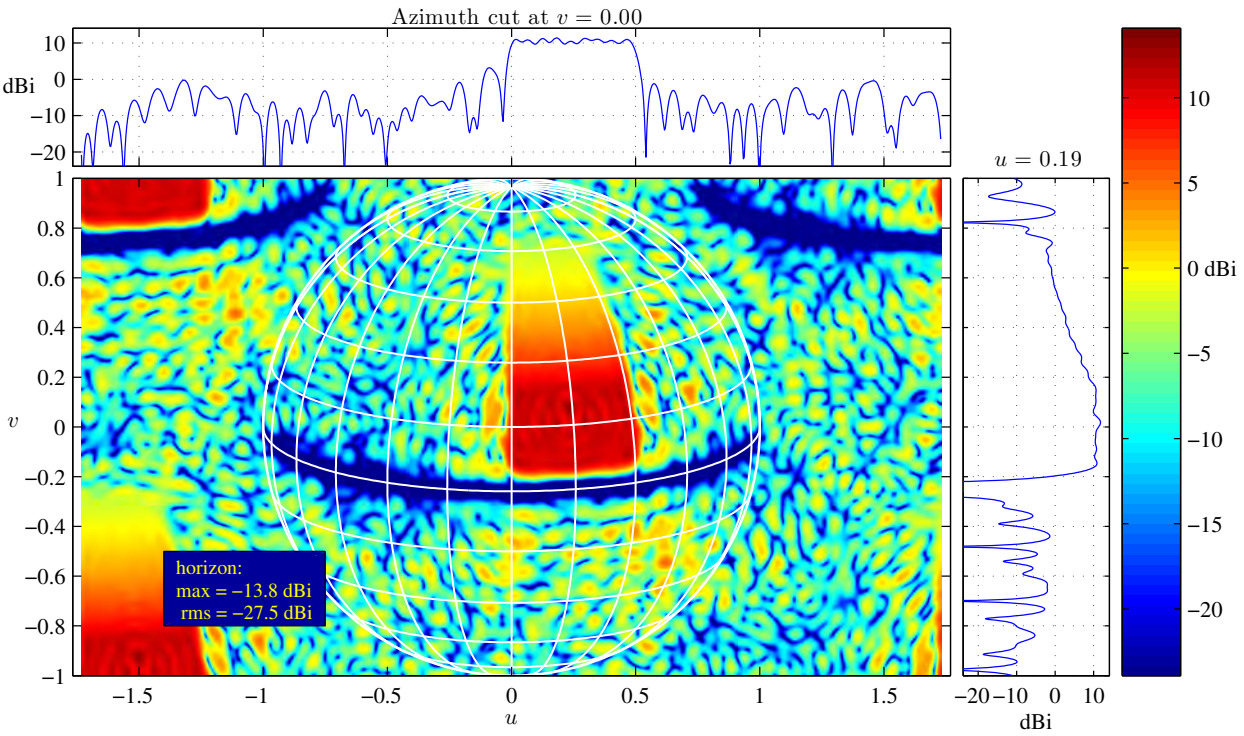


Fig. 5: Optimized array factor for the third example. The plots above and to the right are azimuth and elevation slices through the beam center.

5. Conclusion

Phase-only array pattern optimization is both less common and harder to solve than conventional receive-pattern synthesis. Modern tools have made general-purpose optimization far more accessible and efficient, and are exploited here with an efficient yet reasonably flexible phase-only objective function that mirrors common aproaches from the non-phase-only world. The examples demonstrate that it is now quite feasible to quickly and directly optimize thousands of element phases to approximate arbitrary responses. The nonconvex pattern-error objective has many local minima with catastrophic features such as nulls in the midst of the mainlobe. For simpler designs fairly simple heuristics were demonstrated to avoid such bad local minima.

These were found to be less effective for more complicated designs, and so more work is needed to produce a fully unsupervised algorithm.

ACKNOWLEDGMENTS

This work was sponsored by the Office of Naval Research under an NRL 6.1 Base Program.

