Sensor Array Signal Processing

Prof. Rodrigo C. de Lamare
CETUC, PUC-Rio, Brazil
and
Department of Electronic Engineering, University of York, UK
delamare@cetuc.puc-rio.br

Course material and assessment

- Lecture notes

- Textbook:
  - Detection, Estimation and Modulation, Part IV: Optimum Array Processing, Harry L. van Trees, Prentice Hall, 2001.

- Assessment:
  - 1 Exam paper (E).
  - 3 Lists of exercises (LE) on the topics covered.
  - Final grade (FG) = 0.25*LE+0.75*E (25% LE and 75% E)

Syllabus

I. Introduction
- Fundamentals
- Sensor arrays
- Discrete-time models

II. Beamforming
- Main principles
- Optimum beamforming
- Robust beamforming
- Adaptive algorithms

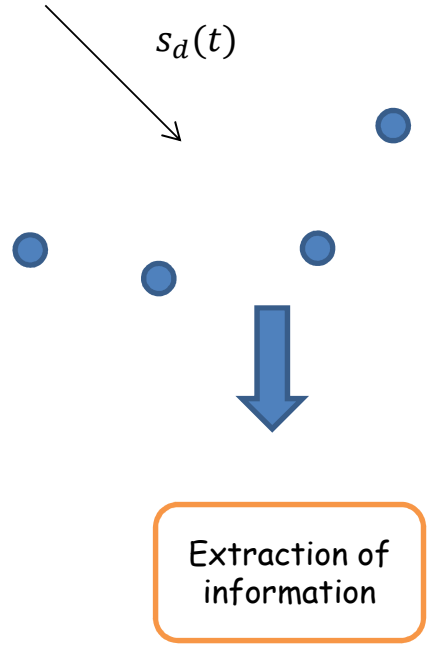
III. Direction finding
- Maximum likelihood estimation
- Cramèr-Rao lower bound
- Capon´s technique
- MUSIC
- ESPRIT

I. Introduction

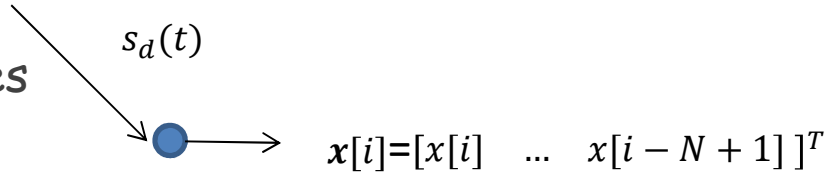
- Sensor array signal processing deals with the extraction of information from signals collected by a set of sensors.

- Types of information:
  - Signal content. Example: communications signals
  - Source location. Example: radar, sonar and localization algorithms.

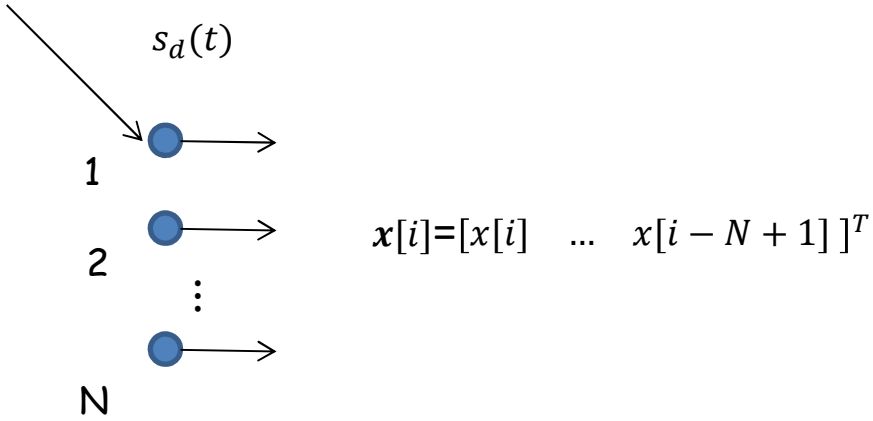
- In this course, we focus on the following fundamental aspects:
  - Discrete-time signal models
  - Beamforming
  - Direction finding



- Data model:
  - Single sensor: time series



  - Sensor array: samples collected at N sensors

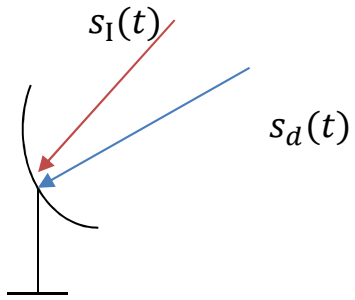


- Applications:

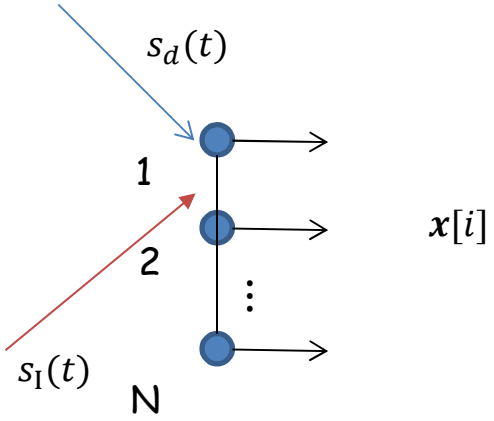
    - Electronic systems
    - Communications
    - Radar and sonar
    - Seismology
    - Astronomy

A. Fundamentals

- Problems in information extraction:
  - Interference.
  - Noise.
- Types of sensors:
  - Aperture:
    - Signals reflected in the direction of a (parabolic) disc are emphasized.
    - Spatial processing can be performed with the help of mechanical steering devices.
  - Sensor arrays:
    - Received signals can be combined and enhanced in a desired direction.
    - Signals can be combined with adjustable weights at the same time -> multiple signals can be extracted simultaneously.



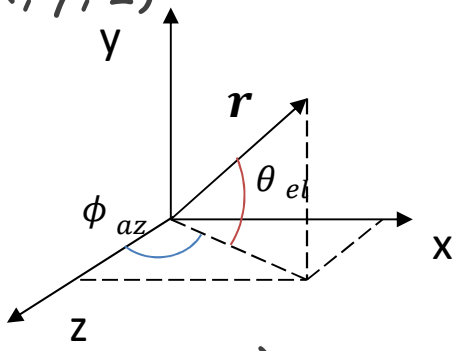
Parabolic disc



Sensor array

Description of spatial signals:

- Representation via Cartesian coordinates (x, y, z)



- Representation via spherical coordinates (R, ϕ_{az}, θ_{el})
  - $R = ||r||$ is the distance from origin
  - ϕ_{az} is the azimuth angle
  - θ_{el} is the elevation angle

Propagation of signals in space:

- Electromagnetic signals: governed by the solution of the wave equation.
- Acoustic signals: governed by the laws of acoustics.

