

# A Comparison of Antenna Placement Algorithms

Abhinav Jauhri

March 30, 2015

# Contributions

---

- ▶ Formulation of the antenna placement problem
- ▶ Evaluation of standard stochastic algorithms on a real-world problem
- ▶ Able to achieve global optimum with as low as **25% evaluations** of search space

# Outline of this talk

---

- ▶ Part 1: Introduction to the antenna placement problem
- ▶ Part 2: Description of stochastic algorithms, and formulation of an instance of antenna placement problem
- ▶ Part 3: Results of real world experiments

# Part 1: Introduction to the antenna placement problem

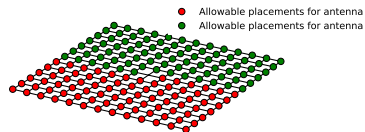
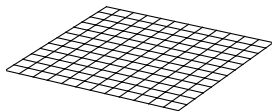
# Antenna Placement Problem

---

Given, platform

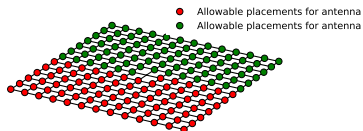
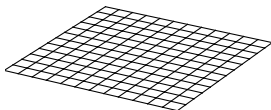
+

allowable placements of antennas

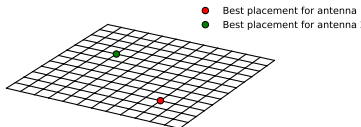


# Antenna Placement Problem

Given, platform + allowable placements of antennas



**Problem:** find best antenna placements



# Antenna Placement Problem

---

Given:

- ▶ platform  $P$  with its surface gridded such that end points represent possible antenna placements
- ▶ set of  $m$  antennas  $A = A_1, A_2, \dots, A_m$  such that  $m > 1$
- ▶ for each  $A_i$ ,  $L_i$  denote the set of allowable placements  $\in \mathbb{R}^3$  such that  $|L_i| = n_i$  and  $\forall i, n_i > 1$ ;  
$$L_i = \{(x_1, y_1, z_1) \dots (x_{n_i}, y_{n_i}, z_{n_i})\}$$

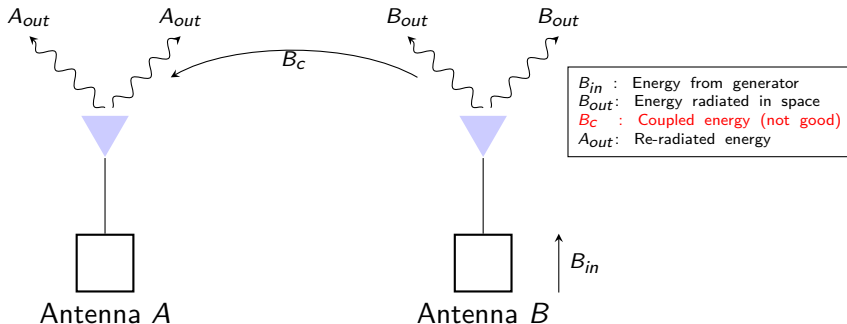
**Problem:** Find a set of  $n$  optimal antenna placements on  $P$

Question: How is a good antenna placement quantified in the context of platform and other antennas?



# Mutual Coupling

When two antennas are in proximity, and one is transmitting, the second will receive some of the transmitted energy.



# Minimize Mutual Coupling

---

$$F_{MC} = \sum_{i=1}^{n-1} \sum_{j=i+1}^n CP(A_i, A_j), \quad (1)$$

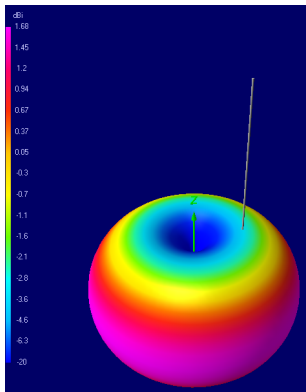
where

- ▶  $CP(\cdot, \cdot) \in \mathbb{R}$  is the coupling between two antennas, and computed using a simulator