REFERENCES

1. J. Hoversten, S. Schafer, M. Roberg, M. Norris, D. Maksimovic, and Z. Popovic, "Co-design of PA, supply and signal processing for linear supply-modulated RF transmitters," *IEEE Trans. Microw. Theory Tech.*, vol. 60, no. 6, pp. 2010 –2020, Jun. 2012.

2. M. Roberg, M. Rodriguez, D. Maksimovic, and Z. Popovic, "Efficient and linear amplification of spectrally confined pulsed AM radar signals," *IEEE Microw. Wireless Compon. Lett.*, vol. 22, no. 6, pp. 279–281, Jun. 2012.

3. C. Baylis, L. Wang, M. Moldovan, J. Martin, H. Miller, L. Cohen, and J. de Graaf, "Designing transmitters for spectral conformity: power amplifier design issues and strategies," *Radar, Sonar Navigation, IET*, vol. 5, no. 6, pp. 681 –685, Jul. 2011.

4. E. Fowle, "The design of FM pulse compression signals," *IEEE Trans. Inf. Theory*, vol. 10, no. 1, pp. 61–67, 1964.

5. J. O. Coleman, D. P. Scholnik, and K. R. McPhail, "Phase-only tapers for regular planar arrays, a heuristic nonlinear-FM approach," in *Proc. IEEE Int'l Symp. on Phased Array Systems & Technology*, Boston, MA, Oct. 2010, pp. 113–120.

6. P. Sridevi, G. Raju, and P. Misra, "Effect of digital phase shifters on phase only controlled SLS difference patterns," in *Proc. Int'l Conf. on Electromagnetic Interference and Compatibility*, 1997, pp. 163–166.

7. R. Kinsey, "Phased array beam spoiling technique," in *IEEE Int'l Symp. Antennas and Propagation*, vol. 2, 1997, pp. 698–701 vol.2.

8. H. Steyskal, "Simple method for pattern nulling by phase perturbation," *IEEE Trans. Antennas Propag.*, vol. 31, no. 1, pp. 163–166, 1983.

9. A. Chakraborty, B. Das, and G. Sanyal, "Determination of phase functions for a desired one-dimensional pattern," *IEEE Trans. Antennas Propag.*, vol. 29, no. 3, pp. 502–506, 1981.

10. ——, "Beam shaping using nonlinear phase distribution in a uniformly spaced array," *IEEE Trans. Antennas Propag.*, vol. 30, no. 5, pp. 1031–1034, 1982.

11. A. S. Dunbar, "On the theory of antenna beam shaping," *Journal of Applied Physics*, vol. 23, no. 8, pp. 847–853, 1952.

12. R. Young, "Antenna pattern control by phase-only weighting," in *IEE Colloquium on Phased Arrays*, 1991, pp. 5/1–5/7.

13. R. Haupt, "Phase-only adaptive nulling with a genetic algorithm," *IEEE Trans. Antennas Propag.*, vol. 45, no. 6, pp. 1009–1015, 1997.

14. G. Mahanti, A. Chakraborty, and S. Das, "Floating-point genetic algorithm for design of a reconfigurable antenna arrays by phase-only control," in *Proc. Asia-Pacific Microwave Conf.*, vol. 5, 2005, p. 3 pp.

15. F. Grimaccia, M. Mussetta, P. Pirinoli, and R. Zich, "Optimization of a reflectarray antenna via hybrid evolutionary algorithms," in *Int'l Zurich Symposium on Electromagnetic Compatibility*, Feb. 2006, pp. 254–257.

16. G. Brown, J. Kerce, and M. Mitchell, "Extreme beam broadening using phase only pattern synthesis," in *IEEE Workshop on Sensor Array and Multichannel Processing*, 2006, pp. 36–39.

17. P. Nayeri, F. Yang, and A. Elsherbeni, "Design of single-feed reflectarray antennas with asymmetric multiple beams using the particle swarm optimization method," *IEEE Trans. Antennas Propag.*, vol. 61, no. 9, pp. 4598–4605, Sep. 2013.

18. A. Trastoy, F. Ares, and E. Moreno, "Phase-only control of antenna sum and shaped patterns through null perturbation," *IEEE Antennas Propag. Mag.*, vol. 43, no. 6, pp. 45–54, 2001.