Solution:

- Wave with unique frequency:

$$s(t, \boldsymbol{r}) = \frac{A}{\|\boldsymbol{r} - \boldsymbol{r}_0\|} e^{j2\pi f_c} \left(t - \frac{\|\boldsymbol{r} - \boldsymbol{r}_0\|}{c} \right)$$

- A is a complex amplitude
- f_c is the carrier frequency of the wave
- c is the speed of the propagation of the wave

- The signals travel over time.

- The propagation in space is governed by the coupling between space and time according to the wave equation.

- The wavelength of the electromagnetic wave is given by

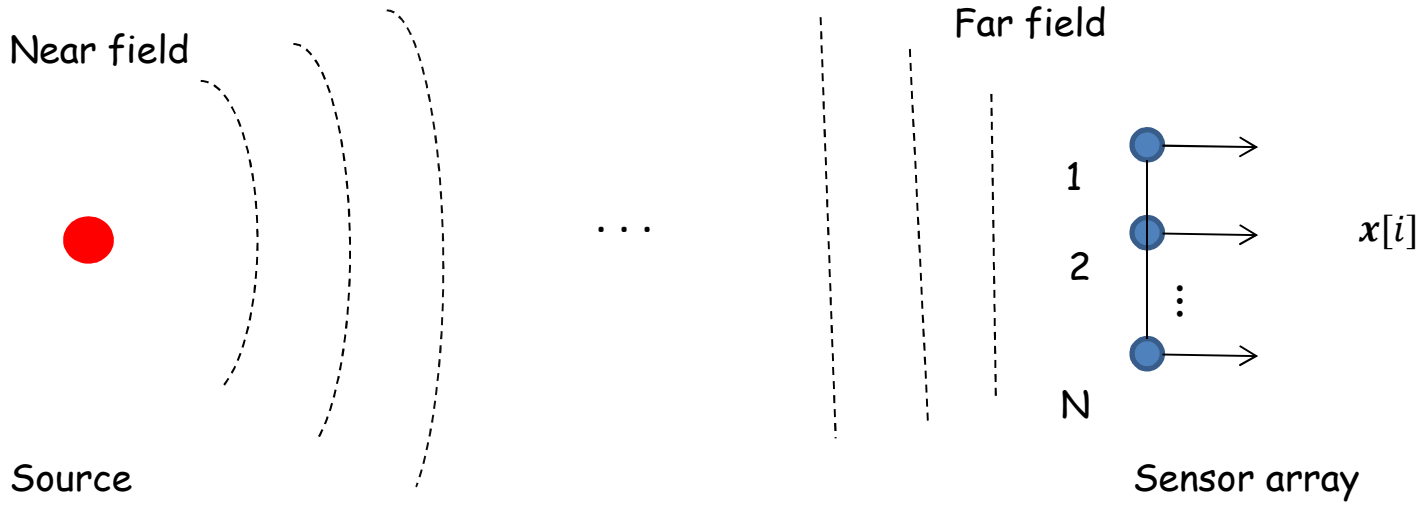
$$\lambda = \frac{c}{f_c},$$

where c is the speed of light.

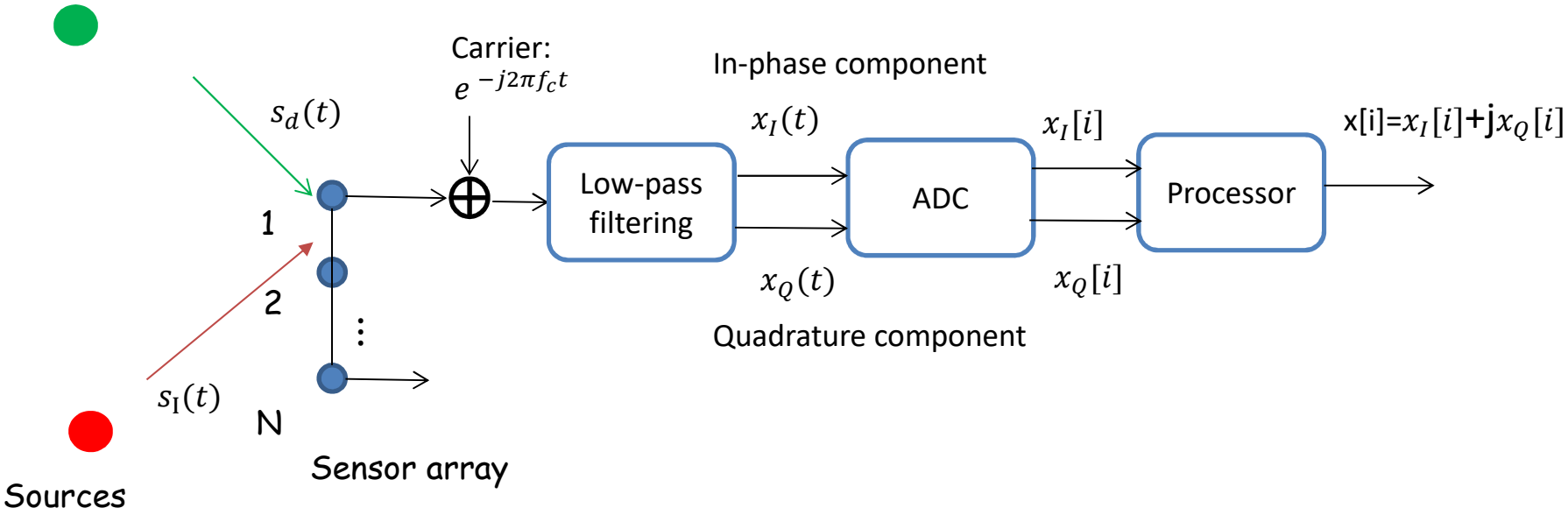
Assumptions:

- Signals are produced by a point source, whose size is much smaller than the distance between the source and the sensors.

- The source is in the far field, which means the source is located at a distance sufficiently large from the sensor array so the spherical wave can be approximated by a plane wave.



Reception of signals:

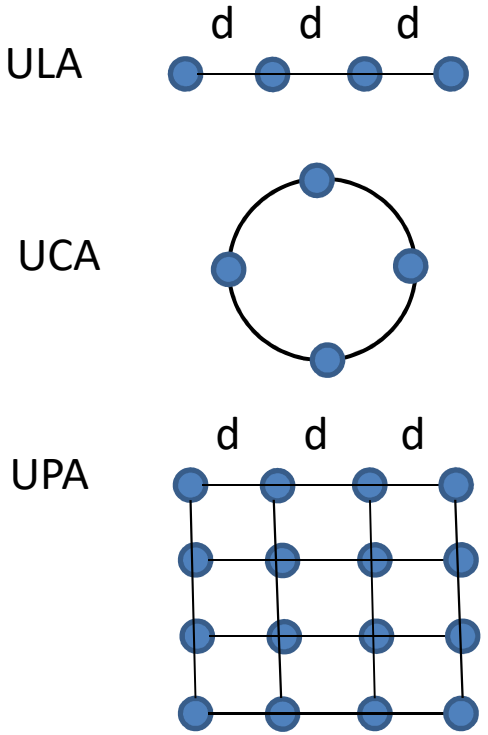


Sufficient statistics:

