Phase Code Discovery for Pulse Compression Radar: A Genetic Algorithm Approach

Xinyan Xie, Runxin Zhang, Yulin Shao, *Member, IEEE*, Lu Lu, *Member, IEEE*

***Abstract*—Discovering sequences with desired properties has long been an interesting intellectual pursuit. In pulse compression radar (PCR), discovering phase codes with low aperiodic autocorrelations is essential for a good estimation performance. The design of phase code, however, is mathematically non-trivial as the aperiodic autocorrelation properties of a sequence are intractable to characterize. In this paper, we put forth a genetic algorithm (GA) approach to discover new phase codes for PCR with the mismatched filter (MMF) receiver. The developed GA, dubbed GASeq, discovers better phase codes than the state of the art. At a code length of 59, the sequence discovered by GASeq achieves a signal-to-clutter ratio (SCR) of 50.84, while the best-known sequence has an SCR of 45.16. In addition, the efficiency and scalability of GASeq enable us to search phase codes with a longer code length, which thwarts existing deep learning-based approaches. At a code length of 100, the best phase code discovered by GASeq exhibit an SCR of 63.23.**

***Index Terms*—Genetic algorithm, pulse compression radar, phase code, mismatched receiver, signal-to-clutter ratio.**

I. Introduction

Pulse compression radar (PCR) is a class of radar that solves the detection range and resolution trade-off in classical radar systems [1]. As illustrated in Fig. 1, the principles of PCR are 1) modulating the transmitted pulse by a sequence of phase codes, and 2) matched filtering (MF) the received signal by the same phase code, which is known as the MF receiver [2]. In so doing, PCR has both a long detection range, because the power of the whole pulse is collected by MF, and a high detection resolution, because the modulated pulse exhibits a large bandwidth – the detection resolution of radar is proportional to the bandwidth of the transmitted pulse [3].

When developing a PCR system, the main design objective is sidelobe suppression by carefully crafted phase codes and the receiver structure. High sidelobes are detrimental to PCR as they increase both the miss-detection rate and the false alarm rate [4]. In the literature, there are two main receiver structures: the MF receiver [5]–[7] and the mismatched filter (MMF) receiver [8]–[10].

X. Xie, R. Zhang, and L. Lu are with the University of Chinese Academy of Sciences, Beijing, 100049, China, and also with the Key Laboratory of Space Utilization, Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, Beijing, 100094, China (emails: {xiexinyan20, zhangrunxin20}@mails.ucas.ac.cn, lulu@csu.ac.cn).

Y. Shao is with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: y.shao@imperial.ac.uk).

This work was supported in part by the Key Research Program of the Chinese Academy of Sciences under Grant ZDRW-KT-2019-1-0103.

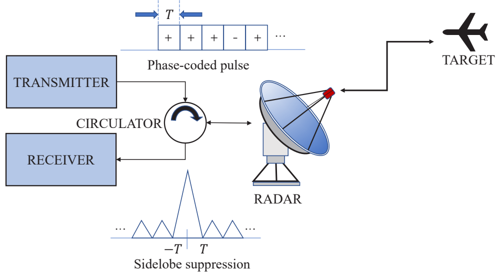

Fig. 1. Schematic diagram of PCR: a phase-coded pulse is emitted; the design objective of the phase code and receiver is sidelobe suppression.

Most research efforts on PCR have been devoted to the design of binary phase code with the MF receiver [7]. In actuality, this problem corresponds to the famous merit factor problem in complex analysis, that is, discovering the binary sequence with the smallest aperiodic autocorrelations. The main techniques are theoretical approaches [5], constrained search [6], exhaustive computation, and stochastic search. We refer readers to the excellent survey [7] for more detailed explanations. On the other hand, as far as the PCR estimation performance is concerned, the other line of receiver design, i.e., the MMF receiver, is more promising. Unlike the MF receiver that cross-correlates the received pulse by the same binary phase code, the MMF receiver utilizes a real sequence to collect the received power [8], [10] and optimizes the real sequence such that the signal-to-clutter ratio (SCR) – which is also known as the signal-to-interference ratio (SIR) – is maximized, thereby achieving remarkable performance gains over the MF receiver [10]. The phase code design for the MMF receiver, however, is relatively few because of the lack of mathematical instruments.

Motivated by data-driven approaches such as deep learning (DL) [11]–[13], recent works [13], [14] revisit the phase-code design problem for PCR with the MMF receiver and discover good phase codes with high SCR. The authors in [14] proposed a deep reinforcement learning (DRL) approach, named AlphaSeq, to search for good phase codes. In AlphaSeq, the phase code design problem is described as a game and the SCR of the discovered phase code is defined as the reward. Over the course of DRL, the algorithm learns to discover better and better phase codes with increasingly higher SCR. On the other hand, HpGAN [13] utilizes the DL-based generative model, i.e., generative adversarial network (GAN), to generate new phase codes.

