1.Probability of detection function

1.Probablity of False Positive

In [11]:
```
import math
SNR=1
n=10
False_pro=math.exp(-SNR*math.sqrt(n))
print('False_pro is {}'.format(False_pro))
```

```
False_pro is 0.04232921962320499
```

2. Noise Floor

2.Find noise floor from data

In [12]:
```
from scipy import stats
import statistics
from statistics import median
from scipy.signal import find_peaks

NF=statistics.median([n_obs_main_trace_env])
NFdBm = np.average(10*np.log10(np.abs(n_obs_main_trace_env)/1e-3))
print('Noise Floor at {} dBm'.format(NFdBm))
```

```
Noise Floor at -106.94752853967715 dBm
```

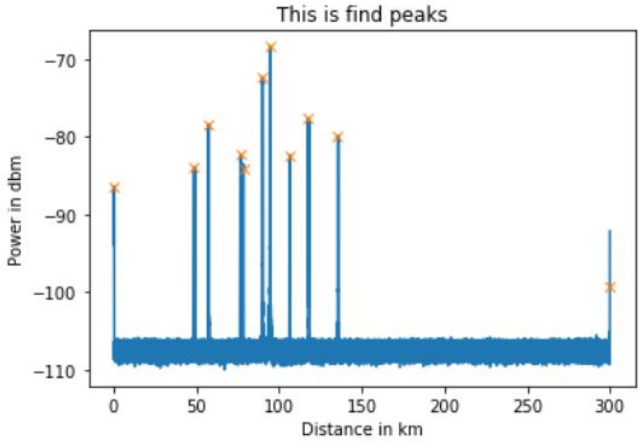
3. Detect Threshold

3.SNR Threshold function

In [13]:
```
S_max=-NFdBm*np.log(False_pro)/math.sqrt(n)
threshold=(1-False_pro)*S_max
print('threshold at {} dBm'.format(threshold))
```

```
threshold at -102.42052311596217 dBm
```

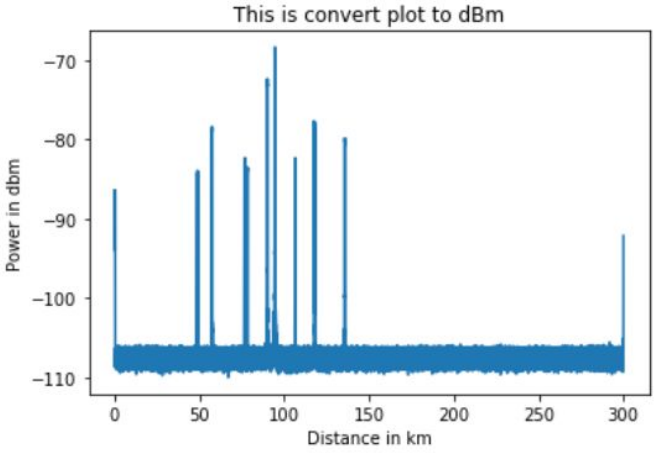
4. Last Peak




```
peaks:
[-86.38971163 -83.97586882 -78.46991618 -82.35399443 -84.14204728
 -72.42916714 -68.42552597 -82.36124662 -77.72046592 -79.86250198
 -99.30880873]
```

---

5. Convert Power to dBm



6. Replace peaks function

```
peaks, _ = find_peaks(power_dbm, height=threshold,distance=10e3)

plt.plot(x/1e3, power_dbm)
plt.plot(peaks*dx/1e3,power_dbm[peaks], "x")
plt.title('This is find peaks')
plt.xlabel('Distance in km')
plt.ylabel('Power in dbm')
```

Add Functions:

a. Probability of False Positive

1.Probablity of False Positive

In [11]:
```
import math
SNR=1
n=10
False_pro=math.exp(-SNR*math.sqrt(n))
print('False_pro is {}'.format(False_pro))
```

```
False_pro is 0.04232921962320499
```

b. Find noise floor

```
import math
from scipy import stats
import statistics
from statistics import median
from scipy.signal import find_peaks

NF=statistics.median([n_obs_main_trace_env])
NFdBm = np.average(10*np.log10(np.abs(n_obs_main_trace_env)/1e-3))
print('Noise Floor at {} dBm'.format(NFdBm))
```

```
Noise Floor at -106.7271595793336 dBm
```

c. SNR threshold

In [12]:
```
n=10
False_pro=0.05
S_max=-NFdBm*np.log(0.05)/math.sqrt(n)
threshold=(1-0.05)*S_max
print('threshold at {} dBm'.format(threshold))
```

```
threshold at -96.2419065442374 dBm
```

d. Report number of targets and distance

In [18]:
```
print('{} Targets found above {} dBm threshold:'.format(len(peaks),threshold))
print('These are their distance(km):')
print(peaks*dx/1e3)
print('')
print('')
print('')
print('peaks:')
print(power_dbm[peaks])
```

```
11 Targets found above -102.42052311596217 dBm threshold:
These are their distance(km):
[1.87500000e-03 4.88268750e+01 5.72838750e+01 7.66867500e+01
 7.85625000e+01 8.99137500e+01 9.43888125e+01 1.06291875e+02
 1.17538875e+02 1.35537750e+02 2.99996625e+02]
```