- Essential information in the form of sensor data $x[i]$ to compute the desired parameters or obtain optimal performance

B. Sensor arrays

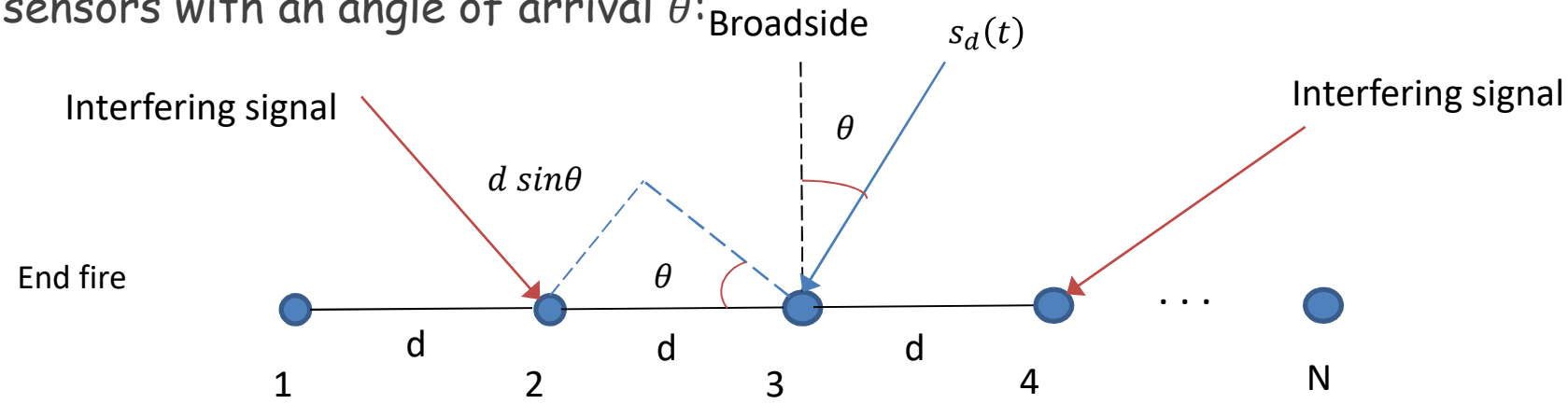
- Common geometries:
  - Uniform linear array (ULA)
  - Uniform circular array (UCA)
  - Uniform planar array (UPA)
- Other geometries:
  - Nested and co-prime arrays
  - Distributed arrays



- Assumption in our exposure: arrays of N isotropic sensors whose geometry affects the radiation properties.

ULA:

- Consider a signal $s_d(t) = s(t)e^{j2\pi f_c t}$ that impinges on a ULA with N sensors with an angle of arrival θ:

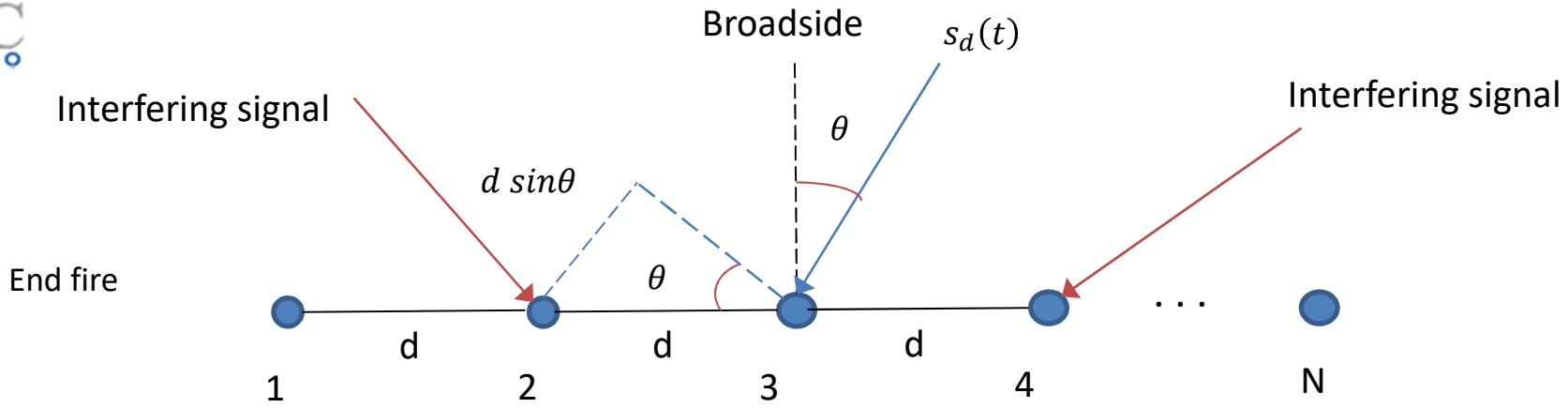


where θ is the direction of arrival

$\tau = \frac{d\ sin\theta}{c}$ is the delay of $s_d(t)$ between two adjacent sensors

d is the spacing between sensor elements

$d \sin\theta$ is the distance between between adjacent sensors

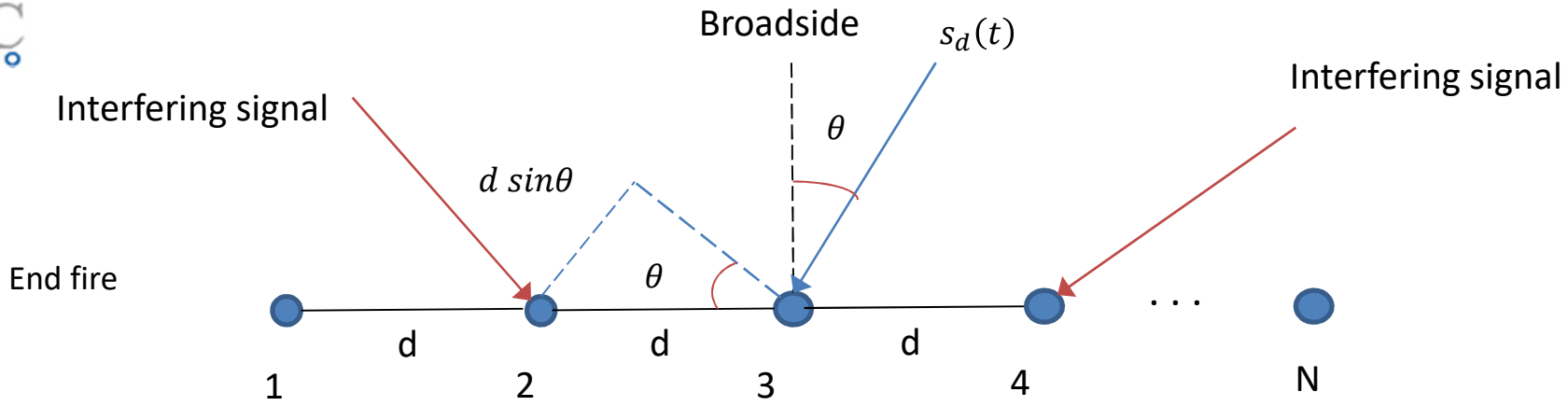


- The propagation delay of $s_d(t)$ between the first and the nth elements is

$$\tau_n = (n-1)\frac{d\sin\theta}{c}$$

- Multiplying $s_d(t)$ by the carrier $e^{-j2\pi f_c t}$ and using the 1st sensor as the reference, we have

$$\begin{aligned} s_1(t) &= s_d(t)e^{-j2\pi f_c t} \\ &= s(t)\ e^{j2\pi f_c t}e^{-j2\pi f_c t} = s(t) \end{aligned}$$