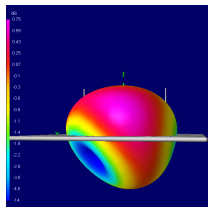
*Example:* If  $n = 3$ , then  $F_{MC} = CP(A_1, A_2) + CP(A_1, A_3) + CP(A_2, A_3)$

# Radiation Pattern

Free-space pattern without platform or other antennas

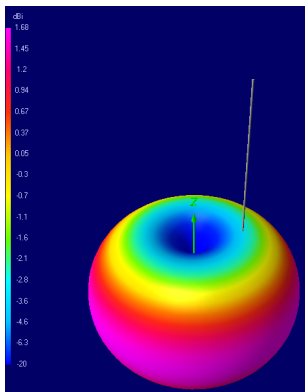


Pattern with platform and antennas (in-situ) similar

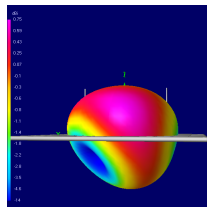


# Radiation Pattern

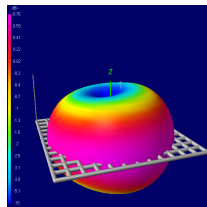
Free-space pattern without platform or other antennas



Pattern with platform and antennas (in-situ) similar



Pattern with platform and antennas (in-situ) similar to free-space pattern



# Minimize Difference in Radiation Pattern

---

$$F_{RP} = \sum_{i=1}^n \sum_{\theta=0}^{\pi} \sum_{\phi=0}^{2\pi} (FSG_i(\theta, \phi) - ISG_i(\theta, \phi))^2, \quad (2)$$

where

- ▶  $\theta, \phi$  spherical coordinates
- ▶  $FSG(\cdot)$  returns free-space gain pattern
- ▶  $ISG(\cdot)$  returns in-situ gain pattern

# Objective Function

---

Find a placement such that  $F$  is minimal:

$$F = \alpha F_{MC} + \beta F_{RP}, \quad (3)$$

where  $\alpha, \beta$  are adjustable weights for each of the objectives

## Part 2: Stochastic Algorithms

# Stochastic Algorithms

---

We will consider algorithms which rely on *randomization* principle.

- ▶ Genetic Algorithm
- ▶ Evolutionary Strategy
- ▶ Simulated Annealing
- ▶ Hill Climbing

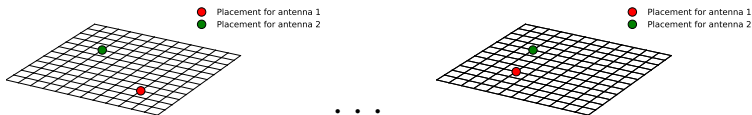
Each algorithm maintains a candidate solution or pool of candidate solutions called population



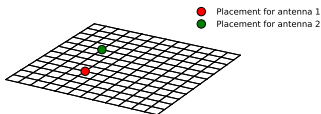
# Candidate Solution

Also referred as an **individual**

- Pool of individuals

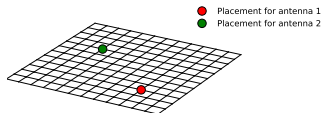


- Single individual

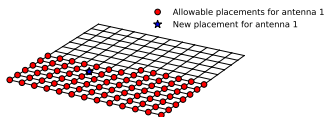


# Stochastic Algorithms: Mutation Operator

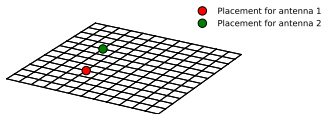
1. Given an individual, select an antenna uniformly at random, let's say antenna 1:



2. For antenna 1, select any other placement:

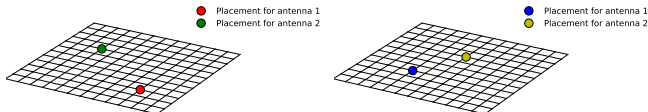


3. Change position for antenna 1 in individual:

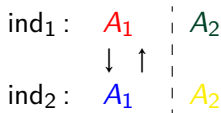


# Stochastic Algorithms: Crossover Operator

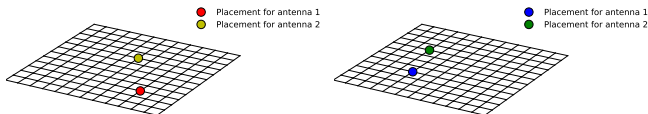
1. Select two individuals from population:



