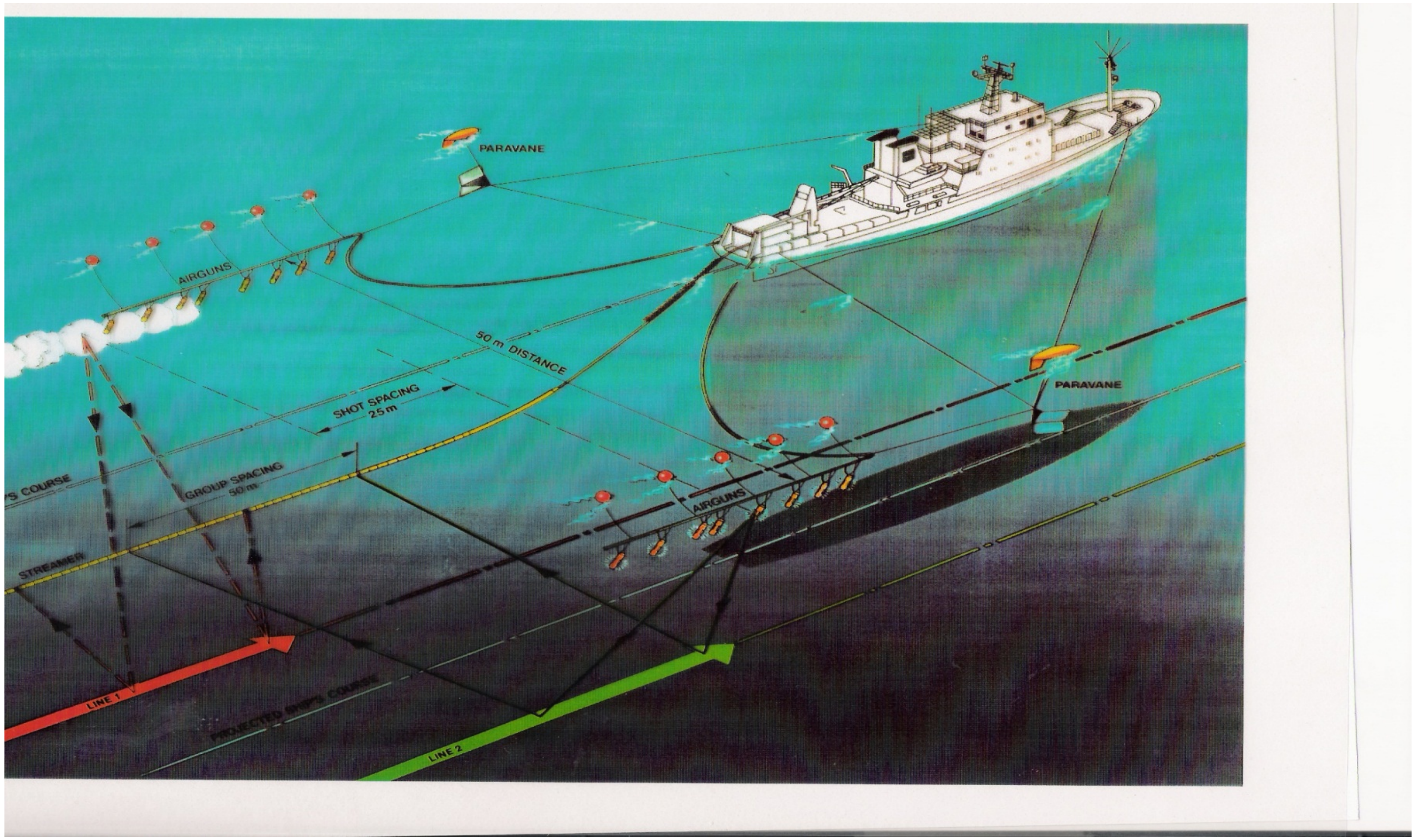
ADAPTIVE BEAMFORMING



Introduction

The aim is that the beamformer learns from the scenario what to do to provide the desired signal, reject interferences and minimises the un-directional noise

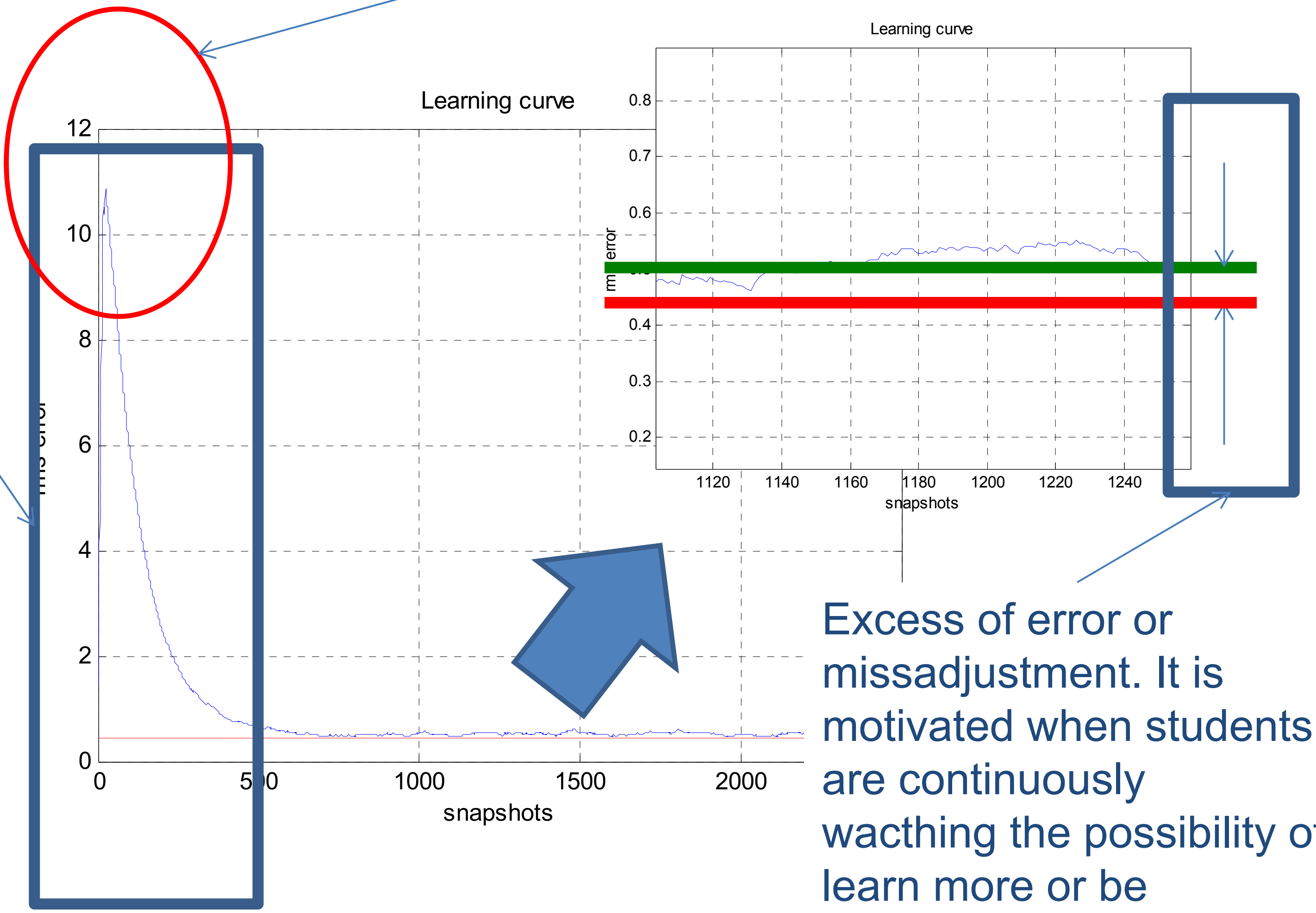
Any learning process needs to define the following:

- Evaluation process. Usually plotted as “learning curve” which reflects how “ignorance” evolve (decreases) with lectures taken. This degree of achievement can be global (all the students or single objective) or individual (multi-objective problem)
- Learning rule. Lectures with supervised or unsupervised (with teacher or without it) taken by the students.
- Confidence of the students on the learning rule. This is equivalent to “how much” of the class is worth to retain to decrease ignorance in the long term view.

Let us concentrate in the (global or individual) learning curve

Ignorance or error is high at the beginning of the learning

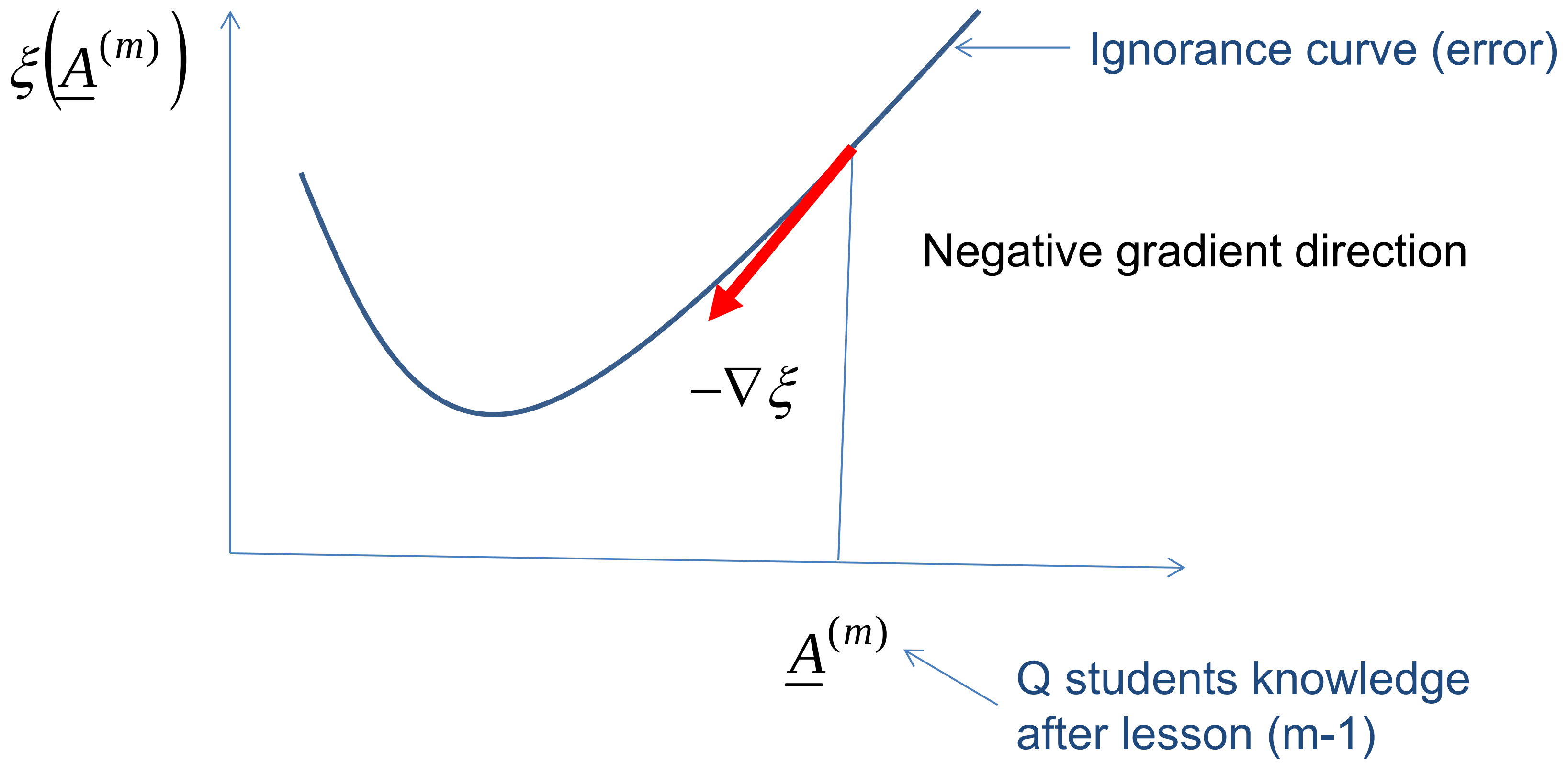
Learning Time or Time for convergence



Excess of error or missadjustment. It is motivated when students are continuously wacthing the possibility of learn more or be prepared to changes.

Learning Rule: The Gradient

The most popular learning rule is to move our knowledge on the contrary of the direction where our ignorance increases.


$\xi\left(\underline{A}^{(m)}\right)$
Ignorance curve (error)
Negative gradient direction
$-\nabla\xi$
$\underline{A}^{(m)}$
Q students knowledge after lesson (m-1)

Similarity Iterative/Adaptive

Let us image the problem where we consider the beamformer $\underline{A}^{(0)}$

This beamformer could be a vector of all ceros for example.

We would like to teach lessons to this beamformer such that it evolves to the optimum for an scenario with R and P being the snapshots covariance and the cross-covariance between the reference and the snapshots

Thus we like to teach the initial beamformer such that

$$\underline{A}^{(0)} \Rightarrow \underline{A}_{opt} = \underline{\underline{R}}^{-1}\underline{P}$$

The ignorance will be the MSE error

$$\xi^{(m)} = P_d + \underline{A}^{(m)H}\underline{\underline{R}}\,\underline{A}^{(m)} + \underline{A}^{(m)H}\underline{P} + \underline{P}^H\underline{A}^{(m)}$$

or

$$\xi^{(m)} = \xi_{\min} + \left(\underline{A}^{(m)} - \underline{A}_{opt}\right)^H \underline{\underline{R}}\left(\underline{A}^{(m)} - \underline{A}_{opt}\right)$$

The gradient is: $\nabla \xi^{(m)} = \underline{\underline{R}}\,\underline{A}^{(m)} - P$

And, the learning rule will be:

$$\underline{A}^{(m+1)} = \underline{A}^{(m)} - \mu \nabla \xi^{(m)} = \underline{A}^{(m)} - \mu\left(\underline{\underline{R}}\,\underline{A}^{(m)} - P\right)$$

$$\underline{A}^{(m+1)} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)\underline{A}^{(m)} + \mu \underline{P}$$

Let us check convergence. Convergence implies that after many lessons taken (m->infinity) the knowledges stabilices on

$$\underline{A}^{(m+1)} = \underline{A}^{(m)} = \underline{A}^{(\infty)}$$

$$\underline{A}^{(\infty)} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)\underline{A}^{(\infty)} + \mu \underline{P} \Rightarrow \underline{A}^{(\infty)} = \underline{\underline{R}}^{-1}\underline{P} = \underline{A}_{opt}$$

Thus, the steady state of this learning system is the desired with absolute minimum ignorance. This is thanks that the ignorance has a single minimum otherwise the rule may converge to local minima. THERE IS NOT MISSADJUSMENT a special feature of iterative, instead adaptive, learning (!!!)

Convergence analysis:

Assuming that the learning rule is for a single student (Q=1), the recursive nature of the learning equation turns to be an IIR with a single pole. Asking for stability is to impose that the pole lie inside the unit circle.