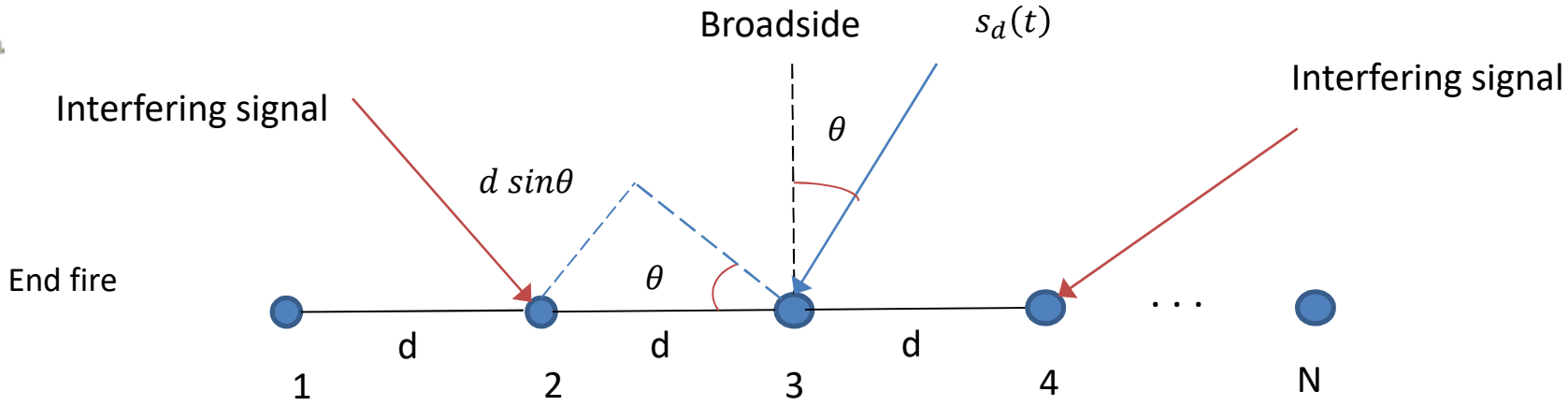
- Generalizing the previous result to the nth sensor, we have

$$s_n(t) = s_d(t)e^{-j2\pi f_c t\,\tau_n} = s(t)\,e^{-j2\pi f_c t}\;e^{j2\pi f_c t}\;e^{-j2\pi f_c \tau_n}$$

$$= s(t)e^{-j2\pi f_c (n-1)\frac{d\,\sin\theta}{c}}$$

$$= s(t)e^{-j2\pi (n-1)\frac{d\,\sin\theta}{\lambda}} =$$

where $\lambda = \frac{c}{f_c}$ is the wavelength of the carrier.



- The discrete-time signal received by a ULA can be written as

$$\boldsymbol{x}[i] = \begin{bmatrix} 1 \\ e^{-j2\pi\frac{d\sin\theta}{\lambda}} \\ \vdots \\ e^{-j2\pi(N-1)\frac{d\sin\theta}{\lambda}} \end{bmatrix} s[i] + \boldsymbol{j}[i] + \boldsymbol{n}[i] \in \mathbb{C}^N$$

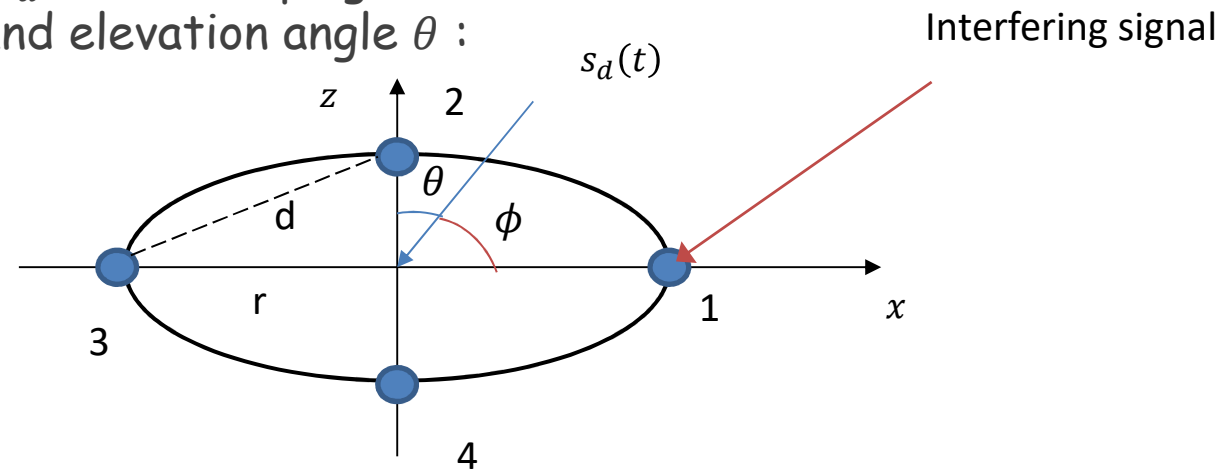
$$= \mathbf{a}(\theta)s[i] + \boldsymbol{n}[i],$$

where $\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ e^{-j2\pi\frac{d\sin\theta}{\lambda}} \\ \vdots \\ e^{-j2\pi(N-1)\frac{d\sin\theta}{\lambda}} \end{bmatrix}$ is the steering vector of a ULA,

$\boldsymbol{n}[i] \in \mathbb{C}^N$ is the noise vector and $\boldsymbol{j}[i] \in \mathbb{C}^N$ is the interference.