In this paper, we put forth a genetic algorithm (GA) approach, dubbed GASeq, to discover phase codes with low aperiodic autocorrelations for PCR with the MMF receiver. Compared with DL-based schemes, our approach discovers better phase code with higher SCR. Specifically, at a length of 59, the best phase code discovered by GASeq achieves an SCR of 50.84, while the best sequences found by AlphaSeq and HpGAN have an SCR of 33.45 and 45.16, respectively. In addition, our approach is much more efficient than DL-based approaches. Compare with AlphaSeq, for example, GASeq visits 166 times fewer states in order for the algorithm to converge and the discovered sequence is much better. Thanks to the high efficiency and scalability, we extend GASeq to discover longer phase codes. At a length of 100, the discover sequence achieves an SCR of 63.23.

The remainder of this paper is organized as follows. Section II describes the system model for PCR and introduces the MMF receiver. Section III details the proposed GA approach, i.e., GASeq. Section IV evaluates the performance of GASeq benchmarked against prior arts. Section V concludes this paper.

Related work on GA – Inspired by Charles Darwin's "natural selection", GA was first introduced by John Holland [15] in 1975 as a metaheuristic approach to solve optimization problems [16]. GA has been applied to a variety of disciplines [17], such as network routing protocol, image processing, data mining, neural networks, to name a few.

GA has also been used in radar systems. In [18], for example, the authors proposed a GA to divide the phase array of radar into the subarrays in order to reduce the hardware cost. Ref. [19] applied GA to a multiple-input and multiple-output (MIMO) radar to find the best locations for transmit and receive antenna arrays. There have also been studies in GA for PCR with the MF receiver. In [20], the authors enhanced the GA by a local search scheme and find phase codes with good peak sidelobe levels. In [21], the authors combined the global minimum convergence property of GA with the fast convergence rate of hamming scanning algorithm, and discovered phase codes with good discriminating factors.

II. System model

We consider a phase-coded pulse compression radar system, where the transmitted pulse is modulated by a sequence of binary codes $\boldsymbol{s} = \{s_n \in \{+1, -1\} : n = 0, 1, ..., N-1\}$, and $+1$ and -1 correspond to phases 0 and π, respectively. The received signal, on the other hand, is a sum of many echoes from various range bins with different amplitudes and delays. In particular, we denote by h_0 the radar cross section (RCS) of the range bin of interest. The received discrete-time model is given by

$$\boldsymbol{y} = h_0\boldsymbol{s} + \sum_{i=1-N, i\neq 0}^{N-1} h_i \boldsymbol{J}_i \boldsymbol{s} + \boldsymbol{w}, \tag{1}$$

where $\{h_i : i = 1-N, 2-N, ..., N-2, N-1, i \neq 0\}$ denotes the RCS of interfering range bins; $\boldsymbol{w}$ is the white Gaussian noise vector; $\boldsymbol{J}_i$, $\forall i$, denote $N \times N$ shift matrices that capture the propagation time needed for the clutters to reach the radar receiver. In particular, $\boldsymbol{J}_i$ can be written as

$$\boldsymbol{J}_i \triangleq \begin{array}{cc} & \begin{array}{ccccc} 1 & \cdots & i & \cdots & N \end{array} \\ \begin{array}{c} 1 \\ \\ \vdots \\ \\ N \end{array} & \begin{pmatrix} 0 & & 1 & \cdots & 0 \\ & 0 & & 1 & \\ \vdots & & 0 & & 1 \\ & & & 0 & \\ 0 & & & & 0 \end{pmatrix} \end{array}, \tag{2}$$

and $\boldsymbol{J}_i = \boldsymbol{J}_{-i}^\top$.

Given the received signal $\boldsymbol{y}$, our goal is to estimate h_0, the SCR of the range bin of interest. To this end, we cross-correlate the received sequence $\boldsymbol{y}$ by a real vector $\boldsymbol{x}$, yielding

$$\boldsymbol{x}^\top \boldsymbol{y} = h_0 \boldsymbol{x}^\top \boldsymbol{s} + \sum_{i=1-N, n\neq 0}^{N-1} h_i \boldsymbol{x}^\top \boldsymbol{J}_i \boldsymbol{s}, \tag{3}$$

where $\boldsymbol{w}$ is omitted since the noise term is often dominated by interference.