Thus, we will write as a FIR (non-recursive) the IIR (recursive) equation of our learning rule:

$$\underline{A}^{(m+1)} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)\underline{A}^{(m)} + \mu \underline{P} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)^m \left(\underline{A}^{(0)} - \underline{A}_{opt}\right) + \underline{A}_{opt}$$

or

$$\boxed{\left(\underline{A}^{(m)} - \underline{A}_{opt}\right)} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)^m \boxed{\left(\underline{A}^{(0)} - \underline{A}_{opt}\right)}$$

Ignorance after m lectures

Initial ignorance

In summary, convergence is ensured whenever

$$\lim_{m \Rightarrow \infty} \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)^m = \underline{\underline{0}}$$

Functions of def. positive matrixes

For any continuous function f(x), the Taylor's series is:

$$f(x)=\sum_{m=0}^{\infty} x^m \frac{f^{(m)}(0)}{m!}$$

For any positive define matrix, we can write successive power of it in terms of its eigenvalues as:

$$\underline{\underline{R}}=\underline{\underline{E}}\,\underline{\underline{D}}\,\underline{\underline{E}}^H \Rightarrow \underline{\underline{R}}^m=\underline{\underline{E}}\,\underline{\underline{D}}^m\,\underline{\underline{E}}^H$$

In consequence, we can define the function f(.) of a matrix as:

$$f(\underline{\underline{R}})=\sum_{m=0}^{\infty}\underline{\underline{R}}^m\frac{f^{(m)}(0)}{m!}=\underline{\underline{E}}\left(\sum_{m=0}^{\infty}\underline{\underline{D}}^m\frac{f^{(m)}(0)}{m!}\right)\underline{\underline{E}}^H$$

Since, for every eigenvalue the series is formed by:

$$\sum_{m=0}^{\infty}\lambda_q^m\frac{f^{(m)}(0)}{m!}=f(\lambda_q)$$

In summary, any continuous function of a matrix PASS DIRECTLY to its eigenvalues, yet preserving the same set of eigenvectors

Using this result on the gradient learning rule we have:

$$\underline{A}^{(m+1)} = \left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)^m \left(\underline{A}^{(0)} - \underline{A}_{opt}\right) + \underline{A}_{opt}$$

Where convergence is assured whenever the powers of the above matrix tend to zero when m tends to infinity. Since the power for any iteration m is:

$$\left(\underline{\underline{I}} - \mu \underline{\underline{R}}\right)^m = \left(\underline{\underline{E}}\,\underline{\underline{E}}^H - \mu \underline{\underline{E}}\,\underline{\underline{D}}\,\underline{\underline{E}}^H\right)^m = \underline{\underline{E}}\left(\underline{\underline{I}} - \mu \underline{\underline{D}}\right)^m \underline{\underline{E}}^H$$

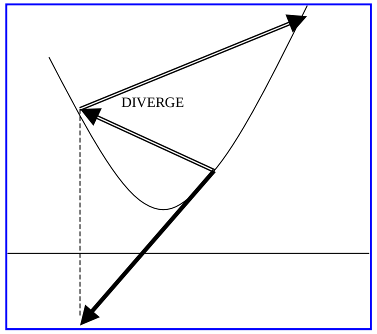
Thus, convergence is ensured for: $\left|1 - \mu \lambda_q\right| < 1 \quad q = 1, Q$

Clearly the most critical eigenvalue is the maximum one, so convergence is ensured when

$$\boxed{\mu < \frac{2}{\lambda_{\max}}}$$

Thus, convergence and its rate is controlled by the so-called eigen-modes that evolve to remove ignorance of the beamformer as:

$$\left|\left(1-\mu\lambda_q\right)^m\right| < \left|\left(1-\frac{2\lambda_q}{\lambda_{\max}}\right)^m\right|$$

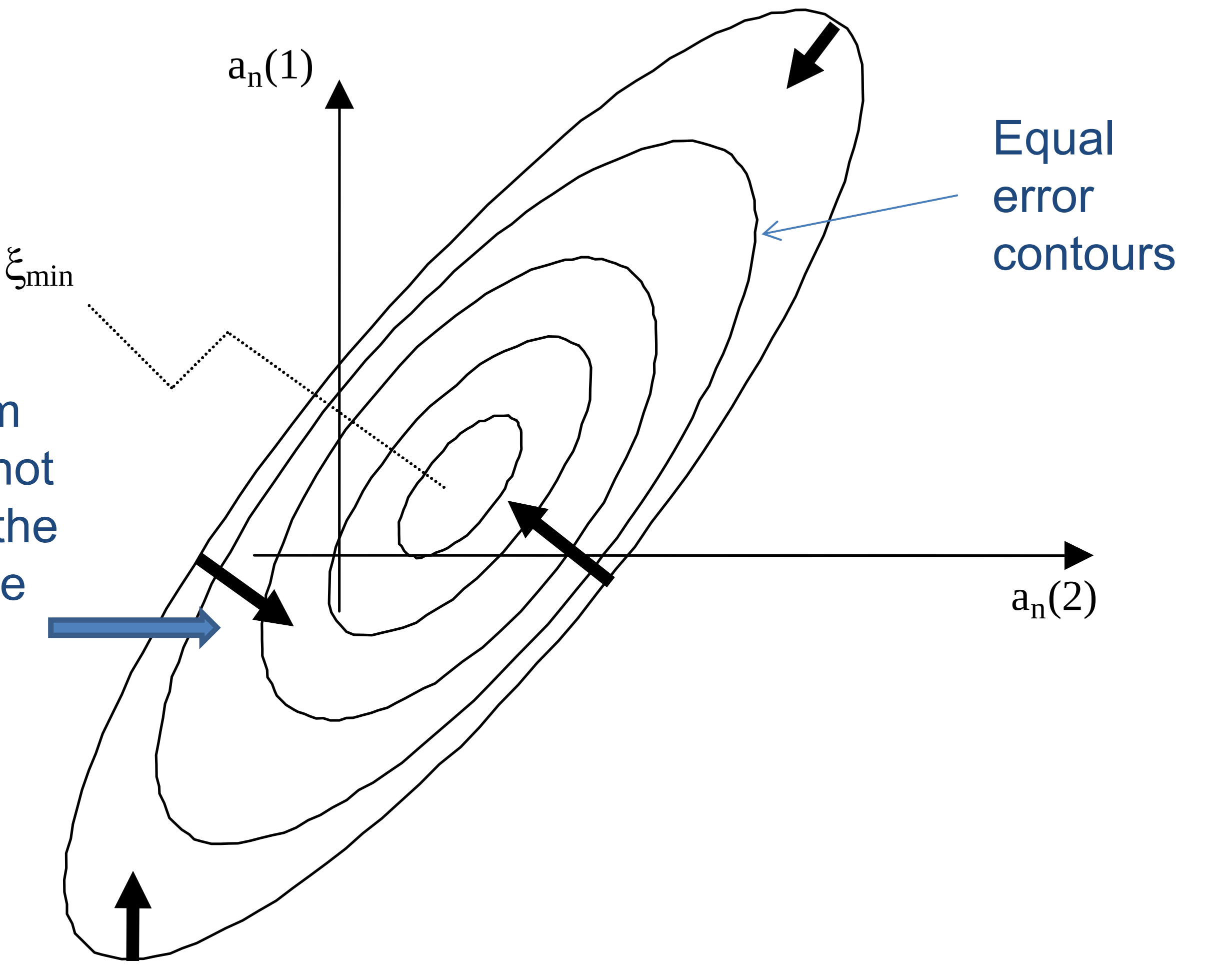


This reveals that high energy modes converge faster than low energy modes. In consequence, the global convergence is bounded by the minimum eigenvalue. Setting the time (number of iterations) as the value of m (n_o) such that the power is reduced to 0.1 the initial value, we get an effective rate of convergence as:

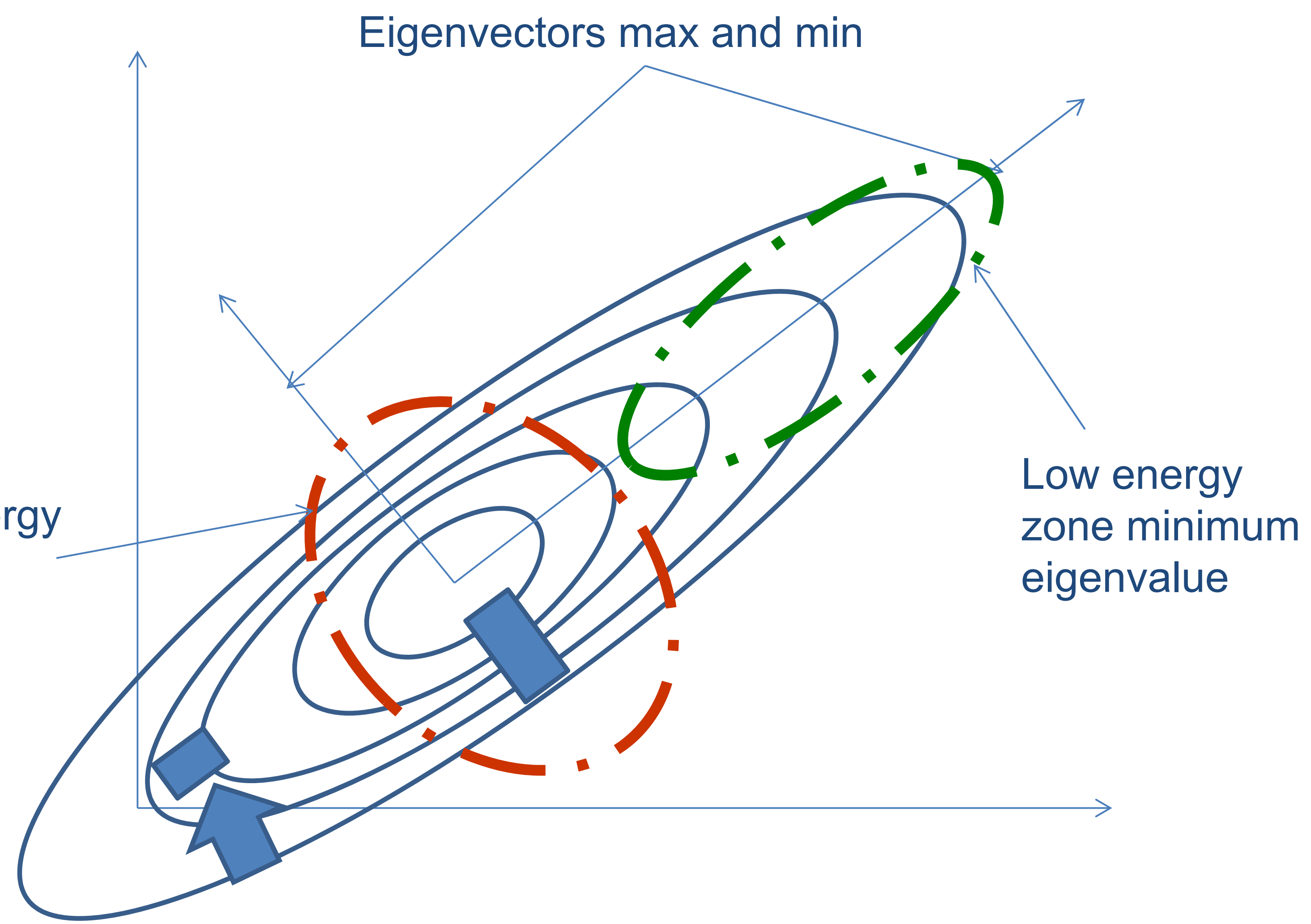
$$\left(1-\mu\lambda_{\min}\right)^{n_0} = 0.1 \Rightarrow n_0 = \frac{\ln(10)}{\ln(1-\mu\lambda_{\min})} \approx 2.3\frac{1}{\mu\lambda_{\min}}$$

This reveals one of the fundamental limitation of the gradient methods which is that the steep size μ has to be high enought to speed up convergence but it is upper bounded in order to avoid convergence.

Geometric view (Q=2)



The maximum gradient does not coincides with the direction to the optimum



One pass of the gradient consumes almost 80% of the path to minimum in the high energy zone meanwhile it goes ahead just 8% on le low energy one

Summary of the Gradient Algorithm

Remind that the goal of this algorithm is to solve iteratively the optimum beamformer without requiring the computation of the inverse covariance matrix.

$$\underline{A}^{(m+1)} = \underline{A}^{(m)} - \mu\left(\underline{\underline{R}}\,\underline{A}^{(m)} - \underline{P}\right)$$

$$\mu < \frac{2}{\lambda_{\max}}$$

$$n_0 \approx 2.3\frac{1}{\mu\lambda_{\min}}$$

Since the goal was to avoid matrix inversion we need a easy to compute version for steep size and convergence rate, avoiding those that require svd of the covariance matrix.

With respect $\boxed{\mu < \frac{2}{\lambda_{\max}}}$ since

$$Trace\left(\underline{\underline{R}}\right) = E\left(\underline{X}_n \underline{X}_n^H\right) = \sum_{q=1}^{Q} \lambda_q = \left|\begin{array}{l}\text{Since the}\\ \text{matrix is}\\ \text{definite}\\ \text{positive}\end{array}\right| < \lambda_{\max}$$

Setting the steep-size equal to

$$\mu = \frac{2\alpha}{tr\left(\underline{\underline{R}}\right)} \quad with \quad 0 < \alpha < 1$$

Convergence is granted

The trace can be estimated recursively from snapshots as:

$$Trace\left(\underline{\underline{R}}\right) \Rightarrow P(n) = \begin{cases} if \quad P(n) > P_o & \beta.P(n) + (1-\beta)\underline{X}_n^H \underline{X}_n \\ & else \quad P_o \end{cases}$$

- Threshold set to prevent RF set-off
- Implemeted as a look-at-table to reduce computational load

β selected in the range 0.9-0.99. The memory introduced by this parameter to detect changes on the snapshot power will be 1/(1-β)

With respect $n_0 \approx 2.3\frac{1}{\mu\lambda_{\min}} = 1.15\frac{\lambda_{\max}}{\lambda_{\min}}$

Note that the maximum eigenvalues is bounded by the trace. When the power at each antenna is the same, the trace is Q times the power in a single antenna P_x

$$\boxed{n_0 \leq 1.15\frac{QP_x}{\sigma^2}}$$

With respect the minimum eigenvalue (see Music for example), it coincides wit the power of the un-directional noise σ^2

As a consequence:

- Large arrays (high Q) will experience slower convergence than arrays with a few sensors
- High power scenarios will relent convergence as well.

Estimating the gradient: LMS

Moving the iterative framework described before to the adaptive arena, the first problem we face is that, for time-varying scenarios, the gradient pass to be a random variable that has to be estimated from data.

The LMS is the simplest estimate (theoretically) the worst teacher (largest variance and randomness it its conferences) for the learning rule BUT, his high dedication (it offers a conference per snapshot) largely compensates the quality of his teaching.

$$\left(\underline{\underline{R}}\,\underline{A}^{(m)} - \underline{P}\right)$$

Estimated by its INSTANTANEOUS value

$$\underline{X}_n \underline{X}_n^H \underline{A}_n - \underline{X}_n d^*(n) = \underline{X}_n\left(\underline{X}_n^H \underline{A}_n - d^*(n)\right) = -\underline{X}_n \varepsilon^*(n)$$

The LMS Learning Rule is:

$$\boxed{\underline{A}_{n+1} = \underline{A}_n + \mu \underline{X}_n \varepsilon^*(n)}$$

Note that the steep-size and the convergence rate remain as stated for any gradient algorithm

Nevertheless, the prize we pay for making adaptive the beamforming is that the learning rule become random. This will cause a MISS-ADJUSTMENT error or an excess of error due to the continuous learning behavior that the adaptive nature of the system forces in the beamformer.

In fact, the miss-adjustment can be considered as additional noise that have to be taken into account in order to control the overall SNR budget of the array system.