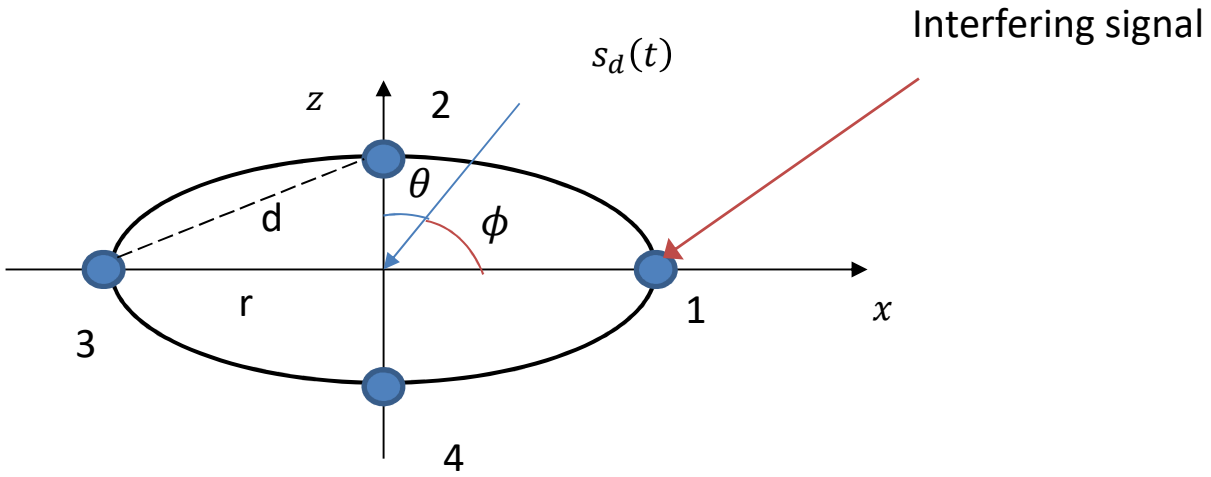
UCA:

- Consider a signal $s_d(t)$ that impinges on a UCA with N sensors with azimuth angle ϕ and elevation angle θ :



where d is the spacing between successive sensors on the circular structure

r is the radius of the circular structure

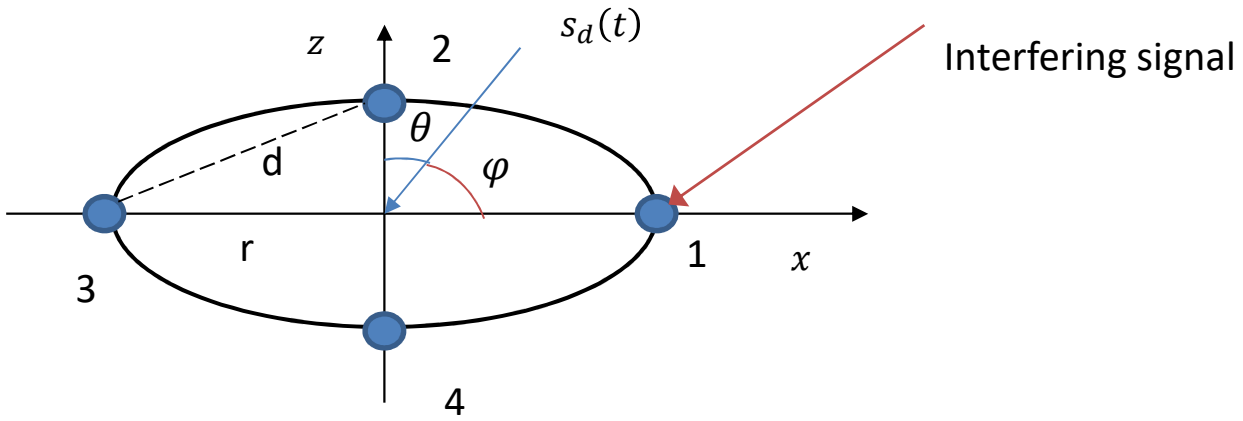


- The propagation delay of $s_d(t)$ between the first and the nth elements is

$$\tau_n = \frac{r}{c}\left(\sin\theta\cos\left(\frac{2\pi n}{N} - \phi\right)\right)$$

- Multiplying the signal $s_d(t)$ by the carrier $e^{-j2\pi f_c}$ and using the nth sensor as the reference, we have

$$\begin{aligned} s_n(t) &= s_d(t)\, e^{-j2\pi f_c \tau_n} e^{-j2\pi f_c t} \\ &= s(t) e^{j2\pi f_c t} e^{-j2\pi f_c t}\, e^{-j2\pi f_c \tau_n} \\ &= s(t) e^{-j2\pi f_c \frac{r}{c}\left(\sin\theta\cos\left(\frac{2\pi n}{N} - \phi\right)\right)} \\ &= s(t) e^{-j2\pi \frac{r}{\lambda}\left(\sin\theta\cos\left(\frac{2\pi n}{N} - \phi\right)\right)} \end{aligned}$$



- The discrete-time signal received by a UCA can be written as

$$\boldsymbol{x}[i] = \begin{bmatrix} e^{-j2\pi\frac{r}{\lambda}(\sin\theta\cos(\phi))} \\ e^{-j2\pi\frac{r}{\lambda}\left(\sin\theta\cos\left(\frac{2\pi}{N}-\phi\right)\right)} \\ \vdots \\ e^{-j2\pi\frac{r}{\lambda}\left(\sin\theta\cos\left(\frac{2\pi(N-1)}{N}-\phi\right)\right)} \end{bmatrix} s[i] + \boldsymbol{n}[i] \in \mathbb{C}^N$$

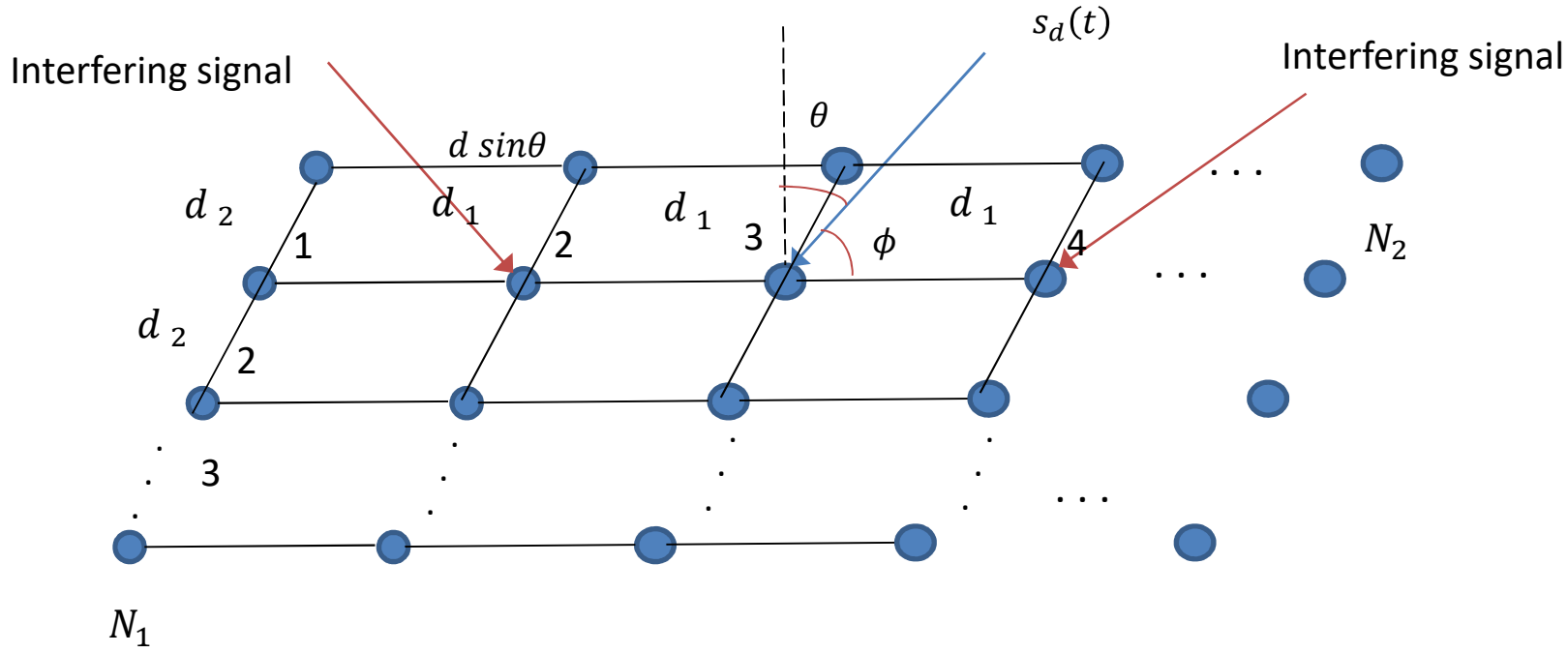
$$= \mathbf{a}(\theta,\phi)s[i] + \boldsymbol{n}[i],$$