2. Select a crossover point, and swap placements prior to the point:

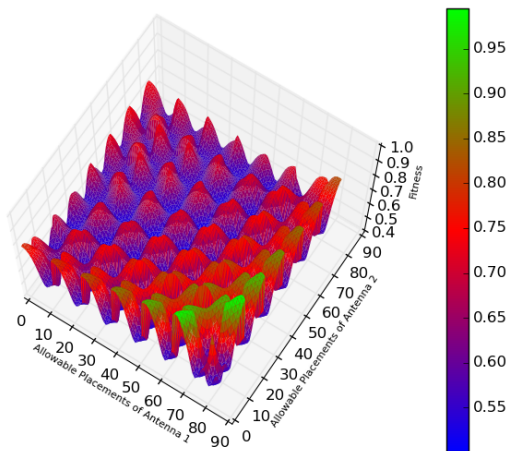


3. Two new offsprings created:



Question: Why use stochastic algorithms?

# Multi-Modal Search Space



# Genetic Algorithm

```

1  P ← generate  $p$  random individuals. Compute
    $fitness(ind_i), i \in [1, p]$ ;
2   $i = 0$  ;
3  while  $i < gen_{max}$  do
4      Elitism: Select  $n_e$  fittest individuals to add to P' ;
5      for  $(p - n_e)/2$  times do
6          /* select returns pair of individuals */
7          M ←  $select(P, 2)$  ;
8          if  $rand(0, 1) < p_c$  then
9              Apply  $crossover(M)$  to get two offsprings
10             O ;
11             Add  $O$  to  $P'$  ;
12         else
13             Add  $M$  to  $P'$  ;
14     Uniformly select  $p_m \cdot (p - n_e)$  individuals from  $P$ ,
15     and apply mutation operator to each ;
16     Update  $P \leftarrow P'$  ;
17     Compute  $fitness(ind_i), i \in [1, p]$ ;
18     Update  $i \leftarrow i + 1$  ;

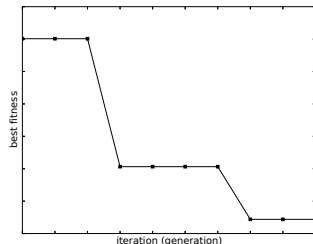
```

# Genetic Algorithm

```

1   $P \leftarrow$  generate  $p$  random individuals. Compute
    $fitness(ind_i), i \in [1, p]$ ;
2   $i = 0$  ;
3  while  $i < gen_{max}$  do
4      ;
5      Elitism: Select  $n_e$  fittest individuals to add to  $P'$  ;
6      for  $(p - n_e)/2$  times do
7          /* select returns pair of individuals
8             */
9              $M \leftarrow select(P, 2)$  ;
10            if  $rand(0, 1) < p_c$  then
11                Apply  $crossover(M)$  to get two offsprings
12                 $O$  ;
13                Add  $O$  to  $P'$  ;
14            else
15                Add  $M$  to  $P'$  ;
16
17            Uniformly select  $p_m \cdot (p - n_e)$  individuals from  $P$ ,
18            and apply  $mutation$  operator to each ;
19            Update  $P \leftarrow P'$  ;
20            Compute  $fitness(ind_i), i \in [1, p]$ ;
21            Update  $i \leftarrow i + 1$  ;

```



Plateaus suggesting  
stagnation of search

# Evolutionary Strategy

---

---

```
1 P ← generate  $\mu$  random individuals ;
2  $i = 0$  ;
3 while  $i < gen_{max}$  do
4   Create  $\lambda/\mu$  offsprings from each  $\mu$  individuals by
   applying mutation operator, and add all offsprings
   to P;
5   Compute  $fitness(ind_i), i \in [1, \lambda]$  ;
6   Keep  $\mu$  best individuals in P, and discard remaining
    $\lambda - \mu$  individuals ;
7   Update  $i \leftarrow i + 1$ 
```