19. R. Voges and J. Butler, "Phase optimization of antenna array gain with constrained amplitude excitation," *IEEE Trans. Antennas Propag.*, vol. 20, no. 4, pp. 432–436, 1972.

20. E. Dufort, "Pattern synthesis based on adaptive array theory," *IEEE Trans. Antennas Propag.*, vol. 37, no. 8, pp. 1011–1018, 1989.

21. D. Chang, C. Hung, C. Hu, K. Ho, and H. Chen, "Synthesis of array antenna with broad nulls," in *Proc. Asia-Pacific Microwave Conf.*, vol. 1, 1993, pp. 88–92.

22. A. Khzmalyan and A. Kondratiev, "The phase-only shaping and adaptive nulling of an amplitude pattern," *IEEE Trans. Antennas Propag.*, vol. 51, no. 2, pp. 264–272, 2003.

23. S. Sun and H. Li, "Simplified beamforming on phase-only arrays," in *Proc. Int'l Conf. on Neural Networks and Signal Processing*, vol. 2, 2003, pp. 1290–1293.

24. I. D. Olin, "Flat-top sector beams using only array element phase weighting: A metaheuristic optimization approach," Naval Research Laboratory, Washington, DC, Tech. Rep. FR-10227, Oct. 2012.

25. O. Bucci, G. D'Etia, and G. Romito, "Optimal synthesis of reconfigurable conformal arrays with phase only control," in *IEEE Int'l Symp. Antennas and Propagation*, vol. 2, 1996, pp. 810–813 vol.2.

26. A. Capozzoli, C. Curcio, G. D'Elia, and A. Liseno, "Fast phase-only synthesis of conformal reflectarrays," *IET Microwaves, Antennas Propagation*, vol. 4, no. 12, pp. 1989–2000, Dec. 2010.

27. M. Arrebola, J. Encinar, and M. Barba, "Multifed printed reflectarray with three simultaneous shaped beams for LMDS central station antenna," *IEEE Trans. Antennas Propag.*, vol. 56, no. 6, pp. 1518–1527, Jun. 2008.

28. J. A. Zornoza and J. A. Encinar, "Efficient phase-only synthesis of contoured-beam patterns for very large reflectarrays," *Int'l Journal of RF and Microwave Computer-Aided Engineering*, vol. 14, no. 5, pp. 415–423, Sep. 2004.

29. L. Vaskelainen, "Phase synthesis of conformal array antennas," *IEEE Trans. Antennas Propag.*, vol. 48, no. 6, pp. 987–991, Jun. 2000.

30. O. Bucci, G. Franceschetti, G. Mazzarella, and G. Panariello, "Intersection approach to array pattern synthesis," *Microwaves, Antennas and Propagation, IEE Proceedings H*, vol. 137, no. 6, pp. 349–357, Dec. 1990.

31. O. Bucci, G. D'Elia, G. Mazzarella, and G. Panariello, "Antenna pattern synthesis: a new general approach," *Proceedings of the IEEE*, vol. 82, no. 3, pp. 358–371, Mar. 1994.

32. D. Day, "Fast phase-only pattern nulling for large phased array antennas," in *Proc. IEEE Radar Conference*, 2009, pp. 1–4.

33. J. O. Coleman and W. M. Dorsey, "Phase-only beam broadening in large transmit arrays using complex-weight gradient descent," in *IEEE Int'l Symp. on Phased Array Systems & Technology*, Waltham MA, Oct. 2013, accepted.

34. H. Lebret and S. Boyd, "Antenna array pattern synthesis via convex optimization," *IEEE Trans. on Signal Processing*, vol. 45, no. 3, pp. 526–32, Mar. 1997.

35. D. Scholnik and J. Coleman, "Optimal array-pattern synthesis for wideband digital transmit arrays," *IEEE J. Sel. Topics Signal Process.*, vol. 1, no. 4, pp. 660–677, 2007.

36. O. Bucci, G. D'Elia, and G. Romito, "Power synthesis of conformal arrays by a generalised projection method," *IEE Proc. Microwaves, Antennas and Propagation*, vol. 142, no. 6, pp. 467–471, Dec. 1995.