where $\mathbf{a}(\theta,\varphi) = \begin{bmatrix} e^{-j2\pi\frac{r}{\lambda}(\sin\theta\cos(\phi))} \\ e^{-j2\pi\frac{r}{\lambda}\left(\sin\theta\cos\left(\frac{2\pi}{N}-\phi\right)\right)} \\ \vdots \\ e^{-j2\pi\frac{r}{\lambda}\left(\sin\theta\cos\left(\frac{2\pi(N-1)}{N}-\phi\right)\right)} \end{bmatrix}$ is the steering vector of a UCA,

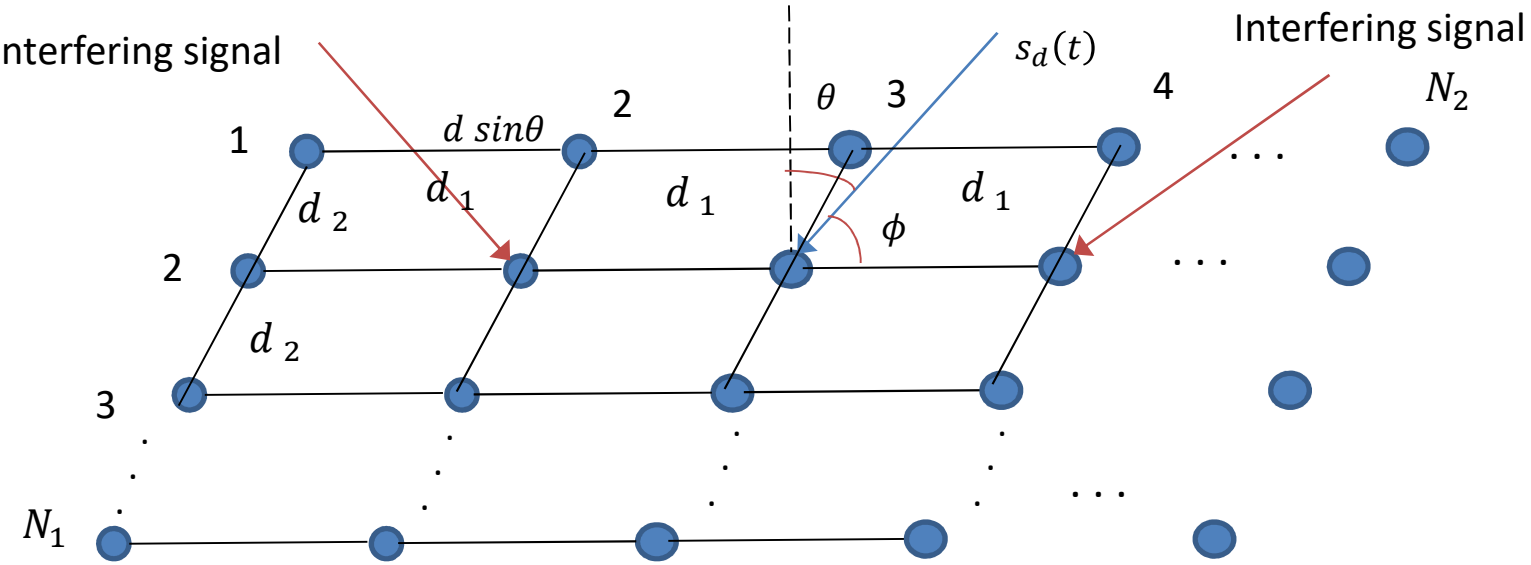
$\boldsymbol{n}[i] \in \mathbb{C}^N$ is the vector with the noise samples and $\boldsymbol{j}[i] \in \mathbb{C}^N$ is the interference.

UPA:

- Consider a signal $s_d(t)$ that impinges on a UPA with $N = N_1 N_2$ sensors with azimuth angle ϕ and elevation angle θ :



where d_1 and d_2 are the distances between adjacent sensors in the vertical and horizontal planes, respectively.