Note that the RCS of different range bins $\{h_i\}$ are unknown to the radar receiver. The sequence discovery problem in PCR is then discovering the optimal pair of sequences $(\boldsymbol{s}, \boldsymbol{x})$ such that the signal-to-clutter ratio (SCR) γ is maximized, where

$$\gamma = \frac{(\boldsymbol{x}^\top \boldsymbol{s})^2}{\sum_{i=1-N, i\neq 0}^{N-1} (\boldsymbol{x}^\top \boldsymbol{J}_i \boldsymbol{s})^2}. \tag{4}$$

Moreover, let $\boldsymbol{R} \triangleq \sum_{i=1-N, i\neq 0}^{N-1} \boldsymbol{J}_i \boldsymbol{s}\boldsymbol{s}^\top \boldsymbol{J}_i^\top$, γ can further be simplified as [10]:

$$\begin{aligned} \gamma &= \frac{(\boldsymbol{x}^\top \boldsymbol{s})^2}{\boldsymbol{x}^\top \boldsymbol{R} \boldsymbol{x}} = \frac{(\boldsymbol{x}^\top \boldsymbol{R}^{\frac{1}{2}} \boldsymbol{R}^{-\frac{1}{2}} \boldsymbol{s})^2}{\boldsymbol{x}^\top \boldsymbol{R} \boldsymbol{x}} \overset{(a)}{\leq} \frac{(\boldsymbol{x}^\top \boldsymbol{R} \boldsymbol{x})(\boldsymbol{s}^\top \boldsymbol{R}^{-1} \boldsymbol{s})}{\boldsymbol{x}^\top \boldsymbol{R} \boldsymbol{x}} \\ &= \boldsymbol{s}^\top \boldsymbol{R}^{-1} \boldsymbol{s}, \end{aligned} \tag{5}$$

where (a) follows from the Cauchy-Schwartz inequality. The equality holds when $\boldsymbol{R}^{\frac{1}{2}} \boldsymbol{x} = \boldsymbol{R}^{-\frac{1}{2}} \boldsymbol{s}$. Thus, for a given $\boldsymbol{s}$, the optimal $\boldsymbol{x}^*$ that maximizes γ is $\boldsymbol{x}^* = \boldsymbol{R}^{-1} \boldsymbol{s}$. The sequence discovery problem in SCR can be refined as

$$\boldsymbol{s}^* = \arg \max_{\boldsymbol{s} \in \{+1, -1\}^N} \boldsymbol{s}^\top \boldsymbol{R}^{-1} \boldsymbol{s}. \tag{6}$$

The combinatorial optimization problem in (6) is non-trivial to solve analytically. Thus, prior work often resorts to algorithmic solutions such as exhaustive search [10], heuristic search [6], and learning-based algorithms [13], [14]. In this paper, we put forth a GA-based solution to discover better phase codes $\boldsymbol{s}$ for SCR.

Remark: *As prior works [10], [13], [14], this paper does not consider fractional misalignment among echoes. If fractional misalignment is further considered, interested readers may refer to [22]–[24] for signal processing techniques to process the received signal. In particular, [23] deals with discrete source (e.g., binary phase codes), while [22], [24] deal with continuous source (e.g., polyphase codes).*

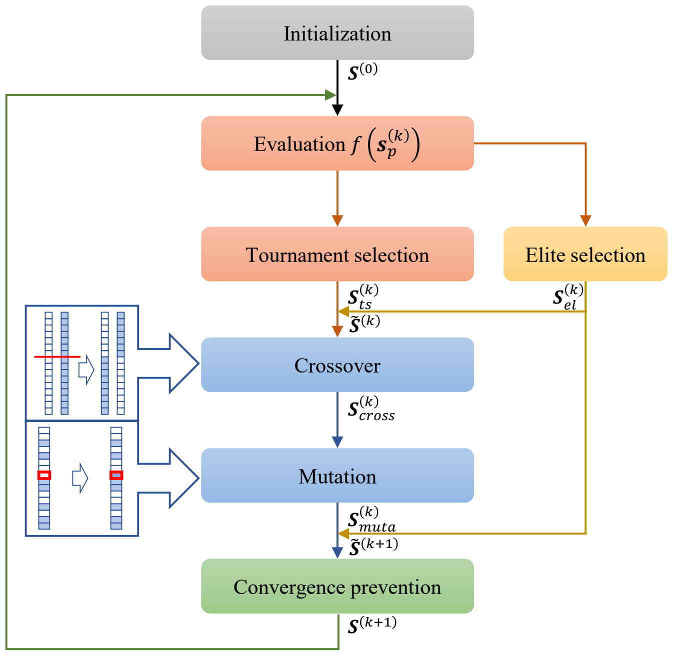

Fig. 2. The signal flow of GASeq for phase code discovery in PCR.

III. GASeq: A Genetic Algorithm Approach

GA algorithms solve combinatorial optimization problems by a natural selection process that mimics biological evolution: in the first generation, it generates a population of phase codes as candidates; in subsequent generations, it repeatedly improves the candidates based on the population of the last generation such that the candidates evolve towards the optimal solution. In this section, we detail our design of the GASeq for phase-code discovery.

To start with, we outline the signal flow of GASeq in Fig. 2.