37. T. Isernia, F. Soldovieri, G. Leone, and R. Pierri, "On the local minima in phase reconstruction algorithms," *Radio Science*, vol. 31, no. 6, pp. 1887–1899, Nov. 1996.

38. T. Isernia, G. Leone, R. Pierri, and F. Soldovieri, "Role of support information and zero locations in phase retrieval by a quadratic approach," *Journal of the Optical Society of America A*, vol. 16, no. 7, pp. 1845–1856, Jul. 1999.

39. L. Vaskelainen, "Iterative least-squares synthesis methods for conformal array antennas with optimized polarization and frequency properties," *IEEE Trans. Antennas Propag.*, vol. 45, no. 7, pp. 1179–1185, Jul. 1997.

40. J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY: Springer-Verlag, 1999.

41. A. Morabito, A. Massa, P. Rocca, and T. Isernia, "An effective approach to the synthesis of phase-only reconfigurable linear arrays," *IEEE Trans. Antennas Propag.*, vol. 60, no. 8, pp. 3622–3631, Aug. 2012.

42. D. E. Dudgeon and R. M. Mersereau, *Multidimensional digital signal processing*. Prentice-Hall, 1984.

43. A. Dutt and V. Rokhlin, "Fast fourier transforms for nonequispaced data," *SIAM Journal on Scientific Computing*, vol. 14, no. 6, pp. 1368–1393, Nov. 1993.

44. ——, "Fast fourier transforms for nonequispaced data, II," *Applied and Computational Harmonic Analysis*, vol. 2, no. 1, pp. 85–100, Jan. 1995.

45. L. Greengard and J. Lee, "Accelerating the nonuniform fast fourier transform," *SIAM Review*, vol. 46, no. 3, pp. 443–454, Jan. 2004.

46. A. Capozzoli, C. Curcio, G. D'Elia, A. Liseno, and P. Vinetti, "Fast CPU/GPU pattern evaluation of irregular arrays," *Applied Comput. Electromagn. Soc. J., Special Issue on Hardware Accelerated Computational Techniques for Electromagnetic Simulations of Complex Problems*, vol. 25, no. 4, pp. 355–372, Apr. 2010.

47. C. G. Broyden, "The convergence of a class of double-rank minimization algorithms 1. General considerations," *IMA Journal of Applied Mathematics*, vol. 6, no. 1, pp. 76–90, Mar. 1970.

48. R. Fletcher, "A new approach to variable metric algorithms," *The Computer Journal*, vol. 13, no. 3, pp. 317–322, Jan. 1970.

49. D. Goldfarb, "A family of variable metric methods derived by variational means," *Mathematics of computation*, vol. 24, no. 109, pp. 23–26, 1970.

50. D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics of computation*, vol. 24, no. 111, pp. 647–656, 1970.

51. D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, Aug. 1989.

52. S. G. Johnson, "The NLopt nonlinear-optimization package." [Online]. Available: http://ab-initio.mit.edu/nlopt

53. L. Sorber, M. V. Barel, and L. D. Lathauwer, "Tensorlab v2.0," Jan. 2014. [Online]. Available: http://www.tensorlab.net

54. M. Schmidt, "minFunc - Unconstrained differentiable multivariate optimization in Matlab," 2012. [Online]. Available: http://www.di.ens.fr/~mschmidt/Software/minFunc.html

55. N. Okazaki, "libLBFGS: A library of limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS)." [Online]. Available: http://www.chokkan.org/software/liblbfgs/

56. D. M. Dunlavy, T. G. Kolda, and E. Acar, "Poblano v1.0: A Matlab toolbox for gradient-based optimization," Sandia National Laboratories, Albuquerque, NM and Livermore, CA, Tech. Rep. SAND2010-1422, Mar. 2010.

57. W. W. Hager and H. Zhang, "The limited memory conjugate gradient method," 2012. [Online]. Available: http://www.math.ufl.edu/~hager/papers/CG/lcg.pdf

58. L. Marcaccioli, R. V. Gatti, and R. Sorrentino, "Series expansion method for phase-only shaped beam synthesis and adaptive nulling," *URSI EMTS*, 2004.