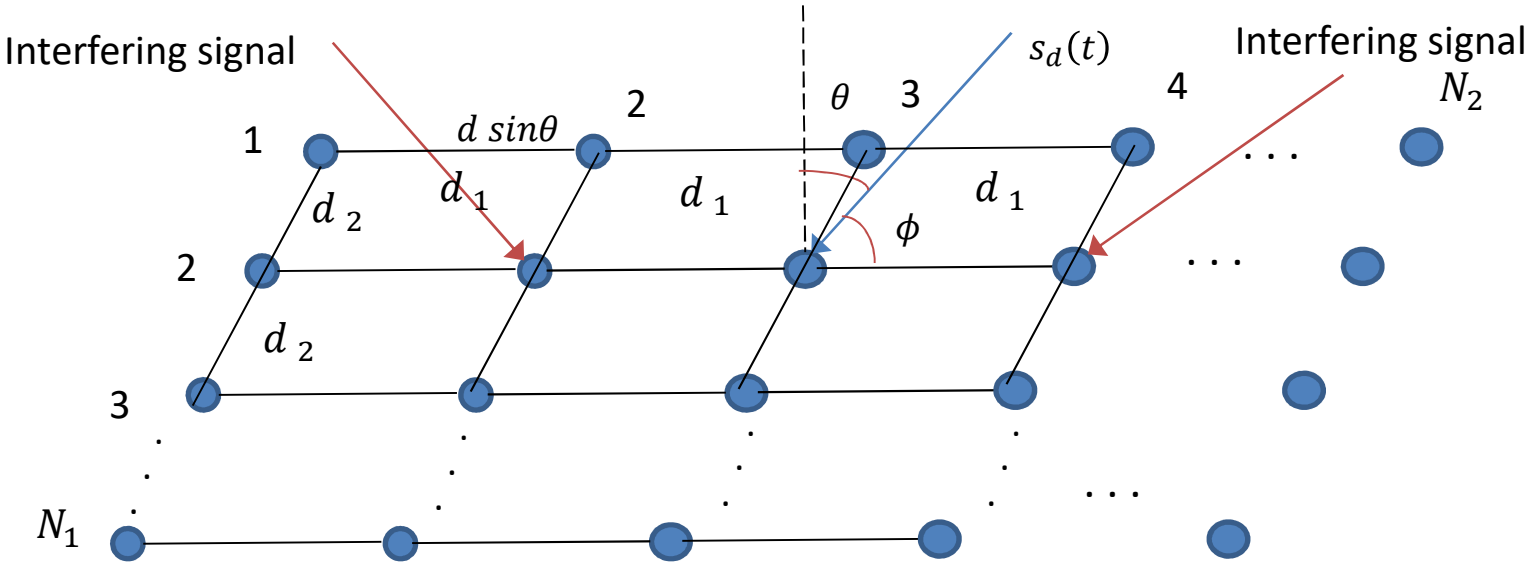


- The propagation delay of $s_d(t)$ introduced in the n_1, n_2th sensor elements is

$$\tau_{n_1,n_2} = \frac{n_1 d_1}{c}(\sin\theta\cos\phi) + \frac{n_2 d_2}{c}(\sin\theta\sin\phi)$$

- Multiplying the signal $s_d(t)$ by the carrier $e^{-j2\pi f_c t}$ and using the n_1, n_2th sensor as the reference, we have

$$\begin{aligned} s_{n_1,n_2}(t) &= s_d(t)e^{j2\pi f_c t}e^{-j2\pi f_c \tau_{n_1,n_2}} \\ &= s(t)e^{-j2\pi f_c \tau_{n_1,n_2}} \\ &= s(t)e^{-j2\pi f_c\left(\frac{n_1 d_1}{c}(\sin\theta\cos\phi)+\frac{n_2 d_2}{c}(\sin\theta\sin\phi)\right)} \\ &= s(t)e^{-j2\pi\left(\frac{n_1 d_1}{\lambda}\sin\theta\cos\phi+\frac{n_2 d_2}{\lambda}(\sin\theta\sin\phi)\right)} \end{aligned}$$



- The discrete-time signal received by a UPA can be written as

$$\boldsymbol{x}[i] = \begin{bmatrix} 1 \\ \vdots \\ e^{-j2\pi\left(\frac{n_1 d_1}{\lambda}\sin\theta\cos\phi + \frac{n_2 d_2}{\lambda}(\sin\theta\sin\phi)\right)} \\ \vdots \\ e^{-j2\pi\left(\frac{(N_1-1) d_1}{\lambda}\sin\theta\cos\phi + \frac{(N_2-1) d_2}{\lambda}(\sin\theta\sin\phi)\right)} \end{bmatrix} s[i] + \boldsymbol{n}[i] \in \mathbb{C}^N$$

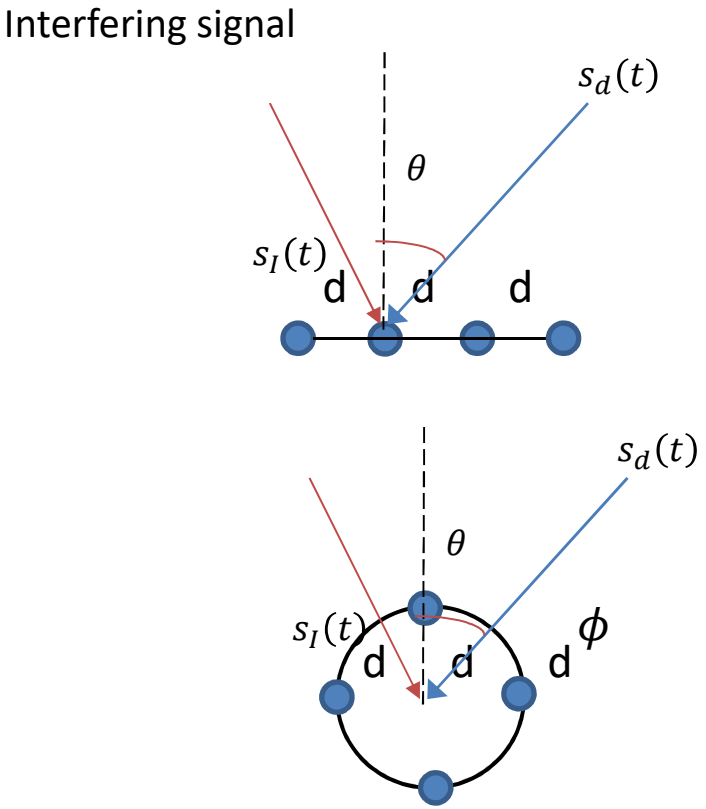
$$= \mathbf{a}(\theta,\phi)s[i] + \boldsymbol{j}[i] + \boldsymbol{n}[i],$$

where $\boldsymbol{n}[i] \in \mathbb{C}^N$ is the noise vector, $\boldsymbol{j}[i] \in \mathbb{C}^N$ is the interference and

$$\mathbf{a}(\theta,\phi) = \begin{bmatrix} 1 \\ \vdots \\ e^{-j2\pi\left(\frac{n_1 d_1}{\lambda}\sin\theta\cos\phi + \frac{n_2 d_2}{\lambda}(\sin\theta\sin\phi)\right)} \\ \vdots \\ e^{-j2\pi\left(\frac{(N_1-1) d_1}{\lambda}\sin\theta\cos\phi + \frac{(N_2-1) d_2}{\lambda}(\sin\theta\sin\phi)\right)} \end{bmatrix}$$ is the steering vector of a UPA.

- Spacing d:
  - Grating/aliasing effects should be avoided by choosing $d \geq \lambda/2$.
  - Affects the performance and the coupling between sensors.
  - Compact antennas require $d \leq \lambda/2$.