Initialization. The initialization phase generates a population of phase codes as initial candidates. Let the population size be P, the initialized population can be written as an $N \times P$ matrix:

$$\boldsymbol{S}^{(0)} = \left[\boldsymbol{s}_1^{(0)}, \boldsymbol{s}_2^{(0)}, \cdots, \boldsymbol{s}_P^{(0)}\right], \tag{7}$$

where each column $\boldsymbol{s}_p^{(k)} \in \{+1,-1\}^N$ is a phase code of length N; the superscript $k = 0, 1, 2, ..., K$ denotes the index of generation (there are $K+1$ generations and the initialized population is the 0-th generation); and the subscript $p = 1, 2, ..., P$ denotes the index of the phase code in the population.

In general, the elements of $\boldsymbol{S}^{(0)}$ are sampled uniformly from $\{+1,-1\}$. Another idea is incorporating existing phase codes discovered by AlphaSeq [14] and HpGAN [13] into $\boldsymbol{S}^{(0)}$ to generate more promising offspring and speed up the convergence. We will compare different initialization schemes in Section IV.

Evaluation. The evaluation process computes the "fitness" of individual phase code in the current population, where fitness reflects how good a phase code is in terms of (6), and can be computed by $f(\boldsymbol{s}_p^{(k)}) \triangleq (\boldsymbol{s}_p^{(k)})^\top \boldsymbol{R}^{-1} \boldsymbol{s}_p^{(k)}$, $\forall \boldsymbol{s}_p^{(k)} \in \boldsymbol{S}^{(k)}$.

Based on the current population $\boldsymbol{S}^{(k)}$ and the evaluated fitness, we next determine the next population $\boldsymbol{S}^{(k+1)}$.

Elite selection. To goal of selection is to retain superior candidates and eliminate inferior candidates from the last population according to their fitness. In order to retain the best candidates of $\boldsymbol{S}^{(k)}$, we first perform elite selection to identify E phases codes in $\boldsymbol{S}^{(k)}$ with the largest fitness. Let us denote the elite codes by $\boldsymbol{S}_{el}^{(k)} \in \mathcal{R}^{N\times E}$, which is a submatrix of $\boldsymbol{S}^{(k)}$. The elite codes will be directly assigned to $\boldsymbol{S}^{(k+1)}$, while the other $P-E$ vacancies in $\boldsymbol{S}^{(k+1)}$ are determined by tournament selection, as detailed below.

Tournament selection. In this operation, we perform $P-E$ tournaments [25] to select $P-E$ candidates to form a new matrix $\boldsymbol{S}_{ts}^{(k)} \subset \boldsymbol{S}^{(k)}$. In each tournament, we randomly select M phase codes without replacement from $\boldsymbol{S}^{(k)}$ (including the elites) and compare their fitnesses. The winner, i.e., the phase code with the maximal fitness among the chosen M phase codes, goes to $\boldsymbol{S}_{ts}^{(k)}$.

For the phase code with the i-th largest fitness in $\boldsymbol{S}^{(k)}$, the probability that it wins a tournament is

$$p_i = \frac{\binom{P-i}{M-1}}{\binom{P}{M}} = \frac{M\cdot(P-i)!(P-M)!}{P!(P-i-M+1)!}. \tag{8}$$

Therefore, the probability that it goes to $\boldsymbol{S}_{ts}^{(k)}$ after tournament selection is $1-(1-p_i)^{P-E}$.

Crossover and mutation. Crossover refers to the operations of replacing and recombining the parts of a pair of candidates (i.e., parents) to generate a new candidate (i.e., child). To start with, we combine the tournament-selected phase codes $\boldsymbol{S}_{ts}^{(k)}$ with the elites $\boldsymbol{S}_{el}^{(k)}$, yielding $\widetilde{\boldsymbol{S}}^{(k)} = \left[\boldsymbol{S}_{ts}^{(k)}, \boldsymbol{S}_{el}^{(k)}\right]$.

With crossover, we first randomly select two phase codes $\boldsymbol{s}_i^{(k)}$, $\boldsymbol{s}_j^{(k)}$, $i, j \in \{1, 2, \cdots, P\}$ from $\widetilde{\boldsymbol{S}}^{(k)}$. Then, we split both $\boldsymbol{s}_i^{(k)}$ and $\boldsymbol{s}_j^{(k)}$ into two parts at a random point and form a new phase code by concatenating the first part of $\boldsymbol{s}_i^{(k)}$ and the second part of $\boldsymbol{s}_j^{(k)}$. For example, if $\boldsymbol{s}_i^{(k)} = [1,-1,1,-1,1,-1,1]^\top$, $\boldsymbol{s}_j^{(k)} = [-1,-1,-1,1,1,1,1]^\top$, and we split after the third symbol, the newly constructed phase code is given by $\boldsymbol{s}_{\text{new}}^{(k)} = [1,-1,1,1,1,1,1]^\top$. The crossover operation will be performed $P-E$ times, after which we obtain $P-E$ new phase codes, forming a new set $\boldsymbol{S}_{\text{cross}}^{(k)} \in \mathcal{R}^{N\times(P-E)}$.