---



# Evolutionary Strategy

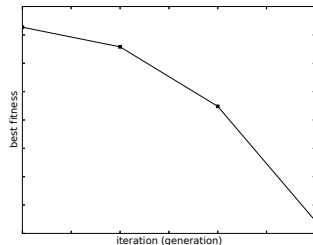
---

```

1  $\mathbf{P} \leftarrow$  generate  $\mu$  random individuals ;
2  $i = 0$  ;
3 while  $i < gen_{max}$  do
4     Create  $\lambda/\mu$  offsprings from each  $\mu$  individuals by
      applying mutation operator, and add all offsprings
      to  $\mathbf{P}$  ;
5     Compute  $fitness(ind_i), i \in [1, \lambda]$  ;
6     Keep  $\mu$  best individuals in  $\mathbf{P}$ , and discard remaining
       $\lambda - \mu$  individuals ;
7     Update  $i \leftarrow i + 1$ 

```

---



No stagnation in  
exploration of search  
space

# Simulated Annealing

---

```
1 c ← generate a random individual ;  
2  $i = 0$  ;  
3 while  $i < i_{max}$  do  
4   n ← mutate(c) ;  
5   if fitness(c) < fitness(n) then  
6     if rand(0,1) <  $e^{-\delta f / T}$  then  
7       c ← n  
8   else  
9     c ← n ;  
10   $T \leftarrow T \cdot f_{cooling}$  ;  
11   $i \leftarrow i + 1$  ;
```

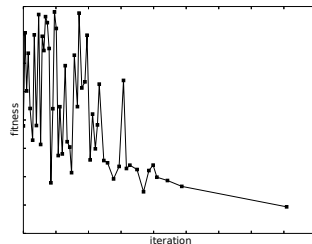
---

# Simulated Annealing

```

1   $c \leftarrow$  generate a random individual ;
2   $i = 0$  ;
3  while  $i < i_{max}$  do
4       $n \leftarrow mutate(c)$  ;
5      if  $fitness(c) < fitness(n)$  then
6          if  $rand(0,1) < e^{-\delta f / T}$  then
7               $c \leftarrow n$ 
8      else
9           $c \leftarrow n$  ;
10      $T \leftarrow T \cdot f_{cooling}$  ;
11      $i \leftarrow i + 1$  ;

```



Fluctuation in fitness  
gradually reduces due to  
cooling

# Hill Climbing

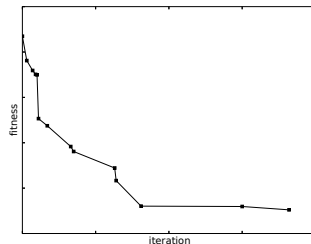
---

```
1 Initialize  $c \leftarrow$  generate a random individual ;
2 Compute  $fitness(c)$  ;
3  $i = 0$  ;
4 while  $i < i_{max}$  do
5      $n \leftarrow mutate(c)$  ;
6     if  $fitness(n) < fitness(c)$  then
7          $c \leftarrow n$ 
8      $i \leftarrow i + 1$ 
```

---

# Hill Climbing

```
1 Initialize  $c \leftarrow$  generate a random individual ;  
2 Compute  $fitness(c)$  ;  
3  $i = 0$  ;  
4 while  $i < i_{max}$  do  
5    $n \leftarrow mutate(c)$  ;  
6   if  $fitness(n) < fitness(c)$  then  
7      $c \leftarrow n$   
8    $i \leftarrow i + 1$ 
```



Only accepts fitter  
individual

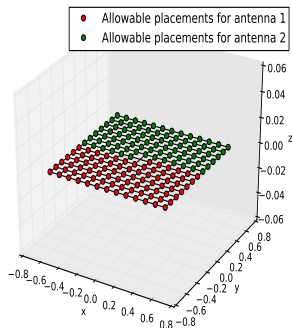
## Part 3: Results of real world experiments

# Experimental Setup

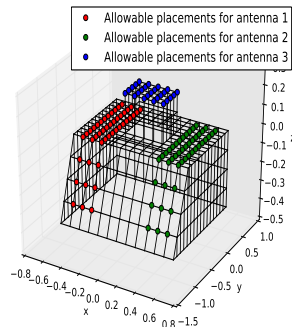
---

1. All test cases describe platforms which are replicas of real-world use cases like mobile devices, tanks, and cars
2. We use a popular *NEC2* simulator<sup>1</sup> to get fitness parameters
3. Evaluate the entire search space using an exhaustive algorithm to find the optimal antenna locations