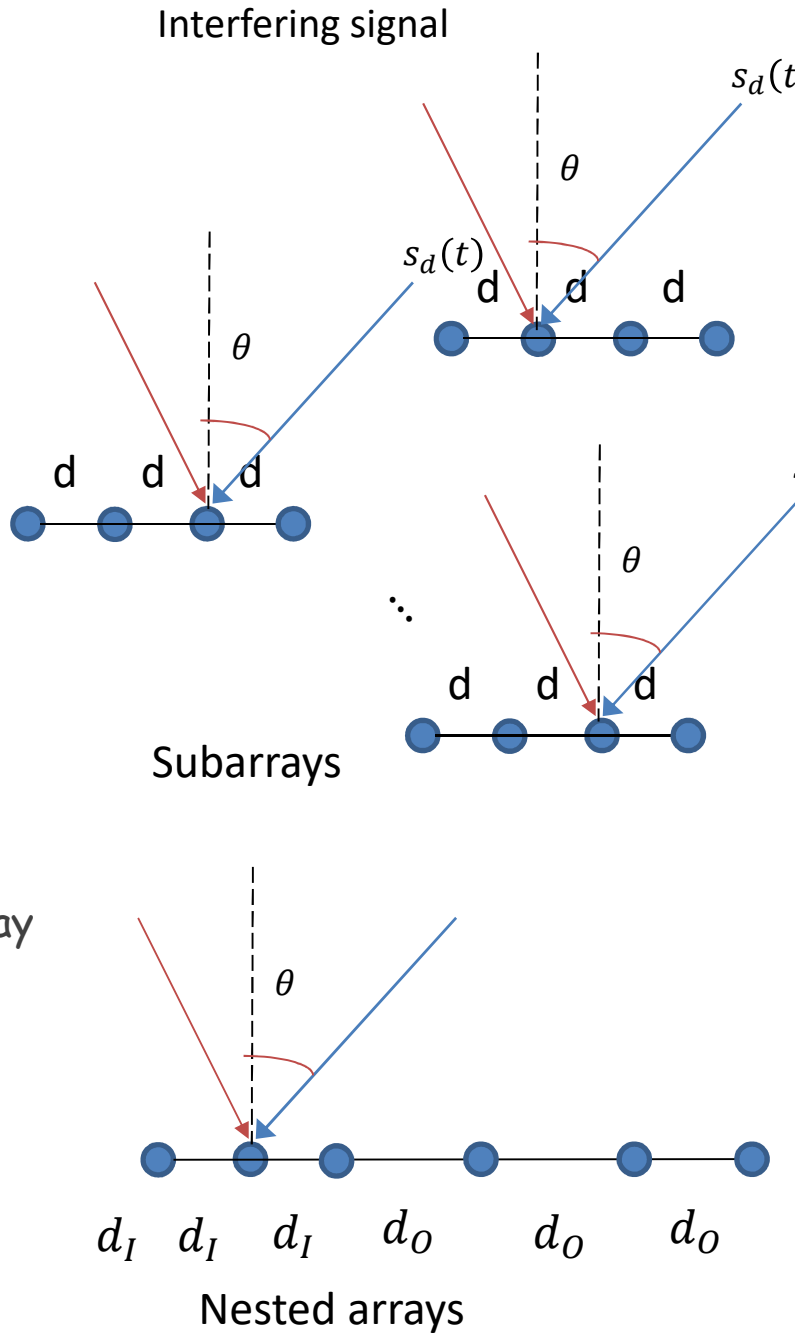
- Choice of sensor arrays:
  - Heavily application dependent.
  - UCAs have the advantage over ULAs and UPAs of creating beams through 360^0 without much change in beamwidth or sidelobe level.



H. Van Trees, *Optimum Array Processing - Detection, Estimation, and Modulation Theory, Part IV, Wiley, 2002.*

- Subarrays:
  - Reduce the cost of hardware and algorithms.
  - Require coordination for processing.

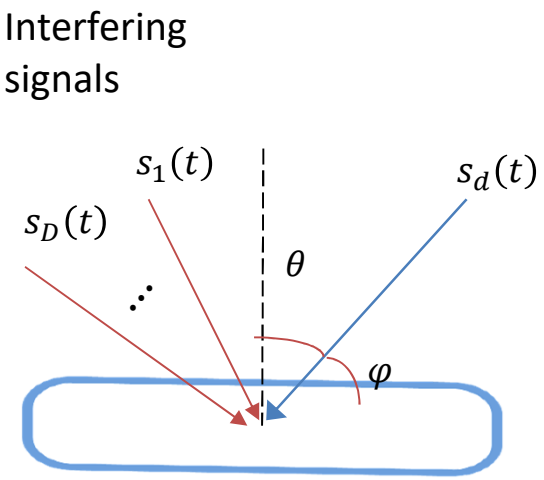
- Sparse arrays:
  - Clever approach to distribute the sensors by altering their spacing: nested, minimum redundancy and co-prime arrays.
  - Can resolve N^2 source signals as opposed to N with conventional sensor arrays.
  - Require more costly processing: use of co-array processing, building of N^2-dimensional covariance matrices and spatial smoothing.



Piya Pal and P. P. Vaidyanathan, Nested arrays: a novel approach to array processing with enhanced degrees of freedom, IEEE Trans. on Signal Processing, vol. 58, pp. 4167--4181, August 2010.

C. Discrete-time models

- The sensor array data model for arbitrary arrays can be expressed by



$$\boldsymbol{x}[i] = \mathbf{a}(\theta, \varphi) s_d[i] + \boldsymbol{j}[i] + \boldsymbol{n}[i] \in \mathbb{C}^N$$

$$= s_d[i]\mathbf{a}(\theta_d, \varphi_d) + \sum_{\boldsymbol{k=1, k\neq d}}^{\boldsymbol{D}} s_k[i]\mathbf{a}(\theta_k, \varphi_k) + \boldsymbol{n}[i]\,,$$

$$= \boldsymbol{A}(\boldsymbol{\theta}, \boldsymbol{\varphi})\boldsymbol{s}[i] + \boldsymbol{n}[i],$$

where $\mathbf{a}(\theta_d, \varphi_d) \in \mathbb{C}^N$ is the steering vector of the dth signal

$\boldsymbol{A}(\boldsymbol{\theta}, \boldsymbol{\varphi}) \in \mathbb{C}^{N \times D}$ is the array manifold

$\boldsymbol{s}[i] \in \mathbb{C}^N$ is the vector with all the signals that impinge on the sensor array

$\boldsymbol{n}[i] \in \mathbb{C}^N$ is the noise vector

Example: writing a code to simulate a ULA

```
function X = uladata(theta, P, L, sig2, N, d)
%
%  Generates L snapshots of ULA sensor data
%
%    theta  <- arrival angles of the D sources in degrees
%    P      <- The D X D covariance matrix of the source signals
%    L      <- number of snapshots to generate
%    sig2   <- noise variance
%    N      <- number of sensors
%    d      <- sensor spacing in wavelengths
%    X      -> N x L data matrix Y = [y(1),...,y(N)]
%
% generate the array manifold A matrix
A=exp(-2*pi*j*d*[0:N-1].'*sin([theta(:).']*pi/180));
%
% generate the source signals
D = max(size(P)); % dimensions of P matrix
s =(sqrtm(P)')*randn(D,L);      % s(t)
%
% generate the noise component
n = sqrt(sig2/2)*(randn(N,L)+j*randn(N,L));
%
% generate the ULA data
X = A*s+n;
```

Example:

```
D = 4;
N = 10;
theta = linspace(10,90,D);
P = eye(D); % same unit power
L = 20;
sig2 = 0.1; % noise power
d = 0.5;
X = uladata(theta, P, L, sig2, N, d);
```