Next, we randomly mutate the elements of $\boldsymbol{S}_{\text{cross}}^{(k)}$. The purpose is to explore the search space and introduce more diversity to the algorithm. To be more specific, for each phase code in $\boldsymbol{S}_{\text{cross}}^{(k)}$, we mutate one element (from -1 to 1 or from 1 to -1) with probability p_{muta} ($0 \le p_{\text{muta}} \le 1$). Denoting by $\boldsymbol{S}_{\text{muta}}^{(k)}$ the mutated $\boldsymbol{S}_{\text{cross}}^{(k)}$, we concatenate it with the elite codes of $\boldsymbol{S}^{(k)}$, yielding $\widetilde{\boldsymbol{S}}^{(k+1)} = \left[\boldsymbol{S}_{\text{muta}}^{(k)}, \boldsymbol{S}_{\text{el}}^{(k)}\right]$.

TABLE I
HYPERPARAMETERS OF GASEQ FOR PHASE CODE DISCOVERY

	Parameters	Definitions
GA	$N = 59$	Length of the binary phase code
	$K = 200$	Number of generations in GA
	$P = 10000$	Number of candidates in the population
	$E = 2000$	Number of elites
	$M = 5$	Number of competitors in a tournament selection
	$p_{\text{muta}} = 0.3$	Mutation rate
	$p_{\text{conv}} = 0.7$	Probability for convergence prevention

Early-convergence prevention. The last step of one generation is early-convergence prevention. Notice that in the above operations, a phase code in $\boldsymbol{S}^{(k)}$ can appear in $\widetilde{\boldsymbol{S}}^{(k+1)}$ multiple times, and the average number of appearance is proportional to the fitness of the phase code. To prevent the algorithm from being dominated by these "better" phase codes and early stopping, we have to reduce the number of repeating phase codes in the new generation.

The early-convergence prevention operates in the following manner. Suppose that a phase code $\boldsymbol{s}$ appears in $\widetilde{\boldsymbol{S}}^{(k+1)}$ for G times. For each appearance, we will decide independently whether to keep or drop it. First, the first appearance is kept to ensure that $\boldsymbol{s}$ appears at least once in $\boldsymbol{S}^{(k+1)}$. Then, the rest appearances will be kept with probability p_{conv}, where $0 \leq p_{\text{conv}} \leq 1$, and dropped from $\widetilde{\boldsymbol{S}}^{(k+1)}$ otherwise. After the above operations, we obtain a new population $\widetilde{\boldsymbol{S}}^{(k+1)}_{\text{left}}$ with the number of phase codes less than P.

Finally, the $(k+1)$-th generation of phase codes $\boldsymbol{S}^{(k+1)}$ is obtained by padding randomly initialized phase codes to $\widetilde{\boldsymbol{S}}^{(k+1)}_{\text{left}}$ such that $\boldsymbol{S}^{(k+1)} \in \mathcal{R}^{N\times P}$.

IV. NUMERICAL EXPERIMENTS

Given the GASeq algorithm described in Section III, this section performs numeral experiments to discover new phase codes for pulse compression radar with the MMF receiver.

A. Experimental setup and baselines

We adopt the same system setup as [13], [14] and the goal is to discover a phase code of length $N = 59$ that maximizes the SCR γ. The default hyper-parameter setting of our GASeq is summarized in Table I. Specifically, the algorithm operates for K generations and the population size is $P = 10,000$. The elite population size is set to $E = 2000$ and the number of competitors in each tournament is $M = 5$. The mutation rate is $p_{\text{muta}} = 0.3$ and the probability for convergence prevention is $p_{\text{conv}} = 0.7$.

There are three baselines: the Legendre sequence [5], AlphaSeq [14], and HpGAN [13]. Let $N = 59$, the Legendre sequence is given by

$$\boldsymbol{s}_{\text{L}} = \begin{bmatrix} +1 & +1 & -1 & +1 & +1 & +1 & -1 & +1 & -1 & +1 \\ -1 & -1 & +1 & -1 & -1 & +1 & +1 & +1 & -1 & +1 \\ +1 & +1 & +1 & -1 & -1 & +1 & +1 & +1 & +1 & +1 \\ -1 & -1 & -1 & -1 & -1 & +1 & +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 & -1 & +1 & +1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 & -1 & -1 & -1 & +1 & -1 & \end{bmatrix},$$

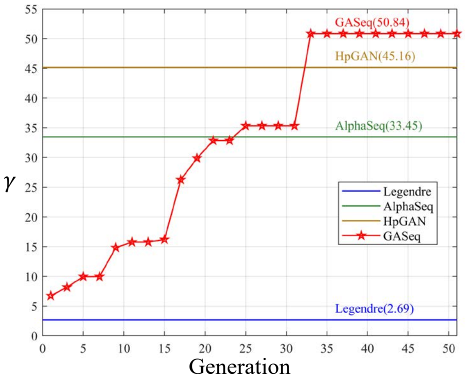

Fig. 3. GASeq to discover a phase-coded sequence for pulse compression radar versus $\boldsymbol{s}_{\text{L}}$, $\boldsymbol{s}_{\text{alpha}}$, $\boldsymbol{s}_{\text{HpGAN}}$.