# Experiments: Test Cases



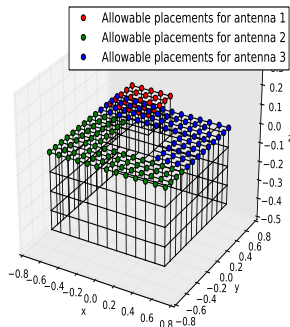
Test Case #1 with 7056( $84 \times 84$ ) allowable placements



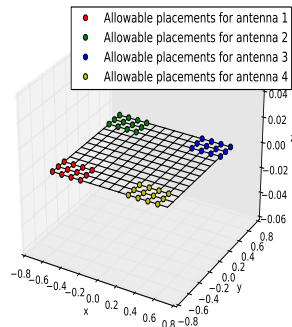
Test Case #2 with 50625( $45 \times 45 \times 45$ ) allowable placements



# Experiments: Test Cases

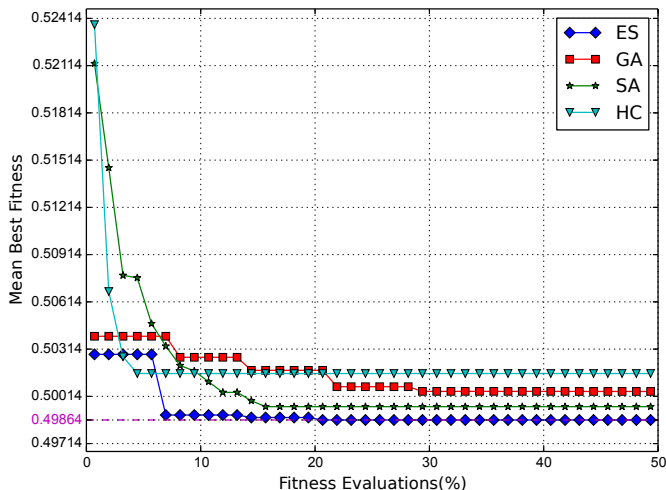


Test Case #3 with 126025( $71 \times 71 \times 25$ ) allowable placements

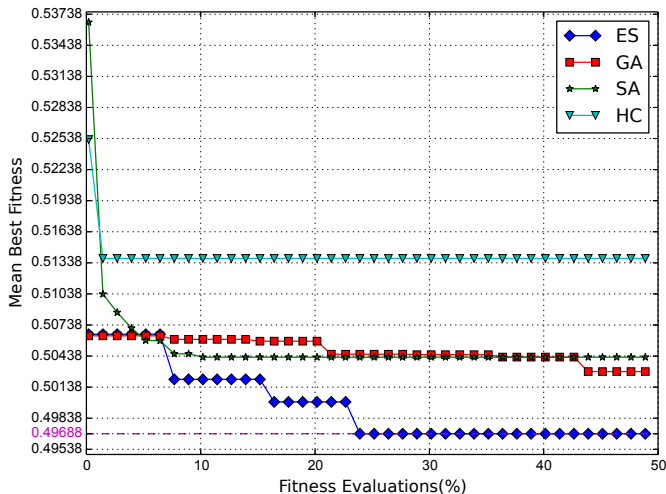


Test Case #4 with 20736( $12 \times 12 \times 12 \times 12$ ) allowable placements