Let us imagine that the optimum beamformer is substracted from the learning rule……

$$\left(\underline{A}_{n+1} - \underline{A}_{opt}\right) = \left(\underline{A}_n - A_{opt}\right) + \mu \underline{X}_n \varepsilon^*(n)$$

Define the coefficients error as $\underline{\tilde{A}}_n = \underline{A}_{opt} - \underline{A}_n$

then $\underline{\tilde{A}}_{n+1} = \underline{\tilde{A}}_n - \mu \underline{X}_n \varepsilon^*(n)$

Also, using that the excess of error due to a given error in the coefficients is given by:

$$\xi_n = \xi_{\min} + \underline{\tilde{A}}_n^H \underline{\underline{R}} \underline{\tilde{A}}_n$$

The expected value of this random variable is:

$$\xi = E(\xi_n) = \xi_{\min} + E\left(\underline{\tilde{A}}_n^H \underline{\underline{R}} \underline{\tilde{A}}_n\right) = \xi_{\min} + E\left(trace\left[\underline{\tilde{A}}_n \underline{\tilde{A}}_n^H \underline{\underline{R}}\right]\right) =$$

$$\boxed{\xi = \xi_{\min} + tr\left[\underline{\underline{\Sigma}}_n \underline{\underline{R}}\right]}$$

where $\underline{\underline{\Sigma}}_n = E\left(\underline{\tilde{A}}_n \underline{\tilde{A}}_n^H\right)$

The miss-adjustment is defined as the difference between the experienced error and the minimum as a percentage of it, i.e. 10% miss-adjustment implies a 10% of excess error over the minimum one.

$$M = \frac{\xi - \xi_{\min}}{\xi_{\min}}.100\% = \frac{tr\left[\underline{\underline{\Sigma}}_n \underline{\underline{R}}\right]}{\xi_{\min}}100\%$$

Going back to the learning rule of the LMS

$$\underline{\tilde{A}}_{n+1} = \underline{\tilde{A}}_n - \mu \underline{X}_n \varepsilon^*(n)$$

We can write how the error evolves with respect the coefficients error vector

$$\varepsilon(n) = \varepsilon_{\min}(n) + \underline{\tilde{A}}_n^H \underline{X}_n$$

$$\varepsilon^*(n) = \varepsilon^*_{\min}(n) + \underline{X}_n^H \underline{\tilde{A}}_n$$

This error has power equal to $\xi_{\min}$ and it is orthogonal to the current snapshot (Orthogonality principle)

Now, we can compute how the covariance of the coefficients evolves from the learning rule

$$\underline{\tilde{A}}_{n+1} = \underline{\tilde{A}}_n - \mu \underline{X}_n \varepsilon^*(n)$$

$$\Downarrow$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{\Sigma}}_n - 2\,\mathrm{cov}\left(\mu \underline{X}_n \left(e^*_{\min} + \underline{X}_n^H \underline{\tilde{A}}_n \right) \underline{\tilde{A}}_n^H \right) + \mu^2 \xi_{\min} \underline{\underline{R}}$$

Being close to convergence $\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{\Sigma}}_n = \underline{\underline{\Sigma}}$

then

$$\underline{\underline{0}} = -2\,\mathrm{cov}\left(\mu \underline{X}_n \left(e^*_{\min} + \underline{X}_n^H \underline{\tilde{A}}_n \right) \underline{\tilde{A}}_n^H \right) + \mu^2 \xi_{\min} \underline{\underline{R}}$$

$$2\mu \underline{\underline{R}}\,\underline{\underline{\Sigma}} = \mu^2 \xi_{\min} \underline{\underline{R}}$$

In summary the covariance of the coefficients error is diagonal, in consequence after arriving close to convergence the students do not cooperate and their ignorance stays uncorrelated.

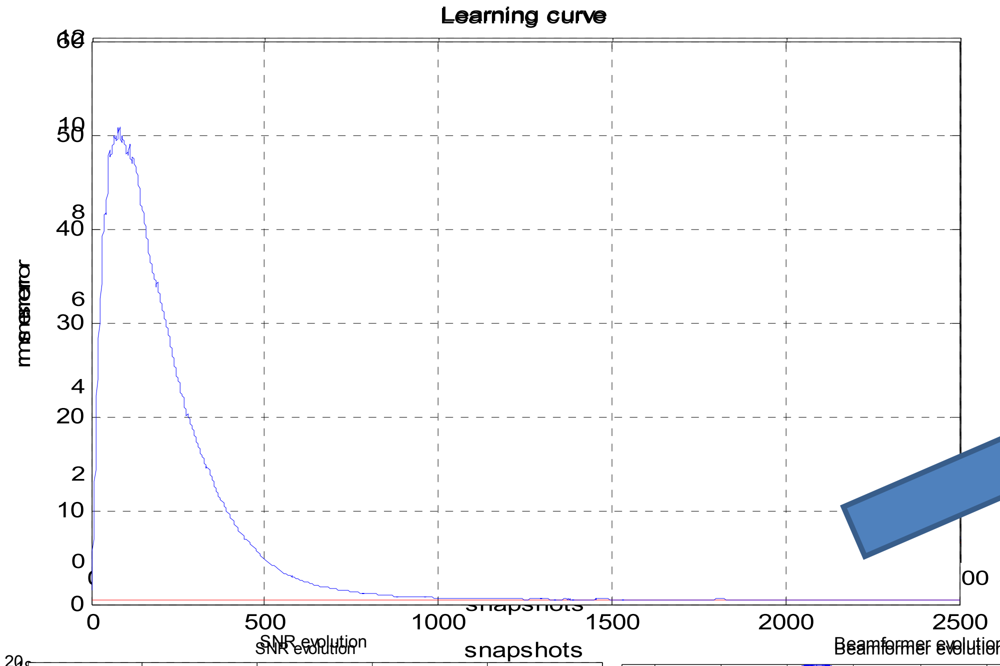
$$\underline{\underline{\Sigma}} = \frac{\mu \xi_{\min}}{2} \underline{\underline{I}}$$

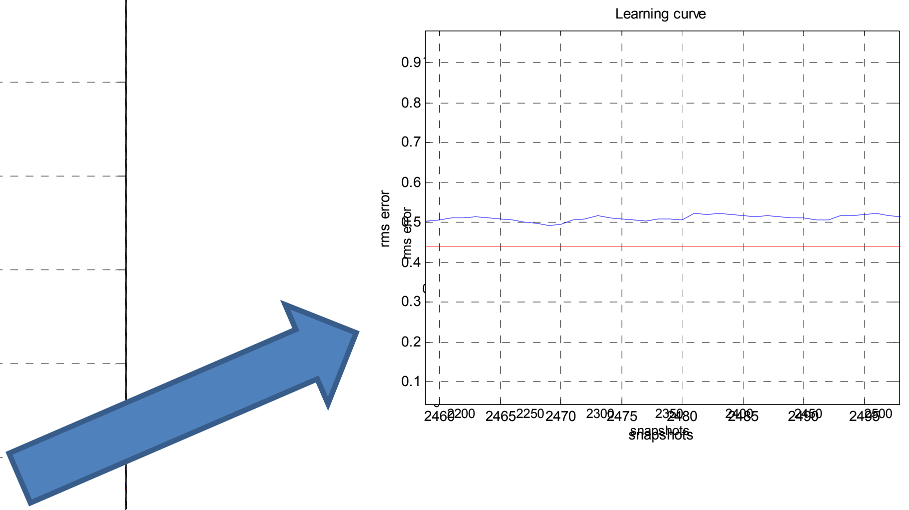
And the miss-adjusment is:

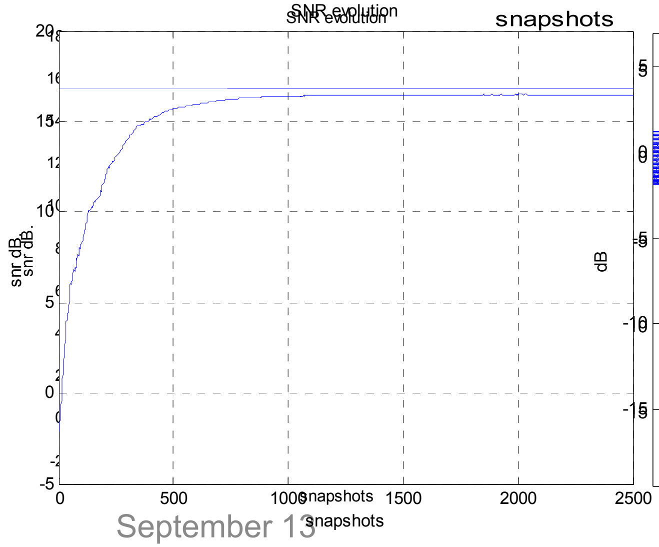
$$\boxed{M = \frac{\mu}{2} tr\left(\underline{\underline{R}}\right) 100\% = \alpha .100\%}$$

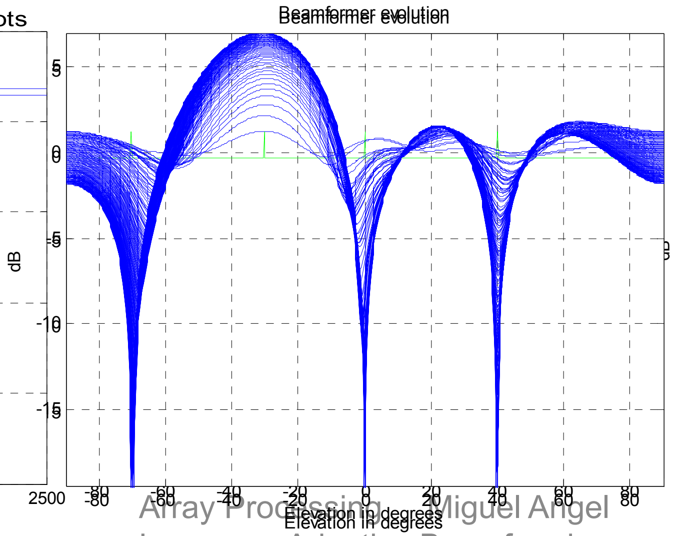
Parameter α controls directly the miss-adjusment, BUT at the same time small values relent convergence

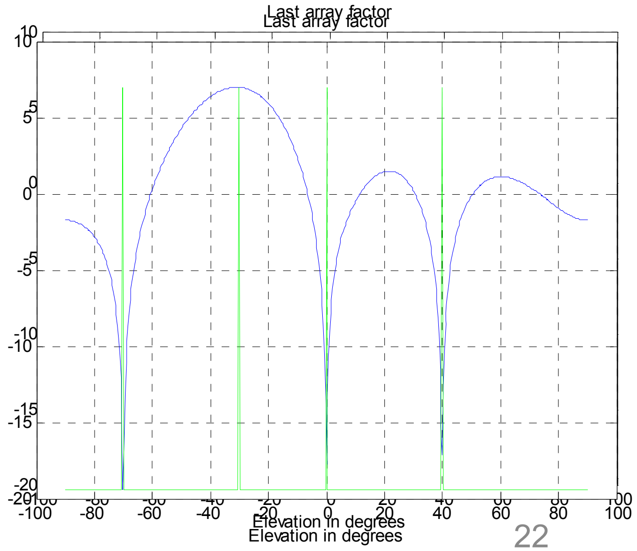
LMS Performance











DSD: The best gradient estimate

Many people do believe that the performance of the “bad teacher highly dedicated” LMS can be overpassed by a perfect teacher but lazzy.

$$\underline{A}^{(m+1)} = \underline{A}^{(m)} - \mu\boxed{\left(\underline{\underline{R}}\,\underline{A}^{(m)} - \underline{P}\right)}$$

$$\mu < \frac{2}{\lambda_{\max}} \qquad n_0 \approx 2.3\frac{1}{\mu\lambda_{\min}}$$

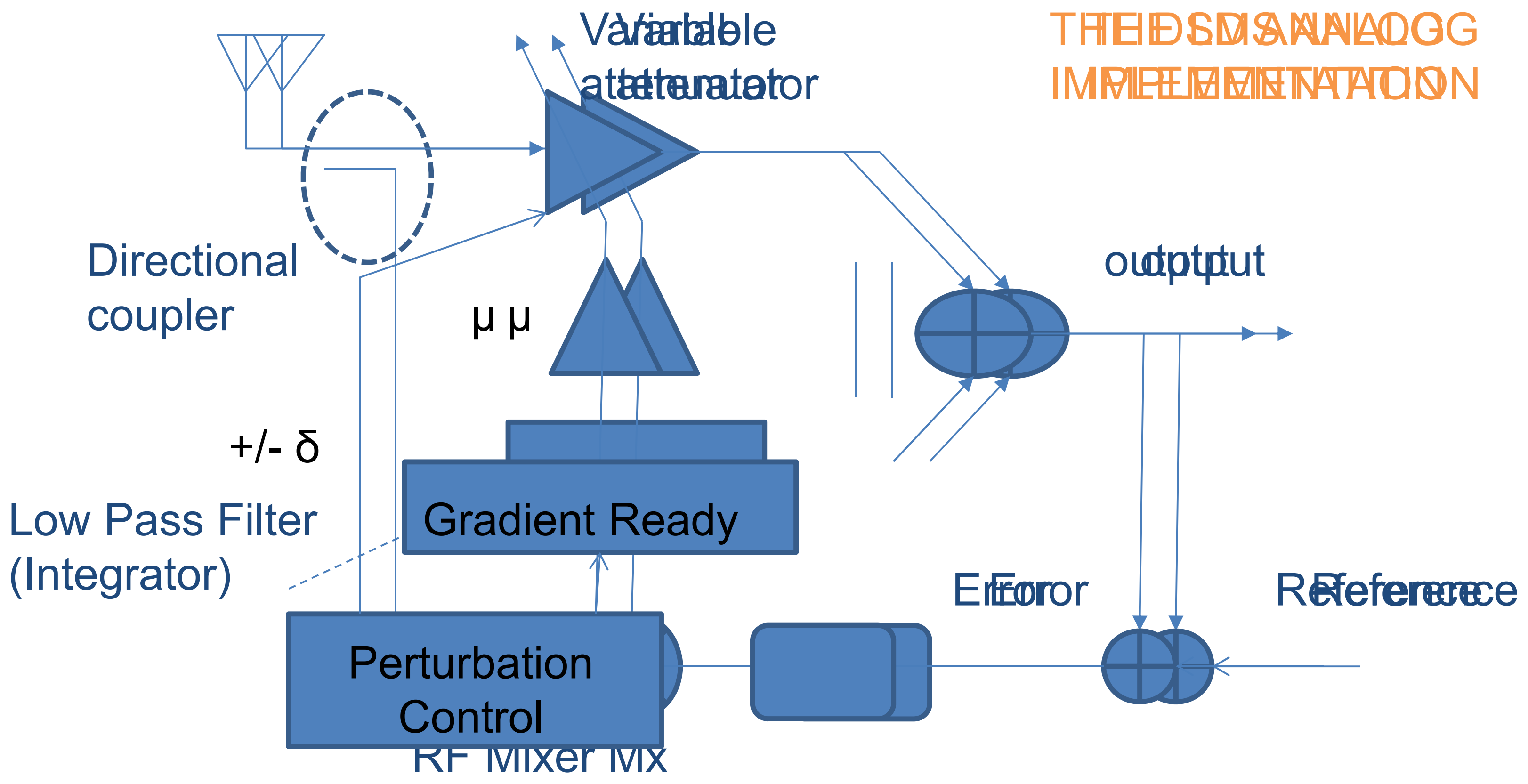
The goal is to estimate the gradient as accurate as posible and much better (less variance) than the instantaneous one.

The best estimate of the gradient we may construct, for weight q is:

$$\frac{\partial\xi}{\partial a_m(q)} = \frac{\xi\left(a_m(q)+\delta\right) - \xi\left(a_m(q)-\delta\right)}{2\delta}$$

Let us study this estimate

- Note that DSD is based on perturbation thus there is not RF processing like in the LMS, i.e. DSD is fully implemented at baseband.