59. J. R. Fienup and C. C. Wackerman, "Phase-retrieval stagnation problems and solutions," *Journal of the Optical Society of America A*, vol. 3, pp. 1897–1907, 1986.

60. W. M. Dorsey and D. P. Scholnik, "A hybrid global-local optimization approach to phase-only array-pattern synthesis," in *Proc. IEEE Int'l Radar Conference*, Arlington, VA, May 2015.

Appendix A

IMPLICATIONS OF SYMMETRY FOR PHASE-ONLY ARRAY FACTORS

Symmetries are often imposed on filter and array coefficients, primarily for the purposes of reducing the number of optimization variables without significantly impacting the achievable magnitude response. Several common symmetries and their corresponding restrictions on the array factors are summarized below:

$\mathrm{a}(\mathbf{x}) = \pm\mathrm{a}(-\mathbf{x})$	$\longleftrightarrow$	$A(\mathbf{u}) = \pm A(-\mathbf{u})$
Even/Odd symmetry		Even/Odd symmetry
$\mathrm{a}(\mathbf{x}) = \pm\mathrm{a}^*(-\mathbf{x})$	$\longleftrightarrow$	$A(\mathbf{u}) = \pm A^*(\mathbf{u})$
Conjugate symmetry/antisymmetry		Real/Imaginary
$\mathrm{a}(\mathbf{x}) = \pm\mathrm{a}^*(\mathbf{x})$	$\longleftrightarrow$	$A(\mathbf{u}) = \pm A^*(-\mathbf{u})$
Real/Imaginary		Conjugate symmetry/antisymmetry

Other symmetries are possible in 2D, but these will suffice for the present discussion. Of these, some have obvious drawbacks. A restriction to purely real or imaginary weights would be of limited use in the phase-only case, since it would restrict the weights to ± 1 or $\pm j$, respectively. Odd symmetry implies $A(\mathbf{0}) = 0$, and thus precludes any mainbeam that included boresight.

Conjugate (linear-phase) symmetry initially appears very attractive because it reduces the number of optimization variables by half with no immediately obvious restriction on the magnitude response. Indeed, conjugate symmetry is widely used when array/filter weights are not magnitude constrained. As shown in the following, however, conjugate symmetry combined with the phase only constraint imposes a rather strict implicit constraint on the array factor mainlobe. We write the phase only constraint as $|\mathrm{a}(\mathbf{x})|^2 = \mathrm{a}(\mathbf{x})\mathrm{a}^*(\mathbf{x}) = 1_{\boldsymbol{\mathcal{X}}}(\mathbf{x})$, where $1_{\boldsymbol{\mathcal{X}}}$ is the indicator function on the array support $\boldsymbol{\mathcal{X}}$. To simplify the argument, we will assume that $\boldsymbol{\mathcal{X}}$ lies on a lattice so that A is periodic. A Fourier convolution property then allows us to write

$$(A * A^\dagger)(\mathbf{u}) = A_1(\mathbf{u}) \tag{A1}$$

where $*$ indicates (periodic) convolution, $A^\dagger(\mathbf{u}) \triangleq A^*(-\mathbf{u})$ is the paraconjugate operation, and A_1 is the array factor resulting from all unity weights. Now A is real and thus entirely positive or negative valued on any null-free mainlobe region. Thus for a conventional array factor containing a single mainlobe surrounded by significantly lower sidelobes, the left side of (A1) will be a boresight beam with a single mainlobe of roughly twice the width of the mainlobe of A. However, we know that A_1 has the narrowest mainlobe of all array factors on the support $\boldsymbol{\mathcal{X}}$, in the sense that it maximizes $|A(\mathbf{0})|$ for a fixed total weight energy $\|\mathbf{a}\|^2$. Thus, evidently imposing conjugate-symmetric symmetry on phase-only weights precludes synthesizing a mainlobe significantly broader than that of A_1. A similar argument holds for conjugate antisymmetry.

This would appear to leave even symmetry as the most viable symmetry out of those listed for phase-only array factors, although its utility is limited by the corresponding restriction on the array factor.