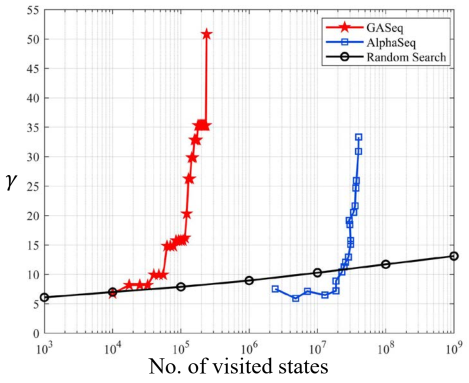

Fig. 4. Searching capability comparison of GA, AlphaSeq, and random search.

and $\gamma(\boldsymbol{s}_{\text{L}}) \approx 2.69$; AlphaSeq is searched by DRL, giving

$$\mathbf{s}_{\text{alpha}} = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & +1 & +1 & +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & +1 & -1 & +1 & -1 & -1 & +1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 & -1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & \end{bmatrix},$$

and $\gamma(\boldsymbol{s}_{\text{alpha}}) \approx 33.45$; HpGAN is searched by GAN, giving

$$\mathbf{s}_{\text{HpGAN}} = \begin{bmatrix} -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & +1 & -1 & +1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 & +1 & +1 \\ -1 & +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 & +1 \\ +1 & +1 & +1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & \end{bmatrix},$$

and $\gamma(\boldsymbol{s}_{\text{HpGAN}}) \approx 45.16$.

B. Main results

To ease reading, we first summarize our main results in this subsection. Given the hyperparameters in Table I, Fig. 3 presents the improvement of γ over the course of evolution. As can be seen, after 32 generations, GASeq converges to a phase code

$$\boldsymbol{s}_{\text{GA}} = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & +1 & +1 & +1 & +1 & -1 & -1 & -1 \\ -1 & -1 & +1 & +1 & +1 & -1 & -1 & +1 & +1 & +1 \\ -1 & +1 & +1 & -1 & -1 & +1 & -1 & -1 & +1 & -1 \\ +1 & -1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & \end{bmatrix},$$

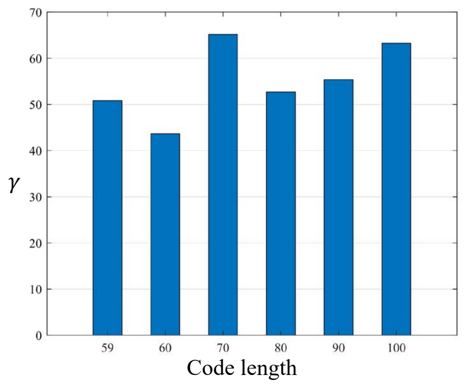

Fig. 5. Search for longer phase code using GASeq: the achieved SCR versus code length.

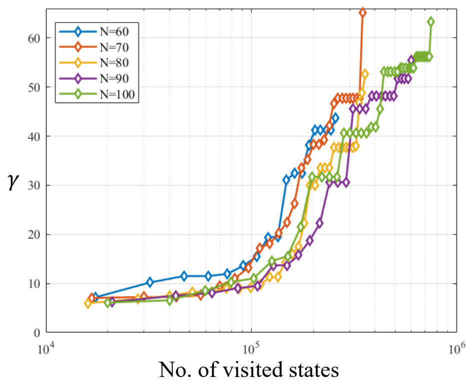

Fig. 6. Search for longer phase code using GASeq: the achieved SCR versus the number of visited states.

the SCR of which is $\gamma(\boldsymbol{s}_{\mathrm{GA}}) = 50.84$. Compared with $\boldsymbol{s}_{\mathrm{L}}$, $\boldsymbol{s}_{\mathrm{alpha}}$, and $\boldsymbol{s}_{\mathrm{HpGAN}}$, $\boldsymbol{s}_{\mathrm{GA}}$ improves the SCR by 48.15, 17.39 and 5.68, respectively.

In addition, GASeq is a much more efficient search algorithm than prior arts. Fig. 4 compares the evolution of γ as a function of the number of visited states for GA, DRL, and random search. As shown, GA discovers $\boldsymbol{s}_{\mathrm{GA}}$ after visiting only 2.4×10^5 states. At such a small number of visited states, the SCRs of the sequences discovered by AlphaSeq and random search are less than 10.