The variance of the estimate

$$\xi(\pm\delta)=\frac{1}{K}\sum_{m=1}^{K}\left|\varepsilon(m)\right|^2 \Rightarrow E\left(\left|\xi(\pm\delta)\right|^2\right)=\frac{1}{K^2}\sum\sum E\left(\left|\varepsilon(m)\right|^2\left|\varepsilon(s)\right|^2\right)$$

since $E\left(\left|\varepsilon(m)\right|^2\left|\varepsilon(s)\right|^2\right)=\xi_{\min}^2$ and $E\left(\left|\xi(\pm\delta)\right|^2\right)=\frac{\xi_{\min}^2}{K}$

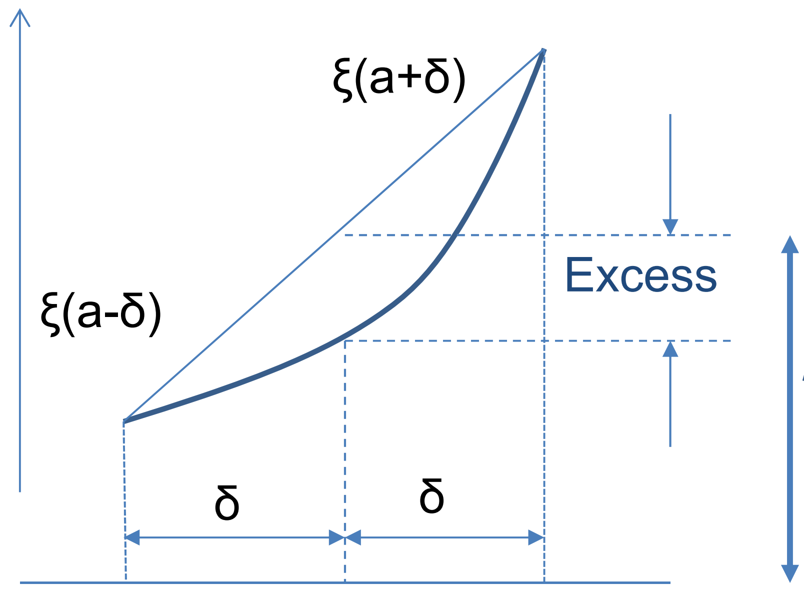
In summary:

$$E\left(\left(\frac{\partial\xi}{\partial a_m(q)}\right)\right)=E\left(\left(\frac{\xi(+\delta)-\xi(-\delta)}{2\delta}\right)^2\right)=\frac{1}{2\delta^2}E\left(\left|\xi(\pm\delta)\right|^2\right)=\frac{\xi_{\min}^2}{2\delta^2K}$$

Clearly, the estimate improves for small perturbations and large samples in the average. We use the minimum error in the above formula, since the derived variance will be relevant for miss-adjustment purposes which occurs in convergence

THE PERTURBATION ERROR

In order to do not duplicate the baseband processing and the aperture, note that any weight does not stay never on its value a(q), it is all-time on perturbation, This fact, promotes and excess of error that is denoted perturbation error



To compute this excess of error we use the Taylor's expansion of the error

$$\frac{\xi(a+\delta)+\xi(a-\delta)}{2}-\xi(a)$$

$$\cong \frac{\delta^2}{2}\frac{d^2\xi}{da(q)^2}=\frac{\delta^2 r(q)}{2}$$

Note that the second derivative of the gradient with respect a weight is the power at the corresponding antenna, i.e. the qq input of the snapshots covariance.

The previous formula is just the perturbation error due to the estimate of the gradient for a single weight. At the next perturbation the coefficient changes so the perturbation does. Along a complete cycle of Q weights or antennas we will use the average.

$$\frac{\delta^2 tr(R)}{2Q}$$

Normalized by the minimum error, The perturbation miss-adjutsment is

$$P = \frac{\delta^2 tr(R)}{2Q\xi_{\min}}$$

Note that regardless the number of up-dates for convergence is the same, each computation of the gradient consumes…..

$$\left.\begin{array}{l} i-q\ components\ \ldots\ldots\ldots\ \ldots 2 \\ \text{Number of antennas.. .... Q} \\ \text{Number of samples... ..... K} \\ \text{2 perturbati ons/antenn a...2} \end{array}\right\} \text{Global for update 4QK snapshots}$$

The DSD miss-adjustment:

$$\underline{A}_{opt} = \underline{A}_{opt}$$

$$\underline{A}^{(m+1)} = \underline{A}^{(m)} - \mu \underline{\nabla}^{m}$$

$$\underline{\tilde{A}}^{m+1} = \underline{\tilde{A}}^{m} + \mu \underline{\nabla}$$

being

$$\underline{\tilde{A}}^{m+1} = \underline{\tilde{A}}^{m} + \mu \underline{\nabla} + \mu \underline{\tilde{\nabla}}$$

Actual gradient

Gradient error (Uncorrelated) with variance already computed previously

$$\frac{\xi_{\min}^{2}}{2\delta^{2}K} \underline{\underline{I}}$$

$$\underline{\nabla} = \underline{\underline{R}}\,\underline{A}_m - \underline{P} = -\underline{\underline{R}}\,\underline{\tilde{A}}_m$$

Thus, the covariance update is:

$$\underline{\underline{\Sigma}}^{m+1} = \underline{\underline{\Sigma}}^{m} - 2\mu \underline{\underline{R}}\,\underline{\underline{\Sigma}}^{m} + \frac{\mu^{2}\xi_{\min}^{2}}{2\delta^{2}K} \underline{\underline{I}}$$

This term is neglected for small steep size

Now in convergence, the covariance matrix does not change, so the final value of it will be found from setting

$$\underline{\underline{\Sigma}}^{m+1} = \underline{\underline{\Sigma}}^{m} = \underline{\underline{\Sigma}}$$

The resulting covariance is:

$$\underline{\underline{\Sigma}} = \frac{\mu \xi_{\min}^2}{4\delta^2 K} \underline{\underline{R}}^{-1}$$

Note here the major difference of other gradient algorithms. As DSD does, they do not allow free search of coefficients after convergence is achieved, i.e. the coordination of coefficients remains at convergence. This severely bounds the performance on time-varying scenarios.

Finally the miss-adjustment error is:

$$M_E = \frac{tr\left[\underline{\underline{\Sigma}}_n \underline{\underline{R}}\right]}{\xi_{\min}} = \frac{\mu Q \xi_{\min}}{4\delta^2 K}$$

The total miss-adjustment will be the sum of the miss-adjustment error plus the perturbation error

$$M = M_E + P = \frac{\mu Q \xi_{\min}}{4\delta^2 K} + \frac{\delta^2 tr(R)}{2Q\xi_{\min}}$$

Note that:

$$M = \frac{a}{x} + bx \quad \text{where} \quad a = \frac{\mu Q \xi_{\min}}{4K}, b = \frac{tr(R)}{2Q\xi_{\min}}, x = \delta^2$$

The optimum perturbation is:

$$\mu = \frac{2\alpha}{tr(R)}$$

$$x = \sqrt{\frac{a}{b}} \Rightarrow \delta^2 = \sqrt{\frac{\mu Q^2 \xi_{\min}^2}{2Ktr(R)}} = \boxed{Q\xi_{\min}\sqrt{\frac{\alpha}{K}}}$$

And, the optimum global miss-adjustment is:

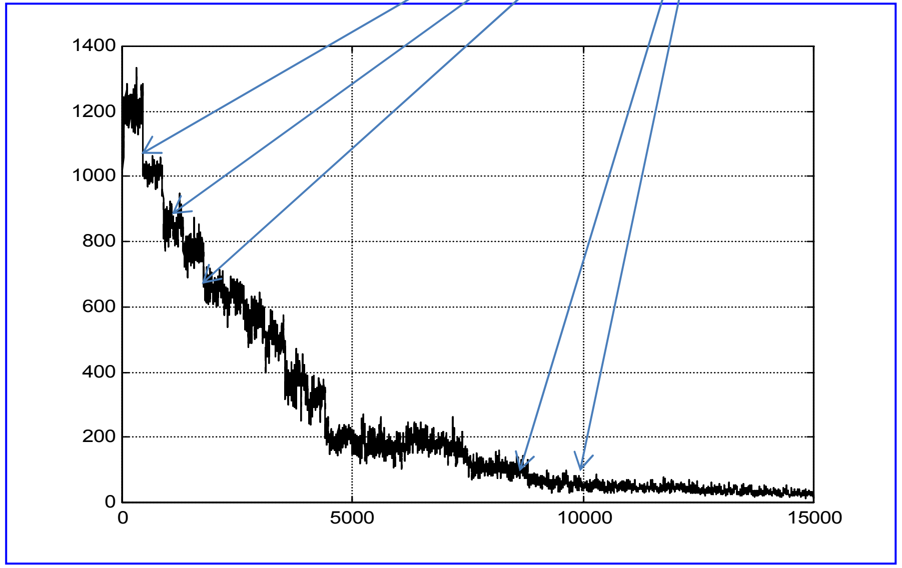
$$\boxed{M_{opt} = 2\sqrt{ab} = \sqrt{\frac{\alpha Q}{K}}}$$

DSD Performance

DSD Design:

- α from the convergence rate
- K from the desired missadjustment
- δ from its desig rule to make equal the perturbation error and the miss-adjustment error.

Convergence after 10 iterations BUT still behaves more slowly than the LMS



The major advantage of the DSD versus LMS is the use in coorperartive or distributed beamforming in wireless sensor networks

Random Search Methods LRS

The linear random search LRS algorithm is inspired on the DSD but with a single perturbation for all the weights.
In a distributed beamforming scenario, the LRS captures the increment on the error from a single random perturbation vector $\underline{\delta}$

The error produced with the weight and the weight perturbed are:

$$\underline{A}^n \Rightarrow \xi(n)$$

$$\underline{A}^n + \underline{\delta} \Rightarrow \xi_+(n)$$

Note that all the components are perturbed simultaneously

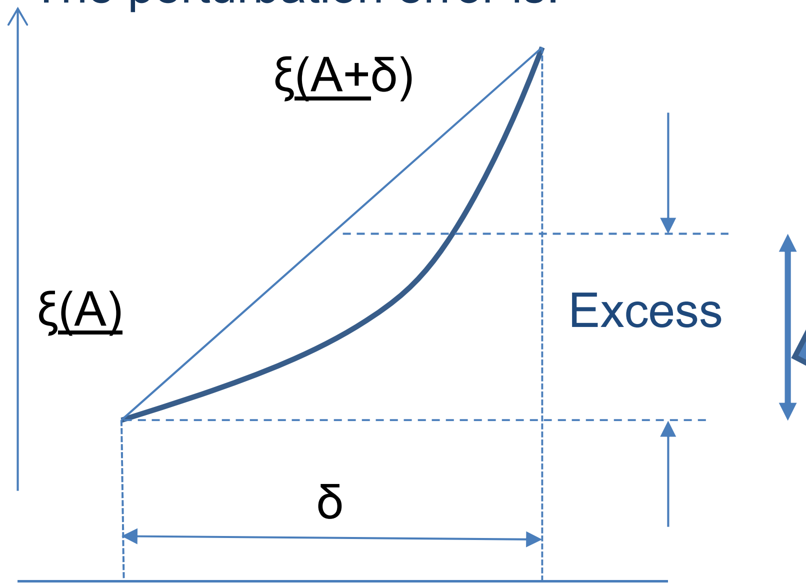
The perturbation is selected random and uncorrelated with zero mean and covariance σ^2

The learning rule is:

$$\underline{A}^{n+1} = \underline{A}^n + \mu \Delta_n \underline{\delta}$$

$$\Delta_n = \xi(n) - \xi_+(n)$$

The perturbation error is:



To compute this excess of error we use
the Taylor's expansion of the error

$$\frac{\xi(\underline{A}_n+\underline{\delta})+\xi(\underline{A}_n)}{2}-\xi(\underline{A}_n)$$

$$=\frac{1}{2}tr\left(\underline{\delta}^H \underline{\underline{R}}\underline{\delta}\right)=\frac{\sigma^2 tr\left(\underline{\underline{R}}\right)}{2}$$

Excess of error for a coefficients error δ

In consequence:

$$M_P=\frac{\sigma^2 tr(R)}{2\xi_{\min}}$$

The variance of $\underline{\Delta}_n$ is:

$$\Delta_n = \xi(n) - \xi_+(n) = \frac{1}{K}\left[\sum_m \left|e_+(n-m)\right|^2 - \sum_m \left|e_+(n-m)\right|^2\right]$$

Thus

$$\operatorname{var}^2\left(\Delta_n \Delta_n^*\right) = \frac{2\xi_{\min}^2}{K}$$

and

$$E\left(\Delta_n \underline{\tilde{A}}_n^H\right) = \underline{\delta}^H \underline{\underline{R\Sigma}}_n$$

since

$$\underline{\tilde{A}}_{n+1} = \underline{\tilde{A}}_n - \mu \Delta_n \underline{\delta}$$

We can compute the covariance evolution of the LRS

$$\underline{\tilde{A}}_{n+1} = \underline{\tilde{A}}_n - \mu\Delta_n \underline{\delta}$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{\Sigma}}_n + \mu^2\sigma^2 \underline{\underline{I}} \frac{2\xi_{\min}^2}{K} - 2\mu E\left(\underline{\delta}\Delta_n \underline{\tilde{A}}_n^H \right) =$$

$$= \underline{\underline{\Sigma}}_n + \mu^2\sigma^2 \underline{\underline{I}} \frac{2\xi_{\min}^2}{K} - 2\mu E\left(\underline{\delta}\underline{\delta}^H \underline{\underline{R}}\underline{\tilde{A}}_n \underline{\tilde{A}}_n^H \right) =$$

$$= \underline{\underline{\Sigma}}_n + \mu^2\sigma^2 \underline{\underline{I}} \frac{2\xi_{\min}^2}{K} - 2\mu\sigma^2 \underline{\underline{R}}\underline{\underline{\Sigma}}_n$$

Setting the stationary regime where the coefficients errors matrix stays the same after successive updates we have:

$$\frac{\mu}{2}\underline{\underline{R}}^{-1}\frac{2\xi_{\min}^2}{K} = \underline{\underline{\Sigma}}_n$$

$$\frac{\mu}{2}\underline{\underline{R}}^{-1}\frac{2\xi_{\min}^{2}}{K}=\underline{\underline{\Sigma}}_{n}$$

Again, note that LRS does not provide freedom to the coefficients at convergence, in fact he forces cooperation at this stage. This is an additional disadvantage again when comparing with the LMS

The miss-adjustment noise is:

$$M_N=\frac{tr\left(\underline{\underline{\Sigma}}_{n}\underline{\underline{R}}\right)}{\xi_{\min}}=\frac{\mu Q\xi_{\min}}{K}$$

And the total miss-adjustment is the sum of the above plus the perturbation missadjustment.

$$M=M_N+M_P=\frac{\mu Q\xi_{\min}}{K}+M_P$$

$$\mu=\frac{2\alpha}{\sigma^{2}tr\left(R\right)}\quad and\quad M_P=\frac{\sigma^{2}tr(R)}{2\xi_{\min}}$$

$$M=\frac{\alpha Q}{KM_P}+M_P \longleftarrow$$

An optimum exists for the perturbation missadjustment

The optimum perturbation error is $M_P^{opt} = \sqrt{\frac{\alpha Q}{K}}$

$$M_P = M_P^{opt} \Rightarrow \sqrt{\frac{\alpha Q}{K}} = \frac{\sigma^2 tr(R)}{2\xi_{\min}}$$