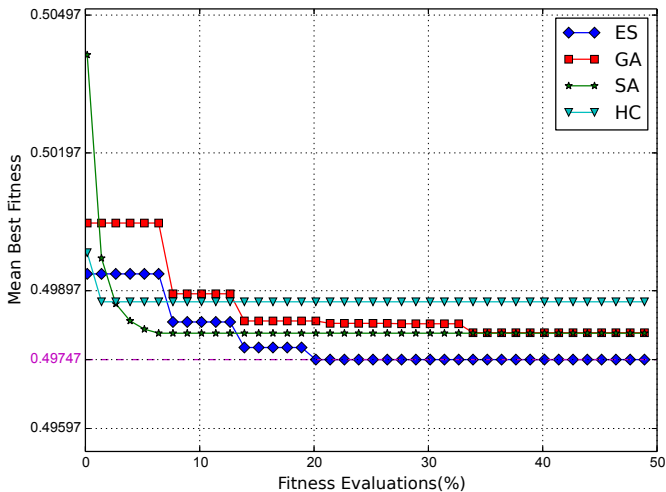
# Results - Test Case 1



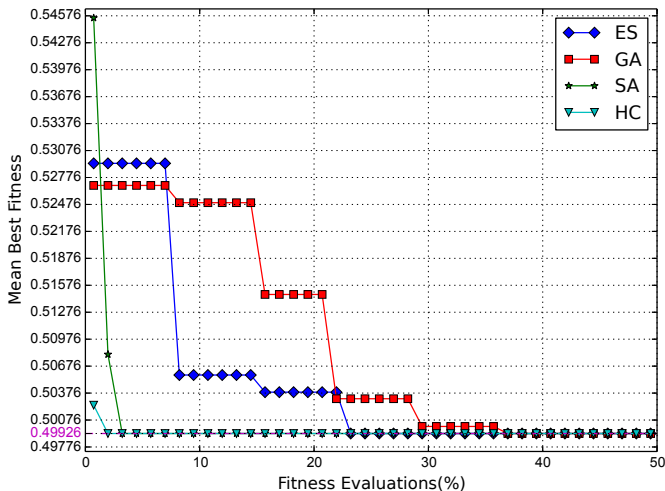
# Results - Test Case 2



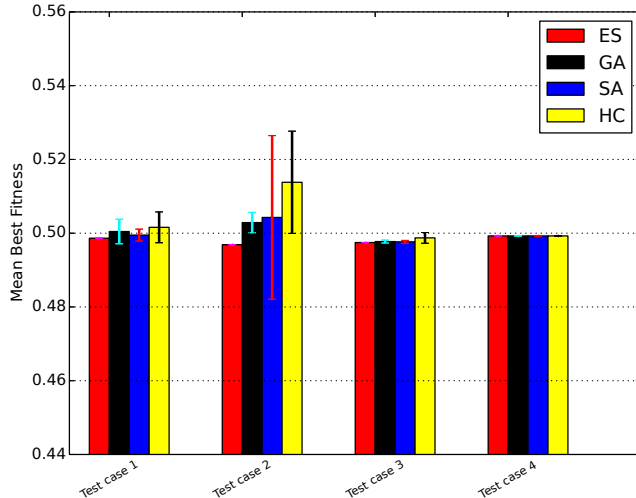
# Results - Test Case 3



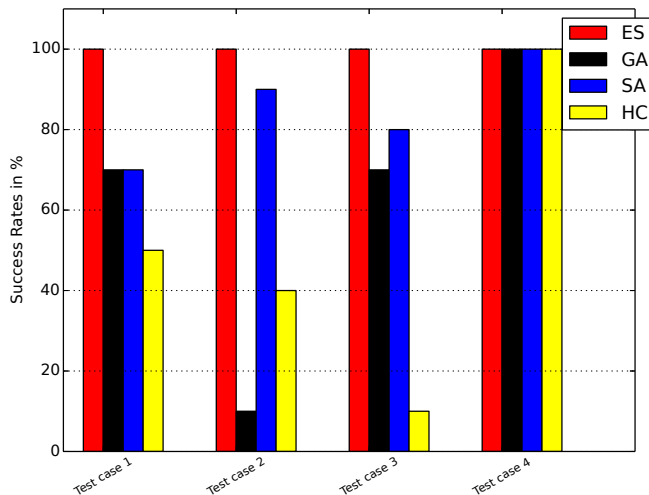
# Results - Test Case 4



# Results - Mean Best Fitness With Std. Dev.



# Results - Success Rates



# Conclusion

---

- ▶ Formulation of the antenna placement problem
- ▶ Generic problem formulation to accommodate multiple antennas and platforms
- ▶ Optimal placements found using stochastic algorithms