Thanks to the efficiency and scalability of GASeq, we further apply it to discover longer phase codes for $N \in [60, 100]$ and present the achieved SCR in Fig. 5. At a code length of $N = 100$, for example, the best discovered sequence has an SCR of 63.23. We emphasize that there is no proportional relationship between SCR and the code length – as can be seen from (4), with the increase in N, both the signal and interference power increases and the ratio is undetermined.

With different code length, the number of visited states of GASeq is shown in Fig. 6. When $N = 100$, GA discovers $\boldsymbol{s}_{\mathrm{GA}}$ with an SCR of 63.23 after visiting 7.5×10^5 states.

C. Impact of initialization, M, and E

There are a number of parameters to be determined in GASeq. In this section, we study the impact of these parameters and show how the default parameter settings are chosen.

In the initialization phase, the first generation of candidates is randomly generated. In addition to that, we can insert existing phase codes in the literature as seeds to generate offspring. Fig. 7 compares the evolution of γ with different seeds.

As can be seen, adding existing phase codes is actually harmful to the evolution of GASeq. This is not surprising as these “better” phase codes dominate the generation, and hence, GASeq explores less compared with the case of pure random initialization.

The second important parameter is the number of individuals being selected in each tournament, i.e., M, because it

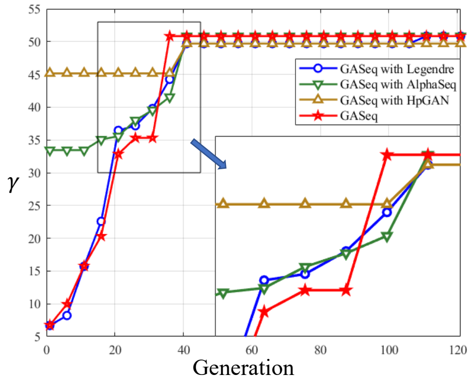

Fig. 7. Impact of different initialization schemes on GASeq.

directly determines the probability that each candidate goes to the next generation, according to (8). With different M, Fig. 8 presents the performance of GASeq.

As shown, a large M often leads to suboptimal performance in convergence. This is because the candidates with larger fitness are more likely to go to the next generation if M is large, and hence, the diversity of the population in the next generation decreases. On the other hand, when M is small, e.g., $M = 2$, GASeq can often converge to the same optimum, but the convergence speed can be slow. Overall, setting $M = 5$ is a good choice for GASeq.

The final parameter is E, the size of the elite set. Succinctly speaking, the setting of E has a similar effect on the performance of GASeq as M: a small elite size encourages exploration while a large elite size encourages exploitation. As shown in Fig. 9, $E = 2000$ is a proper setting as it leads to the optimal phase code while ensuring fast convergence.

V. Conclusion

Designing phase codes with low aperiodic autocorrelations is an important problem in pulse compression radar (PCR) for sidelobe suppression, but is non-trivial due to the lack of mathematical instruments, especially when used in conjunction

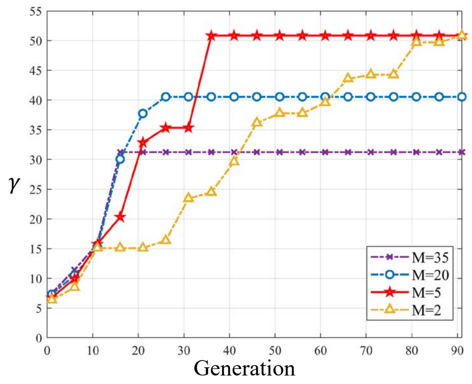

Fig. 8. Impact of the number of individuals being selected in each tournament, M, on the performance of GASeq.

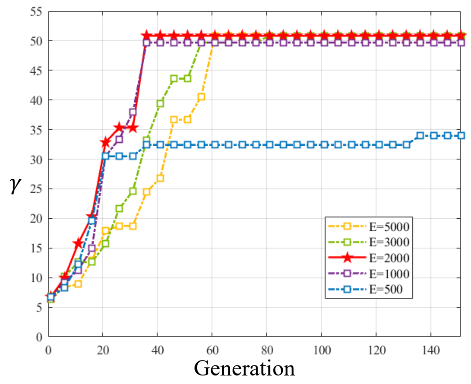

Fig. 9. Impact of the elite size E on the performance of GASeq.

with the mismatched filter (MMF) receiver to maximize the signal-to-clutter ratio (SCR). To meet the challenge, this paper put forth a genetic algorithm (GA) approach. Our specific contributions and results are summarized as follows.