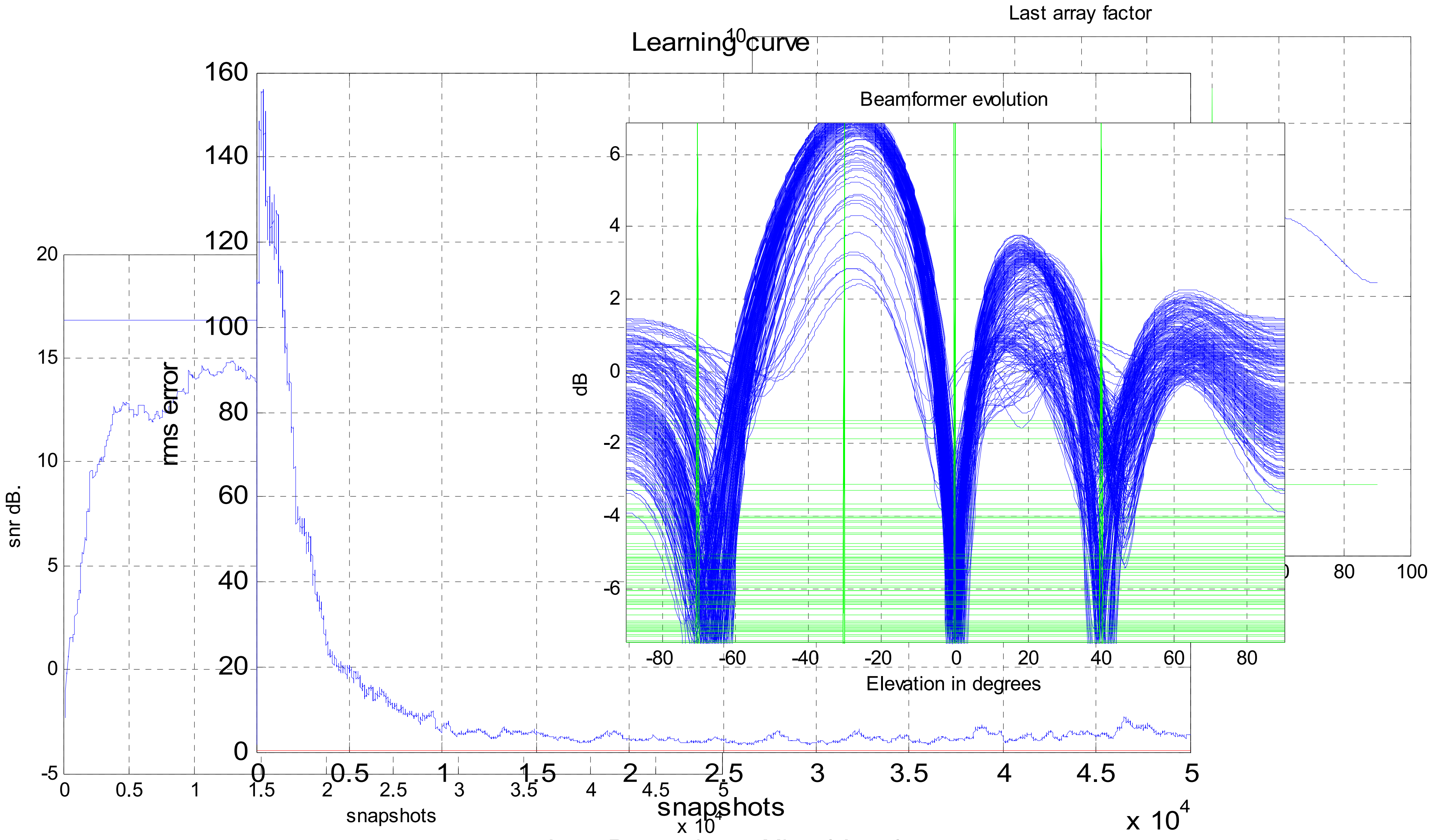
Obtain α from the desired convergence rate

Find σ² from the above equation and K from the desired missadjustment

$$M^{opt} = 2\sqrt{\frac{\alpha Q}{K}}$$

The optimum design is when the miss-adjustment error is equal to the perturbation missadjustment, or the global is just twice any one of them

Performance of the LRS



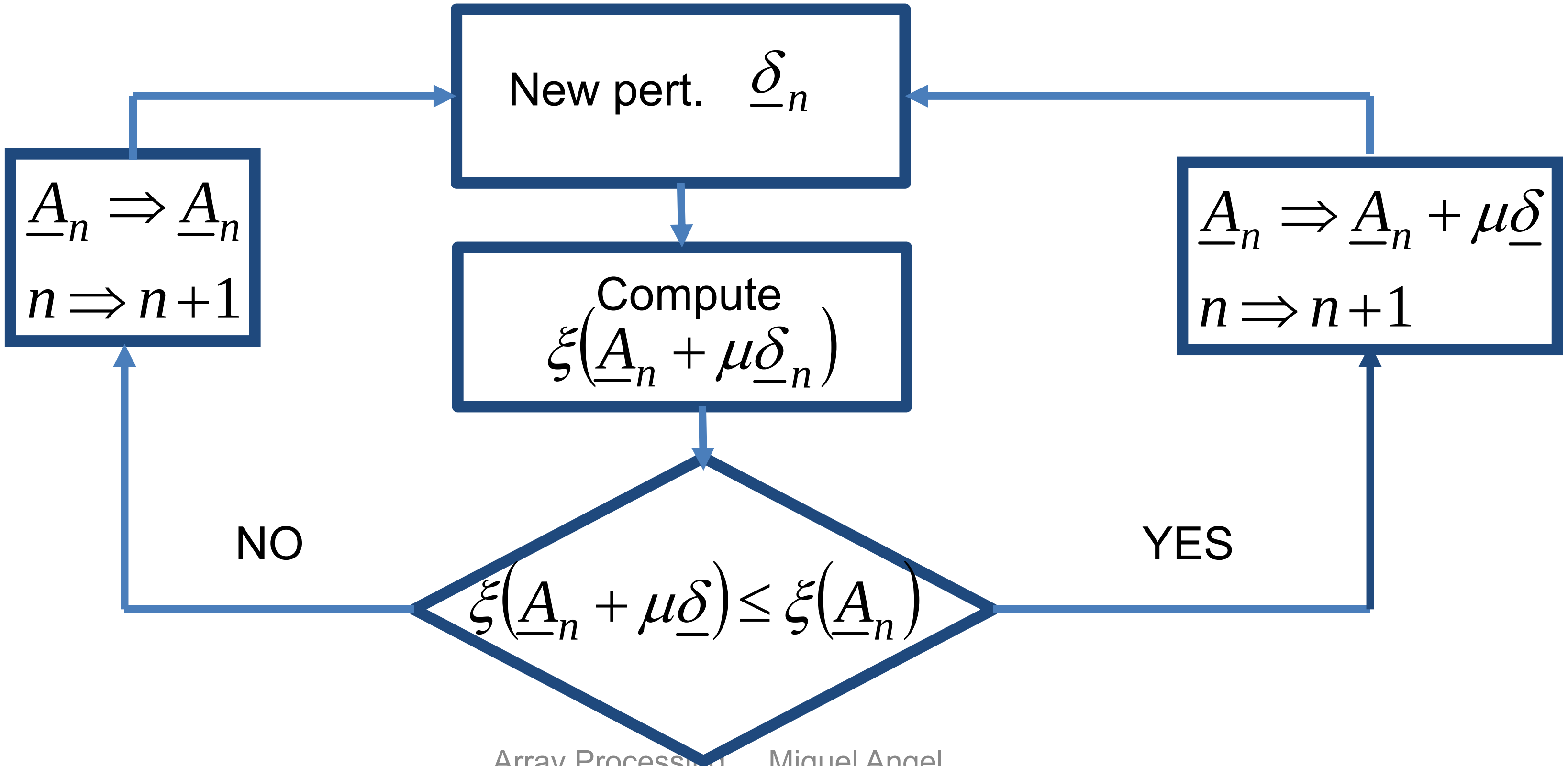
Random Search Algorithms: RS

Random Search refers to those algorithm that use random perturbation in order to decrease their errors.

A vector perturbation $\underline{\delta}_n$ is used over the current weights, with variance equal to one as:

$$\underline{\delta} = \cos(\theta) - j\sin(\theta)$$

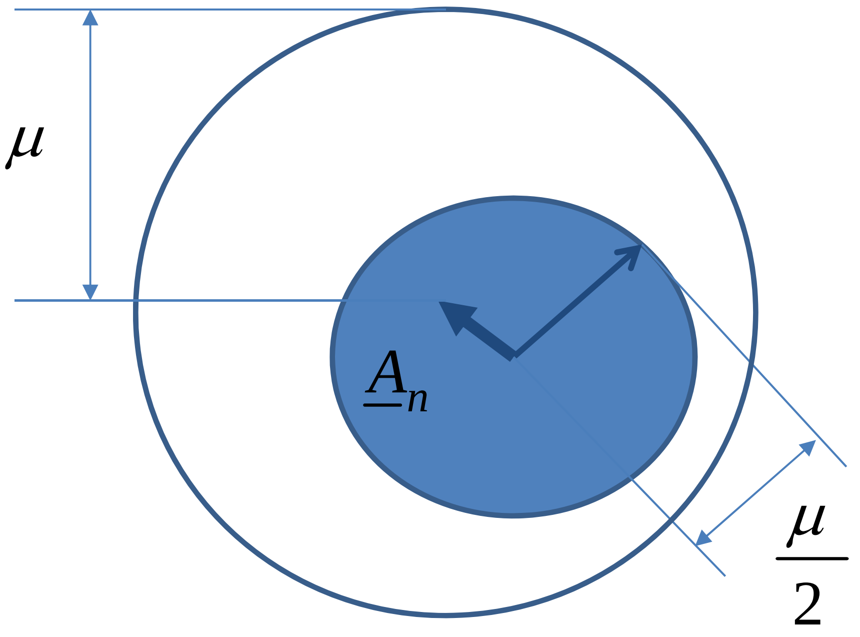
Where Θ is a random variable with uniform distributed between $0, 2\pi$



New pert. $\underline{\delta}_n$
Compute $\xi\left(\underline{A}_n + \mu\underline{\delta}_n\right)$
$\xi\left(\underline{A}_n + \mu\underline{\delta}\right) \leq \xi\left(\underline{A}_n\right)$
NO
YES
$\underline{A}_n \Rightarrow \underline{A}_n$
$n \Rightarrow n+1$
$\underline{A}_n \Rightarrow \underline{A}_n + \mu\underline{\delta}$
$n \Rightarrow n+1$

Random search algorithm always converge since they just consolidate good moves in the error surface.

In order to have an idea of the miss-adjustment of RS algorithm, note that when the beamformer is within a circle of radius μ/2, there is no further improvement from a perturbation of size μ



In consequence:

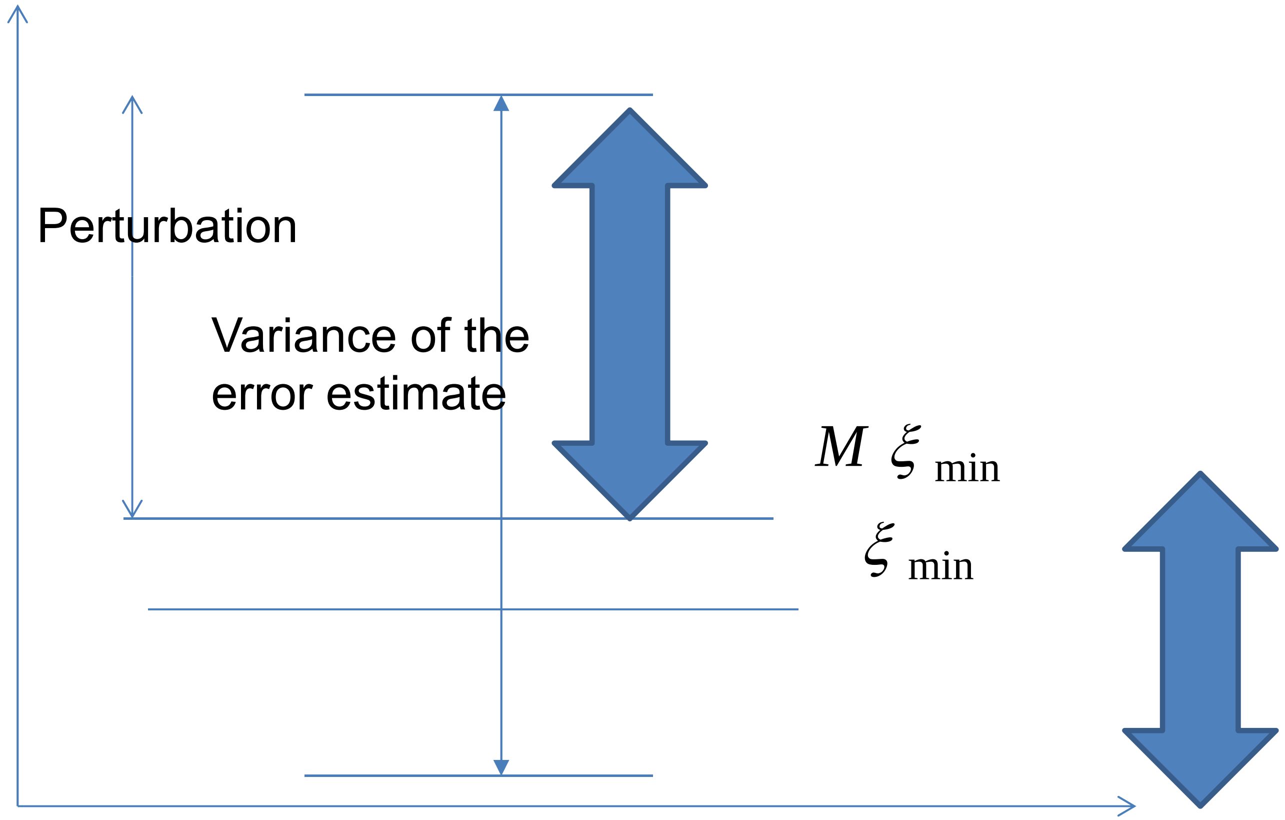
$$\underline{\underline{\Sigma}}_n = \frac{9\mu^2}{8}\underline{\underline{I}}$$

and

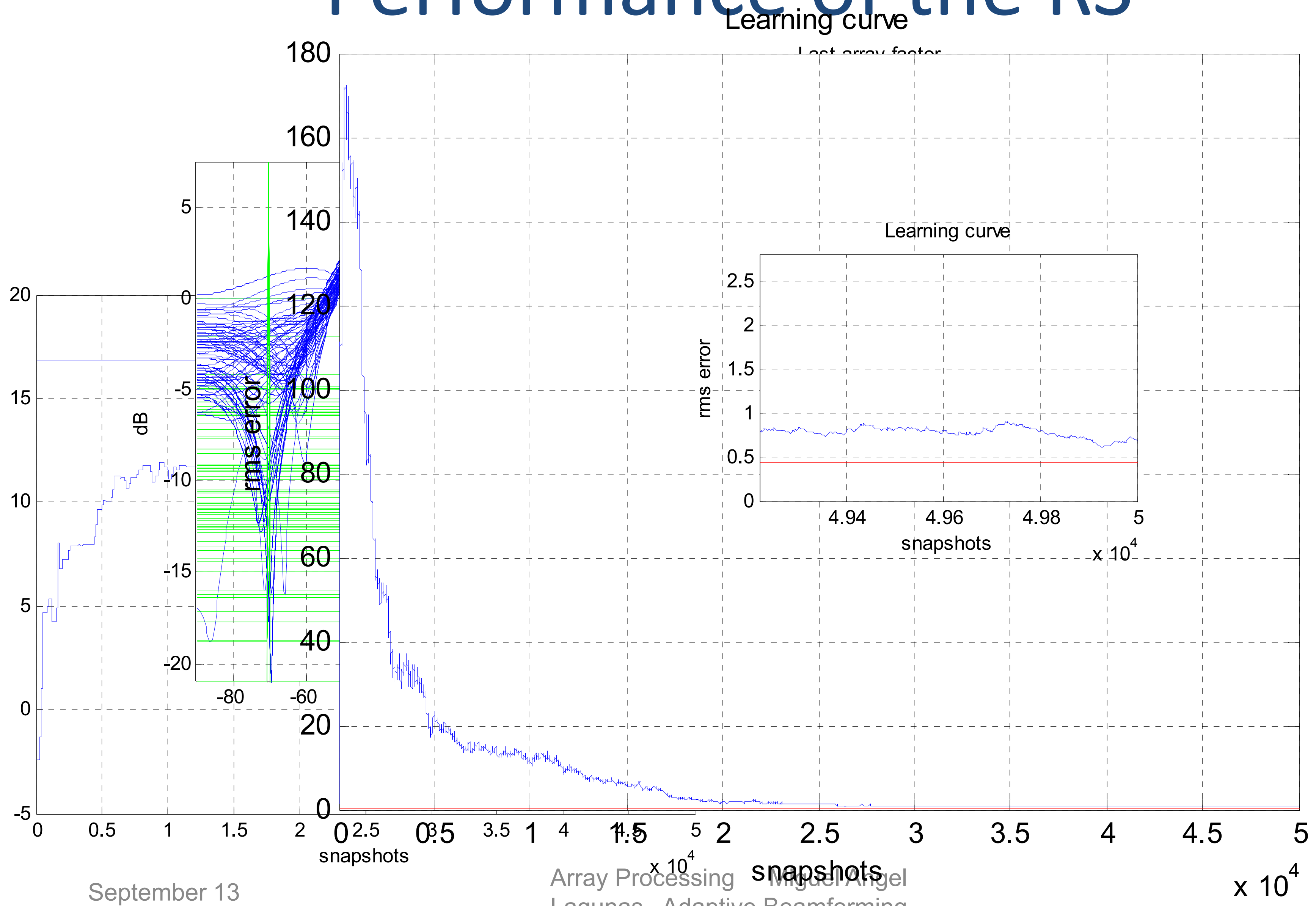
$$M = \frac{9\mu^2 tr(R)}{8\xi_{\min}}$$

In addition, RS algorithms need to guarantee that the miss-adjustment stays below the variance of the error measurement performed by K samples. This implies that:

$$\xi_{\min} \leq \frac{9\mu^2 tr(R)}{8} \leq \frac{2\xi_{\min}}{\sqrt{K}}$$

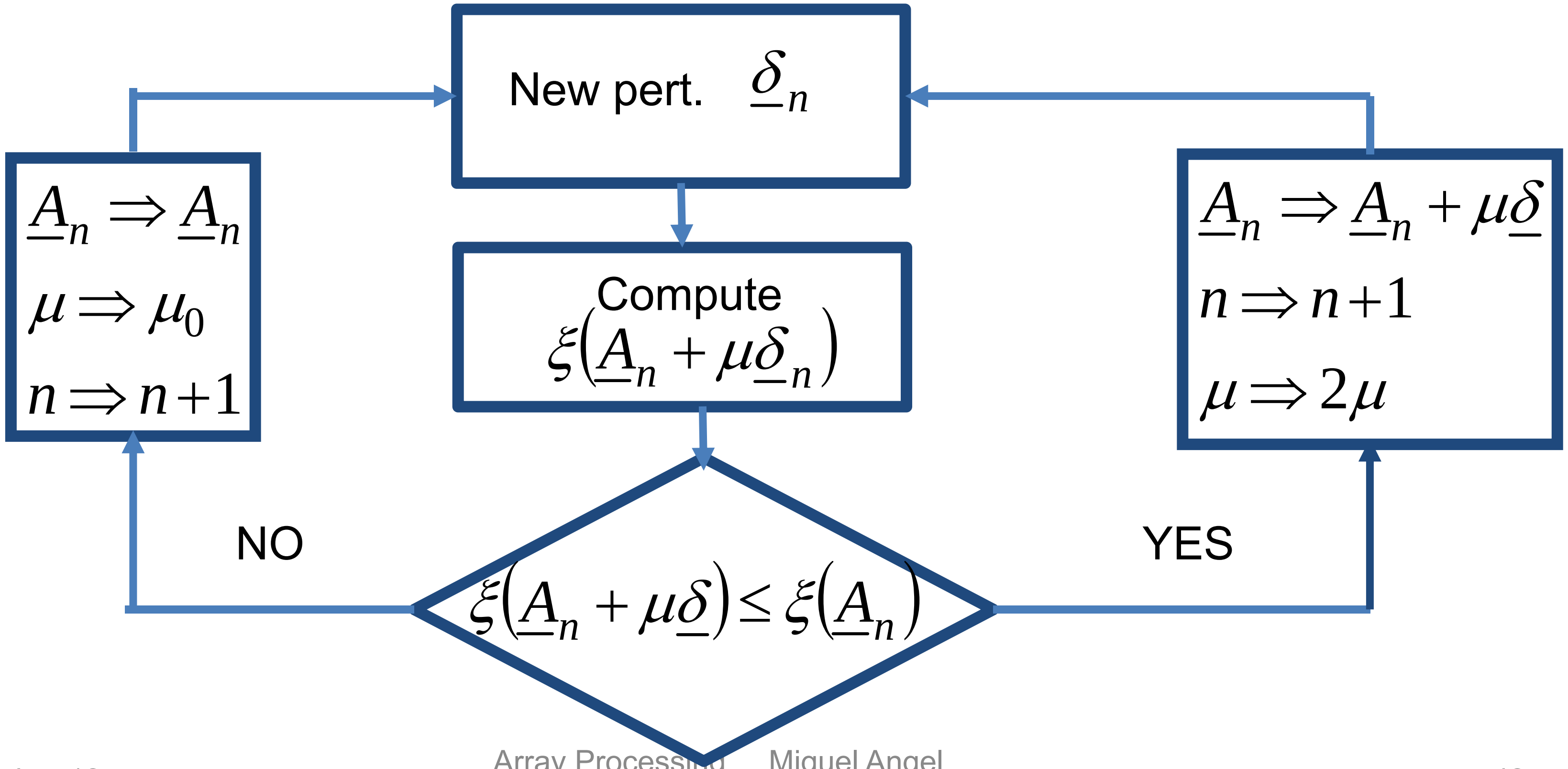


Performance of the RS

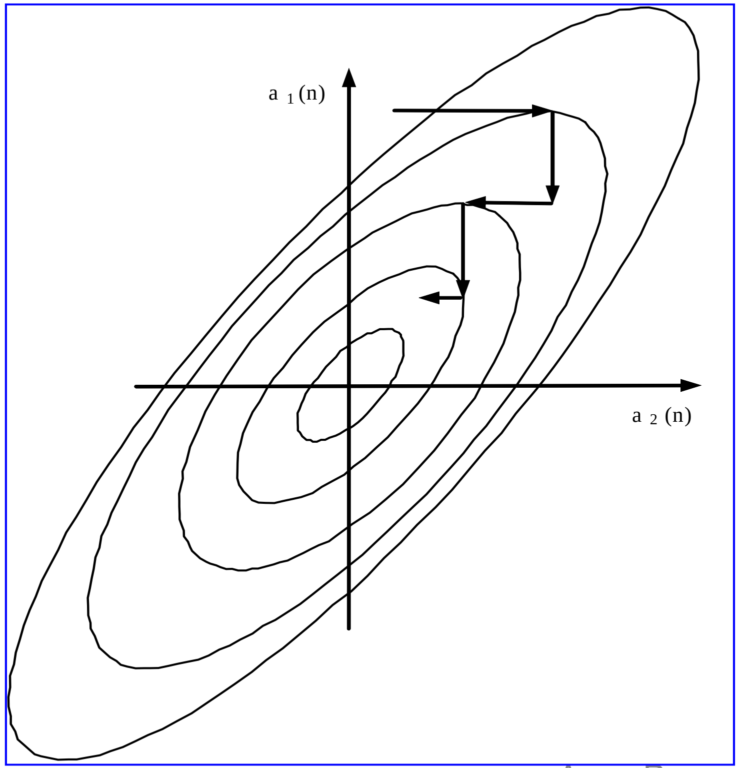


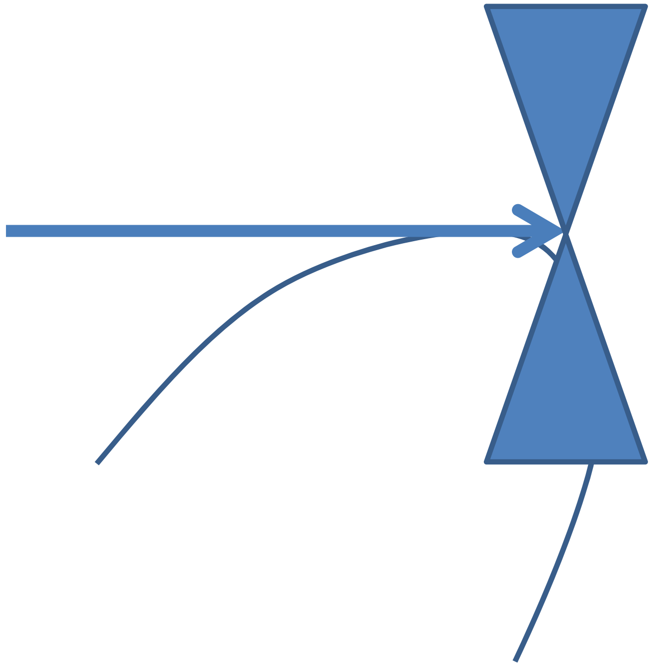
Accelerated and Guided RS: ARS and GARS

The ARS accelerates μ (uses 2 times the previous steep-size) when a good move in the surface error is discovered. For a bad move the steep size is reset to a minimum value


New pert. $\underline{\delta}_n$
Compute $\xi\left(\underline{A}_n + \mu\underline{\delta}_n\right)$
$\xi\left(\underline{A}_n + \mu\underline{\delta}\right) \leq \xi\left(\underline{A}_n\right)$
NO
YES
$\underline{A}_n \Rightarrow \underline{A}_n$
$\mu \Rightarrow \mu_0$
$n \Rightarrow n+1$
$\underline{A}_n \Rightarrow \underline{A}_n + \mu\underline{\delta}$
$n \Rightarrow n+1$
$\mu \Rightarrow 2\mu$

Extra guidance GARS can be provided to the ARS: For example, when the move on a given direction is fully exploited and a bad move is sensed, the ARS ask for a new random perturbation. A guided version, before this option exploits that the last point stays close to the tangent point with the limit surface of error, in consequence is good to select the angle in the perturbation as 90º or -90º. If this guide fails then the GARS resort to the random perturbation again.



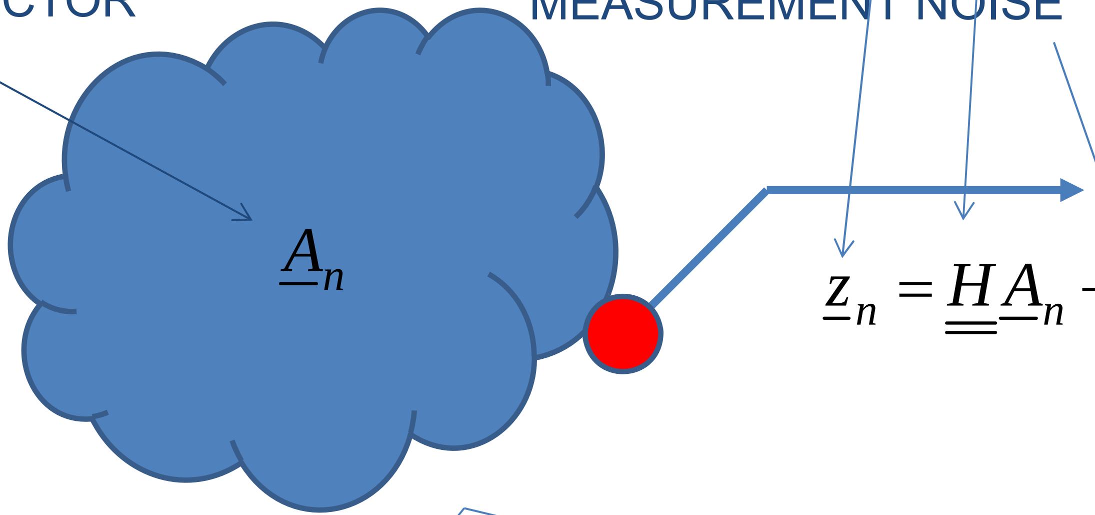


The Kalman Filter

The Gauss Model

We characterize the "world" we like to observe and track by means an STATE VECTOR

We have an OBSERVATION the state from outside trough an OBSERVATION MATRIX and with a MEASUREMENT NOISE



$$\underline{z}_n = \underline{\underline{H}}\,\underline{A}_n + \underline{w}_n$$

Any differential equation between an input and an output can be written as a STATE EQUATION

$$\underline{A}_{n+1} = \underline{\underline{F}}\,\underline{A}_n + \underline{v}_n$$

This equation is described by the TRANSITION MATRIZ and the INNOVATION

THE MEASUREMENT EQUATION

$$\underline{z}_n = \underline{\underline{H}}\,\underline{A}_n + \underline{w}_n$$

The observation vector, this is the only available information that we have from the system and it will be the main input for the algorithm

The state vector is the target of the algorithm. We aim to estimate it as accurate as possible

The measurement noise. It is assumed independent of the state and distributed Gaussian with zero mean and variance matrix equal to L

What is know by us from this equation:
- The observation vector
- The observation matrix H
- The covariance of the noise L

THE STATE EQUATION

$$\underline{A}_{n+1} = \underline{\underline{F}}\,\underline{A}_n + \underline{v}_n$$

The transition matrix is given by KNOWN relation between state variables. Full IGNORANCE is reflected by setting this matrix equal to the identity matrix.