We developed GASeq, a GA for phase code discovery in PCR with the MMF receiver. GASeq exhibits three main advantages over existing deep learning (DL)-based phase codes: 1) Superiority. GASeq discovered better phase codes with higher SCR than the state of the art. 2) Efficiency. GASeq converges much faster and visits significantly fewer states than DL-based approaches to find a sequence with the same SCR. 3) Scalability. GASeq can be readily extended to search longer phase codes, which thwarts DL-based schemes due to the high complexity.

Moving forward, a straightforward extension of GASeq is discovering non-binary (such as polyphase) phase codes to further improve the estimation performance. To that end, an enhancement on the GA is needed as the phase codes can have continuous, as opposed to discrete, phase or amplitude. A promising candidate for continuous-domain optimization is the differential evolution algorithm and its variants.

References

[1] E. C. Farnett, G. H. Stevens, and M. Skolnik, "Pulse compression radar," *Radar handbook*, vol. 2, pp. 10–11, 1990.

[2] M. Golay, "Sieves for low autocorrelation binary sequences," *IEEE Trans. Inf. Theory*, vol. 23, no. 1, pp. 43–51, 1977.

[3] D. R. Wehner, "High resolution radar," *Norwood*, 1987.

[4] J. Tsao and B. D. Steinberg, "Reduction of sidelobe and speckle artifacts in microwave imaging: The CLEAN technique," *IEEE Trans. Antennas and Prop.*, vol. 36, no. 4, pp. 543–556, 1988.

[5] M. Golay, "The merit factor of Legendre sequences (corresp.)," *IEEE Trans. Inf. Theory*, vol. 29, no. 6, pp. 934–936, 1983.

[6] J. Brest and B. Bošković, "A heuristic algorithm for a low autocorrelation binary sequence problem with odd length and high merit factor," *IEEE Access*, vol. 6, pp. 4127–4134, 2018.

[7] J. Jedwab, "A survey of the merit factor problem for binary sequences," in *Int. Conf. Sequences and Their Appl.*, 2004, pp. 30–55.

[8] H. Rohling, "Mismatched filter design for pulse compression," in *IEEE Int. Conf. Radar*, 1990.

[9] B. Zrnic, A. Zejak, A. Petrovic, and I. Simic, "Range sidelobe suppression for pulse compression radars utilizing modified RLS algorithm," in *IEEE Int. Symp. Spread Spectrum Tech. Appl.*, vol. 3, 1998, pp. 1008–1011.

[10] P. Stoica, J. Li, and M. Xue, "On binary probing signals and instrumental variables receivers for radar," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3820–3825, 2008.

[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[12] Y. Shao, A. Rezaee, S. C. Liew, and V. W. S. Chan, "Significant sampling for shortest path routing: A deep reinforcement learning solution," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2234–2248, 2020.

[13] M. Zhang, Z. Zhou, L. Li, Z. Liu, M. Yang, and Y. Feng, "HpGAN: Sequence search with generative adversarial networks," *IEEE Trans. Neural Netw. Learning Syst.*, 2021.

[14] Y. Shao, S. C. Liew, and T. Wang, "AlphaSeq: Sequence discovery with deep reinforcement learning," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 31, no. 9, pp. 3319–3333, 2019.

[15] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[16] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091–8126, 2021.

[17] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm – a literature review," in *IEEE Int. conf. machine learning, big data, cloud and parallel computing*, 2019, pp. 380–384.

[18] D. Wang, H. Hu, and Z. Yang, "Improved genetic algorithm for the configuration optimization of the subarrays in phased array radar," in *IEEE Int. Congress Image and Signal Proc., BioMedical Engineering and Inf.* IEEE, 2016, pp. 930–934.

[19] C. Vasanelli, R. Batra, and C. Waldschmidt, "Optimization of a MIMO radar antenna system for automotive applications," in *IEEE European Conf. Antennas and Prop.*, 2017, pp. 1113–1117.

[20] M. A. Nasrabadi and M. H. Bastani, "A new approach for long low autocorrelation binary sequence problem using genetic algorithm," in *IEEE Int. Conf. Radar*, 2006.

[21] E. P. C. Rao, G. Bommagani, and S. Singh, "Phase coded sequences design for pulse compression radar," *Helix*, vol. 12, no. 1, pp. 10–17, 2022.

[22] Y. Shao, D. Gündüz, and S. C. Liew, "Federated learning with misaligned over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 3951–3964, 2021.

[23] Y. Shao, S. C. Liew, and L. Lu, "Asynchronous physical-layer network coding: symbol misalignment estimation and its effect on decoding," *IEEE Trans. Wireless Commun.*, vol. 16, no. 10, pp. 6881–6894, 2017.

[24] Y. Shao, D. Gündüz, and S. C. Liew, "Bayesian over-the-air computation," *arXiv:2109.03780*, 2022.

[25] B. L. Miller, D. E. Goldberg *et al.*, "Genetic algorithms, tournament selection, and the effects of noise," *Complex systems*, vol. 9, no. 3, pp. 193–212, 1995.