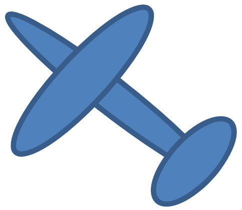
The innovation vector is a Gaussian vector with zero mean and covariance matrix V. It represents the un-predicted evolution of the state. When the ignorance about the observation matrix is high, the diagonals of V have to be high

What is known from this equation: - The transition matrix
- The innovation covariance matrix

Some examples: Radar and Sonar

The state vector to be estimated at every n

$$\underline{A}_n = \begin{bmatrix} x_n \\ y_n \\ v_{xn} \\ v_{yn} \end{bmatrix}$$



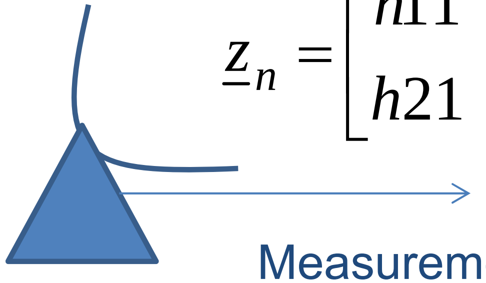
Non-accelerated movement.

$$\left(x_n, y_n, v_{xn}, v_{yn}\right)$$

Actual position and velocity of the target

Note that not all the components to be tracked are necessarily observed at the measurement

$$\underline{z}_n = \begin{bmatrix} h11 & h12 & \boxed{\begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}} \\ h21 & h22 & \end{bmatrix} \underline{A}_n + \underline{w}_n$$



Measurement

Non accelerated movement

Maneuver capabilities of the target for unpredicted acceleration reflected on the variance of innovation

$$\underline{A}_{n+1} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underline{A}_n + \underline{v}_n$$

Time elapsed from two returns

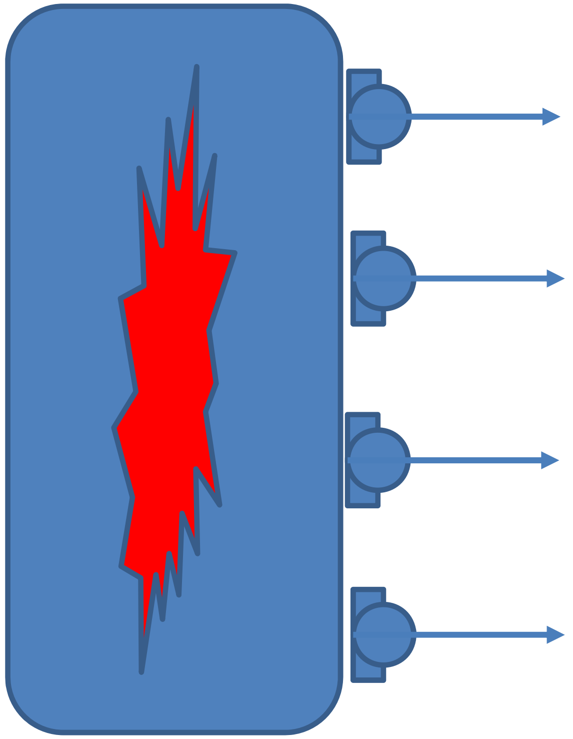
NON INTRUSSIVE MEASUREMENTS

$$\underline{A}_n = \begin{bmatrix} T1_n \\ T2_n \\ T3_n \\ T4_n \end{bmatrix}$$

The state vector are temperatures at equally spaced levels. Also the velocity of changed can be included.

$$\underline{A}_{n+1} = \underline{\underline{F}}\,\underline{A}_n + \underline{v}_n$$

Differential equation describing heat evolution inside, predicted and unpredicted contributions

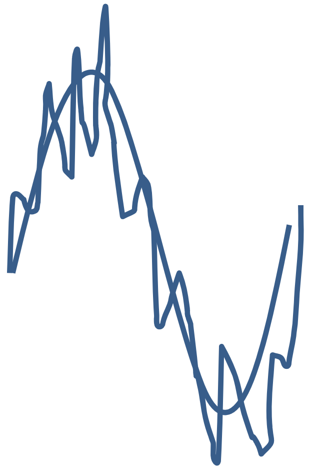


Measurement vector of a number of components usually greater than the number of states to track.

$$\underline{z}_n = \underline{\underline{H}}\,\underline{A}_n + \underline{w}_n$$

The observation matrix describes heat propagation from the internal layers to the external cover

Non-linear measurement: PLL



Single carrier i-q components with un-know phase and frequency

The state vector includes the instantaneous phase and the instantaneous frequency.

$$\underline{A}_n = \begin{bmatrix} \theta(n) \\ f(n) \end{bmatrix}$$

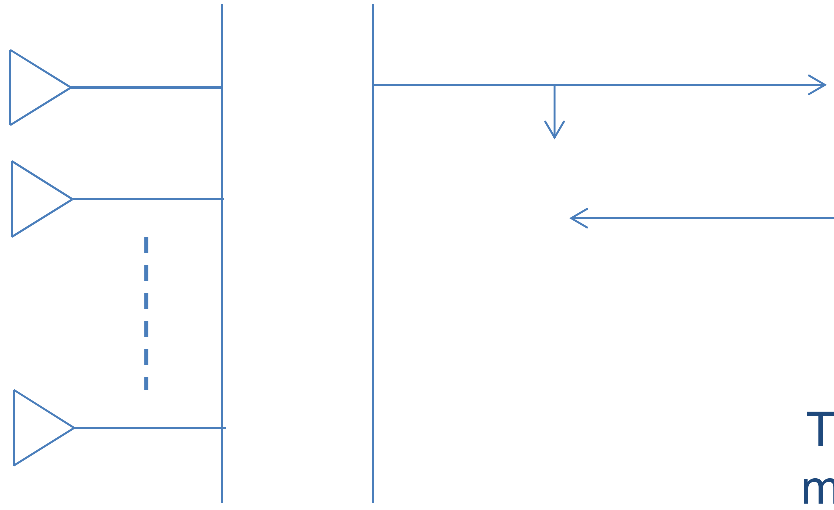
$$\underline{A}_{n+1} = \begin{bmatrix} 1 & 2\pi T \\ 0 & 1 \end{bmatrix} \underline{A}_n + \underline{v}_n$$

Innovation includes possible jerk from the Doppler frequency

$$\underline{z}_n = A \begin{bmatrix} \cos(\theta(n)) \\ -\sin(\theta(n)) \end{bmatrix} + \underline{w}_n$$

Non-linear measurement, i.e. the i-q components of the received signal. Note that frequency is not observed

Beamforming



Observation (Reference)

$$\underline{z}_n^* = \underline{X}_n^H \underline{A}_n + \varepsilon_{\min,n}^*$$

The observation matrix (vector) is the current snapshot.

The state vector to estimate is the optimum beamformer

The observation noise is the minimum error which is orthogonal to the snapshot

Full ignorance about the evolution of the optimum beamformer

$$\underline{A}_{n+1} = \underline{A}_n + \underline{v}_n$$

The Kalman's Algorithm

GAUSS MODEL

$$\underline{A}_{n+1} = \underline{\underline{F}}\,\underline{A}_n + \underline{v}_n$$

$$\underline{z}_n = \underline{\underline{H}}\,\underline{A}_n + \underline{w}_n$$

Starting with an initial state vector (it could be the zero vector) and the corresponding (high) ignorance

$$\underline{\hat{A}}_0 \quad \underline{\underline{\Sigma}}_0 \qquad \text{where } \underline{\underline{\Sigma}}_n = E\left(\underline{\tilde{A}}_n \underline{\tilde{A}}_n^H\right)$$

$$\text{being} \quad \underline{\tilde{A}}_n = \underline{A}_n - \underline{\hat{A}}_n$$

The algorithm mimics the model with two equation in order to track the state vector, the two equations are:

$$\underline{\hat{z}}_n = \underline{\underline{H}}\,\underline{\hat{A}}_n$$

$$\underline{\hat{A}}_{n+1} = \underline{\underline{F}}\,\underline{\hat{A}}_n + \underline{\underline{K}}_n \underline{\varepsilon}_n$$

where $\underline{\underline{K}}_n$ Is the so-called GAIN MATRIX

and $\underline{\varepsilon}_n$ Is the model error

THE MODEL ERROR AND ITS COVARIANCE

$$\underline{\varepsilon}_n = \underline{z}_n - \underline{\hat{z}}_n = \underline{\underline{H}}\left(\underline{A}_n - \underline{\hat{A}}_n\right) + \underline{w}_n$$

This equation relates the system error with the state vector

$$\underline{\varepsilon}_n = \underline{\underline{H}}\,\underline{\tilde{A}}_n + \underline{w}_n$$

Thus, the covariance of the system error is: This equation reveals that the system error is due to the state error plus the measurement error.

$$\underline{\underline{E}}_n = \underline{\underline{H}}\,\underline{\underline{\Sigma}}_n\,\underline{\underline{H}}^H + \underline{\underline{L}}$$

PROGATION OF THE STATE ERROR

$$\underline{A}_{n+1} = \underline{\underline{F}}\,\underline{A}_n + \underline{v}_n$$

$$\underline{\hat{A}}_{n+1} = \underline{\underline{F}}\,\underline{\hat{A}}_n + \underline{\underline{K}}_n\,\underline{\varepsilon}_n$$

Subtracting these two equations we have the propagation of the state error vector

$$\underline{\tilde{A}}_{n+1} = \underline{\underline{F}}\,\underline{\tilde{A}}_n + \underline{v}_n - \underline{\underline{K}}_n\,\underline{\varepsilon}_n$$

THE ORTHOGONALITY PRINCIPLE

Since the goal in the design of the gain matrix is to reduce as much as possible the covariance of the state error, we apply the orthogonality principle between the error and the data used to minimize this error:

$$E\left(\underline{\tilde{A}}_{n+1}\underline{\varepsilon}_n^H\right) = \underline{\underline{0}} \quad \Rightarrow \quad \underline{\underline{0}} = \underline{\underline{F}}E\left(\underline{\tilde{A}}_n\underline{\varepsilon}_n^H\right) - \underline{\underline{K}}_n E\left(\underline{\varepsilon}_n\underline{\varepsilon}_n^H\right)$$

$$\underline{\underline{K}}_n = \underline{\underline{F}}\,\boxed{E\left(\underline{\tilde{A}}_n\underline{\varepsilon}_n^H\right)}\,\underline{\underline{E}}_n^{-1}$$

Using the equation of the system error in terms of the state vector, we have:

$$E\left(\underline{\tilde{A}}_n \underline{\varepsilon}_n^H\right) = E\left(\underline{\tilde{A}}_n \left[\underline{\tilde{A}}_n^H \underline{\underline{H}} + \underline{w}_n\right]\right) = \underline{\underline{\Sigma}}_n \underline{\underline{H}}$$

In summary,

$$\underline{\underline{K}}_n = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n \underline{\underline{H}}^H \underline{\underline{E}}_n^{-1}$$

$$\underline{\underline{E}}_n = \underline{\underline{H}}\,\underline{\underline{\Sigma}}_n \underline{\underline{H}}^H + \underline{\underline{L}}$$

In order to complete the algorithm iteration we need to propagate the state error covariance.

PROPAGATION OF THE STATE ERROR COVARIANCE

Using $$\underline{\tilde{A}}_{n+1} = \underline{\underline{F}}\,\underline{\tilde{A}}_n + \underline{v}_n - \underline{\underline{K}}_n \underline{\varepsilon}_n$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n \underline{\underline{F}}^H + \underline{\underline{V}} + \boxed{\underline{\underline{K}}_n \underline{\underline{E}}_n \underline{\underline{K}}_n^H - \underline{\underline{F}} E\left(\underline{\tilde{A}}_n \underline{\varepsilon}_n^H\right) \underline{\underline{K}}_n^H} - \underline{\underline{K}}_n E\left(\underline{\varepsilon}_n \underline{\tilde{A}}_n^H\right) \underline{\underline{F}}^H$$

This term is zero due to the orthogonality principle

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n \underline{\underline{F}}^H - \underline{\underline{K}}_n \underline{\underline{E}}_n \underline{\underline{K}}_n^H + \underline{\underline{V}}$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n \left(\underline{\underline{I}} - \underline{\underline{H}}\,\underline{\underline{E}}_n^{-1} \underline{\underline{H}}^H \underline{\underline{\Sigma}}_n\right) \underline{\underline{F}}^H + \underline{\underline{V}}$$

$$\boxed{\begin{array}{c} n=0 \\ \hat{\underline{A}}_0 \quad \underline{\underline{\Sigma}}_0 \end{array}}$$

Summary

$$\underline{\underline{E}}_n = \underline{\underline{H}}\,\underline{\underline{\Sigma}}_n\,\underline{\underline{H}}^H + \underline{\underline{L}}$$

$$\underline{\underline{K}}_n = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n\,\underline{\underline{H}}^H\,\underline{\underline{E}}_n^{-1}$$

$$\underline{\varepsilon}_n = \underline{z}_n - \underline{\underline{H}}\,\hat{\underline{A}}_n$$

$$\hat{\underline{A}}_{n+1} = \underline{\underline{F}}\,\hat{\underline{A}}_n + \underline{\underline{K}}_n\,\underline{\varepsilon}_n$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n\,\underline{\underline{F}}^H - \underline{\underline{K}}_n\,\underline{\underline{E}}_n\,\underline{\underline{K}}_n^H + \underline{\underline{V}}$$

$$n \Rightarrow n+1$$

Adaptive beamforming with Kalman

$$\boxed{\begin{array}{c} \underline{\hat{A}}_0 = \underline{0} \quad \underline{\underline{\Sigma}}_0 = 10^6 \underline{\underline{I}} \\ \\ \underline{\underline{V}} = v_0 \underline{\underline{I}} = 10^{-3} \underline{\underline{I}} \end{array}}$$

$$E_n = \underline{X}_n^H \underline{\underline{\Sigma}}_n \underline{X}_n + \xi_{\min}$$

$$\underline{K}_n = \underline{\underline{\Sigma}}_n \underline{X}_n / E$$

$$\varepsilon(n) = d(n)^* - \underline{X}_n^H \underline{\hat{A}}_n$$

$$\underline{\hat{A}}_{n+1} = \underline{\hat{A}}_n + \underline{K}_n \varepsilon(n)$$

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{\Sigma}}_n \left(\underline{\underline{I}} - \frac{\underline{X}_n \underline{X}_n^H \underline{\underline{\Sigma}}_n}{\xi_{\min} + \underline{X}_n^H \underline{\underline{\Sigma}}_n \underline{X}_n} \right)$$

$$n \Rightarrow n+1$$

The covariance propagation tends, when convergence, to the covariance of the innovation, in consequence the miss-adjustment is:

$$M = \frac{trace\left(\underline{\underline{V}}\,\underline{\underline{R}}\right)}{\xi_{\min}} = \frac{v_0 tr\left(\underline{\underline{R}}\right)}{\xi_{\min}}$$

Convergence: At the initial iteration we can assume that the state error covariance is diagonal

$$\underline{\underline{\Sigma}}_n \approx s_n^2 \underline{\underline{I}}_n \Rightarrow E_n = s_n^2 \underline{X}_n^H \underline{X}_n + \xi_{\min}$$

Small versus the first term

The new steep-size

$$\underline{\underline{\Sigma}}_{n+1} \approx \underline{\underline{\Sigma}}_n \left(\underline{\underline{I}} - \frac{\underline{X}_n \underline{X}_n^H s_n^2}{\xi_{\min} + s_n^2 \underline{X}_n^H \underline{X}_n} \right)$$

$$\underline{K}_n \approx \left(\frac{1}{\underline{X}_n \underline{X}_n^H} \right) \underline{X}_n$$

Thus, the diagonal terms of the error covariance evolve as:

$$\left(1-\frac{|x_n(q)|^2 s_n^2}{\xi_{\min}+s_n^2 \underline{X}_n^H \underline{X}_n}\right)$$

Assuming that the power received at each antenna is the same, and that the global power sensed by the aperture does not changes, this term is approximately equal to:

$$\left(1-\frac{|x_n(q)|^2 s_n^2}{\xi_{\min}+s_n^2 \underline{X}_n^H \underline{X}_n}\right) \approx \left(1-\frac{P_x s_n^2}{\xi_{\min}+s_n^2 Q P_x}\right) \approx \left(1-\frac{1}{Q}\right)$$

Convergence is achieved after two times Q (the number of antennas) iterations

$$\left(1-\frac{1}{Q}\right)^{n_c} = 0.1 \Rightarrow n_c = \frac{-2.3026}{\ln\left(1-Q^{-1}\right)} \approx 2.3Q$$

An adaptive steep-size (with a similarity with the LMS)

The maximum steep size for zero iterated error in LMS

$$\mu \approx \left(\frac{1}{\underline{X}_n \underline{X}_n^H} \right)$$

Small since miss-adjustment use to stay far below the minimum error in any practical design

$$\mu \approx \left(\frac{v_0}{\xi_{\min}} \right)$$

Non-linear measurement: Extended Kalman Algorithm

Example: For the PLL design, the measurement equation was non-linear. Nevertheless for small state error, the system error can be linearized using the first term of the Taylor's series of the non-linearity.

$$\underline{z}_n = A(n)\begin{bmatrix} \cos(\theta(n)) \\ -\sin(\theta(n)) \end{bmatrix} + \underline{w}_n \qquad \hat{\underline{z}}_n = \hat{A}(n)\begin{bmatrix} \cos(\hat{\theta}(n)) \\ -\sin(\hat{\theta}(n)) \end{bmatrix}$$

$$\underline{\varepsilon}_n = A(n)\begin{bmatrix} \left(\hat{A}(n)+\tilde{A}(n)\right)\cos\left(\hat{\theta}(n)+\tilde{\theta}(n)\right) - \hat{A}(n)\cos\left(\hat{\theta}(n)\right) \\ -\left(\hat{A}(n)+\tilde{A}(n)\right)\sin\left(\hat{\theta}(n)+\tilde{\theta}(n)\right) + \hat{A}(n)\sin\left(\hat{\theta}(n)\right) \end{bmatrix} + \underline{w}_n$$

Neglecting the products of errors after Taylor's first term approximation we have……………………

$$\underline{\varepsilon}_n = \begin{bmatrix} \tilde{A}(n)\cos\left(\hat{\theta}(n)\right) - \tilde{\theta}(n)\hat{A}(n)\sin\left(\hat{\theta}(n)\right) \\ -\tilde{A}(n)\sin\left(\hat{\theta}(n)\right) - \tilde{\theta}(n)\hat{A}(n)\cos\left(\hat{\theta}(n)\right) \end{bmatrix} + \underline{w}_n$$

This defines the observation matrix since it relates linearly the system error with the state error

$$\underline{\varepsilon}_n = \underbrace{\boxed{\begin{bmatrix} \cos\left(\hat{\theta}(n)\right) & -\hat{A}(n)\sin\left(\hat{\theta}(n)\right) & 0 \\ -\sin\left(\hat{\theta}(n)\right) & -\hat{A}(n)\cos\left(\hat{\theta}(n)\right) & 0 \end{bmatrix}}}_{\underline{\underline{H}}_n} \begin{bmatrix} \tilde{A}(n) \\ \tilde{\theta}(n) \\ \tilde{f}(n) \end{bmatrix} + \underline{w}_n$$

¡¡ In fact, we only need linearity to relate system error with state error !!!

$$\underline{\hat{A}}_0 = \underline{0} \qquad \underline{\underline{F}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2\pi T \\ 0 & 0 & 1 \end{bmatrix} \qquad \underline{\underline{H}}_n = \begin{bmatrix} \cos\left(\hat{\theta}(n)\right) & -\hat{A}(n)\sin\left(\hat{\theta}(n)\right) & 0 \\ -\sin\left(\hat{\theta}(n)\right) & -\hat{A}(n)\cos\left(\hat{\theta}(n)\right) & 0 \end{bmatrix}$$

$$\underline{\underline{\Sigma}}_0 = \begin{pmatrix} 10^2 & 0 & 0 \\ 0 & 90 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\underline{\underline{V}} = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 0.03 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}$$

$$E_n = \underline{\underline{H}}_n^H \underline{\underline{\Sigma}}_n \underline{\underline{H}}_n + \sigma^2 \underline{\underline{I}}$$

$$\underline{\underline{K}}_n = \underline{\underline{\Sigma}}_n \underline{\underline{H}}_n^H \underline{\underline{E}}_n^{-1}$$

$$\underline{\varepsilon}_n = \underline{z}_n - \underline{\hat{z}}_n = \underline{z}_n - \underline{\hat{A}}_n \begin{pmatrix} \cos\left(\hat{\theta}(n)\right) \\ -\sin\left(\hat{\theta}(n)\right) \end{pmatrix}$$

$$\underline{\hat{A}}_{n+1} = \underline{\hat{A}}_n + \underline{\underline{K}}_n \underline{\varepsilon}_n$$

Kalman filter for PLL

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{F}}\,\underline{\underline{\Sigma}}_n \underline{\underline{F}}^H - \underline{\underline{K}}_n \underline{\underline{E}}_n \underline{\underline{K}}_n^H + \underline{\underline{V}}$$

$$n \Rightarrow n+1$$

Square-Root Filter

Sometimes, very small miss-adjustment may promote that the covariance equation becomes eventually non- positive definite. In such a case, the algorithm locally diverges showing like a “periodic” re-setting to initial conditions just after achieving convergence.

This problem is easy to solve for the scalar measurement case, i.e. adaptive beamforming, thanks to the following expression of the covariance in terms of its square-root matrix.

$$\underline{\underline{\Sigma}}_{n+1} = \underline{\underline{\Sigma}}_n - \frac{\underline{\underline{\Sigma}}_n \underline{X}_n \underline{X}_n^H \underline{\underline{\Sigma}}_n}{\xi_{\min} + \underline{X}_n^H \underline{\underline{\Sigma}}_n \underline{X}_n} \quad \text{with the square-root of the covarianc}$$

$$\underline{\underline{\Sigma}}_n = \underline{\underline{S}}_n \underline{\underline{S}}_n^H \quad \text{we have} \quad \underline{\underline{S}}_{n+1} = \underline{\underline{S}}_n \left(\underline{\underline{I}} - \frac{\underline{\underline{S}}_n^H \underline{X}_n \underline{X}_n^H \underline{\underline{S}}_n}{\xi_{\min} + \underline{X}_n^H \underline{\underline{S}}_n \underline{\underline{S}}_n^H \underline{X}_n} \right) \underline{\underline{S}}_n^H$$

$$\left(\underline{\underline{I}} - \alpha \underline{d}_n \underline{d}_n^H\right) = \left(\underline{\underline{I}} - \beta \underline{d}_n \underline{d}_n^H\right)\left(\underline{\underline{I}} - \beta \underline{d}_n \underline{d}_n^H\right)$$

$$with \quad \underline{d}_n = \underline{\underline{S}}_n^H \underline{X}_n \qquad \alpha = \frac{1}{\xi_{\min} + \underline{d}_n^H \underline{d}_n}$$

The adequate value of β is:

$$\beta = 1 - \sqrt{1-\alpha^2} = \frac{1}{\left|\underline{d}_n\right|^2}\left(1 - \sqrt{\frac{1}{1+\left(\frac{\left|\underline{d}_n\right|^2}{\xi_{\min}}\right)}}\right)$$

With this the SRK algorithm is……..

Adaptive beamforming SRK

$$\boxed{\begin{array}{c} \underline{\hat{A}}_0 = \underline{0} \quad \underline{\underline{\Sigma}}_0 = 10^6 \underline{\underline{I}} \\[2ex] \underline{\underline{V}} = v_0 \underline{\underline{I}} = 10^{-3} \underline{\underline{I}} \end{array}}$$

$$E_n = \underline{d}_n^H \underline{d}_n + \xi_{\min}$$

$$\underline{K}_n = \underline{\underline{S}}_n \underline{d}_n / E_n$$

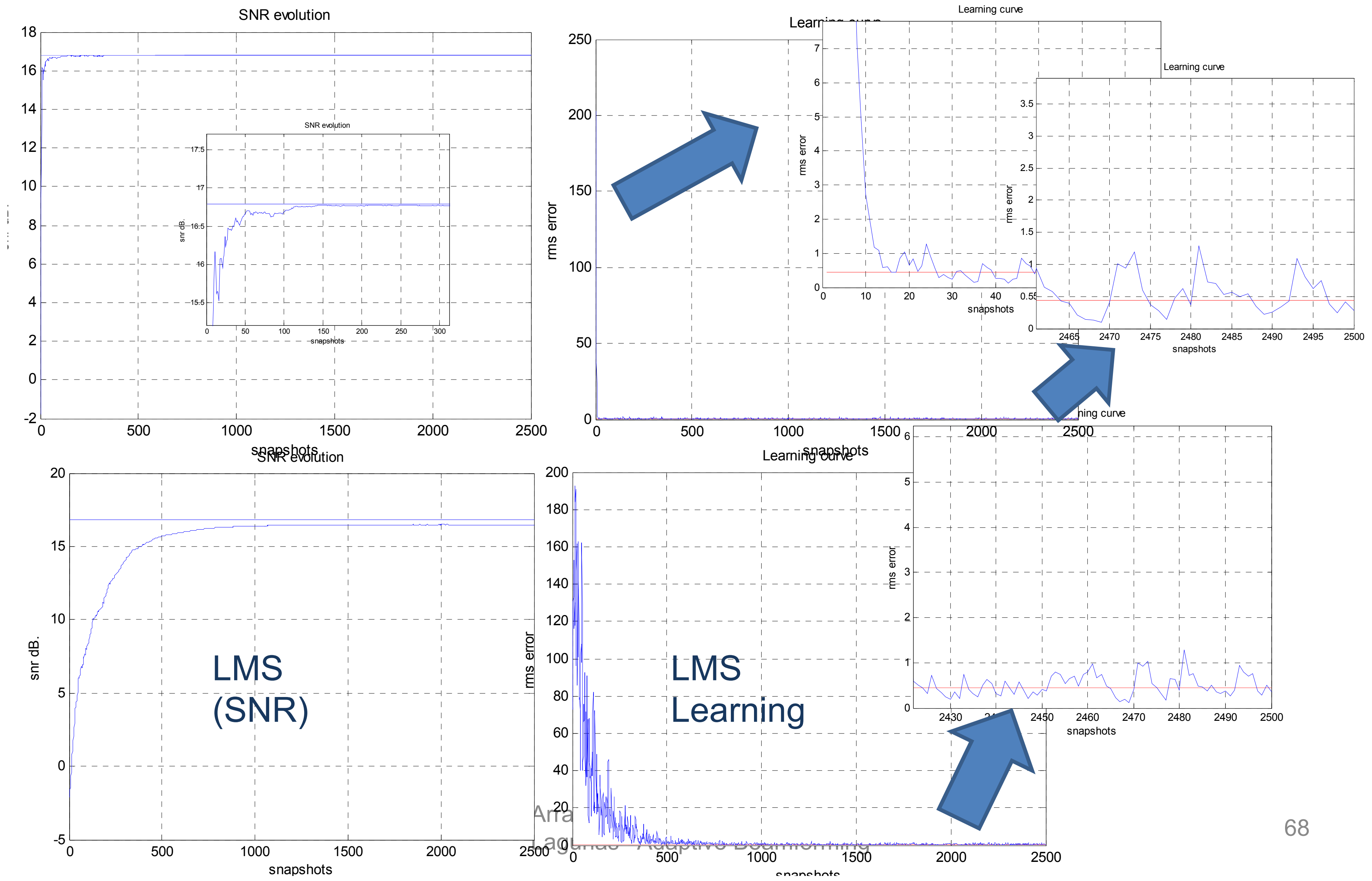
$$\varepsilon(n) = d(n)^* - \underline{X}_n^H \underline{\hat{A}}_n$$

$$\underline{\hat{A}}_{n+1} = \underline{\hat{A}}_n + \underline{K}_n \varepsilon(n)$$

$$\underline{\underline{S}}_{n+1} = \underline{\underline{S}}_n \left(\underline{\underline{I}} - \beta \underline{d}_n \underline{d}_n^H \right) + \underline{\underline{V}}^{1/2}$$

$$n \Rightarrow n+1$$

SRK Performance



LMS
(SNR)

LMS
Learning

The Recursive Least Squares (RLS)

The RLS algorithm is, basically, a different view of the Kalman filter.

Basically, the idea is that any adaptive beamformer should adapt its design to the changes in the data covariance and P-vector respectively. If both terms are updated in a recursive manner as:

$$\underline{\underline{R}}_{n+1} = \beta.\underline{\underline{R}}_n + (1-\beta).\underline{X}_n.\underline{X}_n^H$$

$$\underline{P}_{n+1} = \beta.\underline{P}_n + (1-\beta).\underline{X}_n.d^*(n)$$

The optimum beamformer will be:

$$\underline{A}_{n+1} = \underline{\underline{R}}_{n+1}^{-1}\underline{P}_{n+1}$$

RLS obeys to this principle but using the lemma of the inverse in the update of the data covariance.

$$\underline{\underline{A}} = \underline{\underline{B}} + \underline{\underline{C}}\,\underline{\underline{D}}\,\underline{\underline{C}}^H \Longrightarrow \underline{\underline{A}}^{-1} = \underline{\underline{B}}^{-1} - \underline{\underline{B}}^{-1}\underline{\underline{C}}\left[\underline{\underline{D}} + \underline{\underline{C}}^H\underline{\underline{B}}^{-1}\underline{\underline{C}}\right]^{-1}\underline{\underline{C}}^H\underline{\underline{B}}$$

Using this lemma with the inverse of the updated covariance…..

$$\underline{\underline{R}}_{n+1}^{-1} = \frac{1}{\beta}\underline{\underline{R}}_{n}^{-1} - \frac{1}{\beta}\underline{\underline{R}}_{n}^{-1}\underline{X}_n\left[\underline{\underline{I}} + \underline{X}_n^H\underline{\underline{R}}_n^{-1}\underline{X}^n\frac{1-\beta}{\beta}\right]^{-1}\frac{1-\beta}{\beta}\underline{X}_n^H\underline{\underline{R}}_n^{-1}$$

In addition……
$$\underline{A}_{n+1} = \underline{\underline{R}}_{n+1}^{-1}\left(\beta\underline{P}_n + (1-\beta)\underline{X}_n d^*(n)\right)$$

The two equations above can be re-formulated in a learning rule similar to the used in Kalman as

$$\underline{A}_{n+1} = \underline{A}_n + \underline{K}_n\varepsilon^*(n)$$

Being……

$$\alpha = \frac{1-\beta}{\beta}$$

$$\phi = \alpha.\left(\underline{X}_n^H.\underline{\underline{R}}_n^{-1}.\underline{X}_n\right)$$

$$\underline{K}_n = \left(\frac{\alpha}{1+\phi}\right)\underline{\underline{R}}_n^{-1}\underline{X}_n \Longrightarrow \underline{K}_n = \frac{(1-\beta).\underline{\underline{R}}_n^{-1}.\underline{X}_n}{\beta + (1-\beta).\left(\underline{X}_n^H.\underline{\underline{R}}_n^{-1}.\underline{X}_n\right)}$$

RLS Algorithm

Data $\underline{a}_n$,R_n^{-1},d(n) y $\underline{X}_n$

1.- Compute the beamformer output $y(n) = \underline{a}_n^H \underline{X}_n$

2.- Compute error with reference

$$\varepsilon(n) = d(n) - y(n)$$

3.- Compute

$$\phi = \alpha\left(\underline{X}_n^H \underline{\underline{R}}_n^{-1} \underline{X}_n\right) \quad being \quad \alpha = \frac{1-\beta}{\beta}$$

4.- Gain Matrix

$$\underline{K}_n = \frac{\alpha \underline{\underline{R}}_n^{-1} \underline{X}_n}{1+\phi}$$

5.- Update weights

$$\underline{A}_{n+1} = \underline{A}_n + \underline{K}_n \varepsilon^*(n)$$

6.- Update the inverse matrix

$$\underline{\underline{R}}_{n+1}^H = \frac{1}{\beta} \underline{\underline{R}}_n^{-1} - \underline{K}_n \underline{K}_n^H \frac{1+\phi}{1-\beta}$$

$$n \Rightarrow n+1$$

The RLS is equivalent to minimise a running average of the MSE error with the same constant used for updates of covariance a P-vector

$$MSE(n) = \beta MSE(n-1) + (1-\beta)\left|\varepsilon(n)\right|^2$$

Or, in a non-recursive manner……….

$$MSE(n) = (1-\beta).\sum_{m=-\infty}^{n} \beta^{n-m}.\left|\varepsilon(m)\right|^2 = (1-\beta).\sum_{m=-\infty}^{n} \beta^{n-m}.\left|d(m) - \underline{a}_{n+1}^H \underline{X}_m\right|^2$$

FromThales Alenia Space (Italy)