Blind Equalization

Sato Algorithm

The Sato algorithm is a special case of the Bussgang algorithm and pioneered the field of blind equalization in *M*-ary PAM systems. The Sato algorithm consists of minimizing a nonconvex cost function

$$J(n) = E[(\hat{x}(n) - y(n))^2]$$

where $\hat{x}(n)$ is an estimate of the data sequence $x(n)$ and $y(n)$ is the output of the equalizer. The estimate is obtained by using the zero-memory nonlinearity

$$\begin{aligned}\hat{x}(n) &= \gamma(y(n)) \\ &= \gamma\,\mathrm{sgn}[y(n)]\end{aligned}$$

where the gain of the equalizer is given by

$$\gamma = \frac{E[x^2(n)]}{E[|x(n)|]}.$$

We note the nonlinearity is the same in the decision-directed algorithm for binary PAM except for the equalizer gain.

The BGR theorem for convergence holds for the Sato algorithm even though the nonlinear function Ψ is not differentiable. According to BGR, global convergence for the Sato algorithm can be achieved provided the transmitted data sequence has a sub-Gaussian pdf and the equalizer is doubly infinite. For finite length equalizers, we have the following results for the Sato algorithm.

1) The cost function, exhibits local minima for discrete QAM input signals.
2) Algorithm may converge to local minima for both discrete and sub-Gaussian inputs.

Constant Modulus Algorithm (Godard)

Godard was the first to propose a family of constant modulus blind equalization algorithms for use in two-dimensional communications systems (*M*-ary phase shift keying like QAM, QPSK, etc...). Specifically, the Godard algorithm minimizes a nonconvex cost function of the form

$$J(n) = E\left[\left(|y(n)|^p - P_p\right)^2\right]$$

where p is a positive integer, and R_p is a positive real constant defined by

$$R_p = \frac{E[|x(n)|^{2p}]}{E[|x(n)|^p]}.$$

The Godard algorithm is designed to penalize deviations of the blind equalizer output, $y(n)$ from a constant modulus. The constant R_p is chosen in such a way that the gradient of the cost function $J(n)$ is zero when perfect equalization [i.e., $\hat{x}(n) = x(n)$] is attained.

The equalizer is updated according to the stochastic gradient algorithm

$$e(n) = y(n)|y(n)|^{p-2}\left(P_p - |y(n)|^p\right)$$
$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \mu \mathbf{u}(n)\varepsilon^*(n)$$

Case 1: p = 1
The cost function for this case is

$$J(n) = E[(|y(n)| - P_1)^2]$$

where

$$R_P = \frac{E[|\xi(n)|^2]}{E[|\xi(n)|]}$$

Case 2: p = 2
The cost function for this case is

$$J(n) = E\left[\left(|y(n)|^2 - P_2\right)^2\right]$$

where

$$R_P = \frac{E[|\xi(n)|^4]}{E[|\xi(n)|^2]}$$

This case is referred to as the Constant Modulus Algorithm (CMA) and is considered to be the most successful among the Bussgang algorithms. Note the use of higher-order statistics (HOS), i.e. fourth order moment in the numerator of R_p.

Blind Source Separation

Blind Source Separation (BSS) is also known as Blind Signal Separation (BSS) and when applied to speech signals is known as Blind Speech Separation (BSS). In all three ways, it's BSS.

The purpose of BSS is to operate on a set of observations (measurements) $u_1(n)$, ..., $u_N(n)$ made at the output of a mixer so as to estimate a set of original source signals, $s_1(n)$, ..., $s_M(n)$ applied to the mixer input. The requirement is to unravel the mixing process and provide reliable estimates of the original source signals.

Figure: Unknown environment

i. The solution to the blind source separation problem depends on the following issues:

ii. Whether the mixer is linear or nonlinear

iii. Whether the mixer is fixed or time-varying

iv. Whether the mixing operation if nonconvolutive or convolutive

v. Whether the sensors are noiseless or noisy

vi. How many sources, *M* and the number of sensors *N*, are related.

The simplest scenario arises when the mixer is linear, fixed, and nonconvolutive, the sensors are all noiseless, and $N = M$. In this case we write

$$\mathbf{x}(n) = \mathbf{A}\mathbf{s}(n)$$

where $\mathbf{x}(n) = [x_1(n),\ x_2(n),\ \mathrm{K}\ ,\ x_M(n)]^T$, $\mathbf{s}(n) = [s_1(n),\ s_2(n),\ \mathrm{K}\ ,\ s_M(n)]^T$, and $\mathbf{A}$ is a square matrix of unknown scalars.

In this scenario each element of $\mathbf{x}(n)$ is a linear combination of source signals. Provided $\mathbf{A}$ is invertible it is possible to recover the original source signals to within a scale factor and index permutation. (By permutation we mean that the arrangement of the demixer outputs may be different than the arrangement of the original source inputs.)

The goal is therefore the adaptively construct a demixing filter $\mathbf{W}$ such that

$$\begin{aligned}\mathbf{y}(n) &= \mathbf{W}\mathbf{u}(n)\\ &= \mathbf{W}\mathbf{A}\mathbf{s}(n)\\ &= \mathbf{P}\Lambda\mathbf{s}(n)\end{aligned}$$

where Λ is a diagonal matrix and $\mathbf{P}$ is a permutation matrix.

Example: Suppose $\mathbf{W}$ is constructed so that $\mathbf{W}\mathbf{A} = \mathbf{P}\Lambda$ where

$$\mathbf{P} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ \Lambda = \begin{bmatrix}\alpha & 0\\ 0 & \beta\end{bmatrix}.$$

Then $\mathbf{x}(n)$ projected onto $\mathbf{W}$ applied to would yield

$$\begin{aligned}\mathbf{y}(n) &= \begin{bmatrix}y_1(n)\\ y_2(n)\end{bmatrix}\\ &= \mathbf{W}\begin{bmatrix}x_1(n)\\ x_2(n)\end{bmatrix}\\ &= \mathbf{W}\mathbf{A}\begin{bmatrix}s_1(n)\\ s_2(n)\end{bmatrix}\\ &= \mathbf{P}\Lambda\begin{bmatrix}s_1(n)\\ s_2(n)\end{bmatrix}\\ &= \begin{bmatrix}\alpha s_1(n)\\ \beta_2(n)\end{bmatrix}\end{aligned}$$

and our outputs would be scaled versions of the original sources in the original order. On the other hand suppose

$$\mathbf{P} = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix},\ \Lambda = \begin{bmatrix}\alpha & 0\\ 0 & \beta\end{bmatrix}$$

then

$$\mathbf{y}(n) = \begin{bmatrix} \beta s_2(n) \\ \alpha s_1(n) \end{bmatrix}$$

and our outputs would be scaled versions of the original sources in the reverse order. With either permutation, we get our original sources which is the goal

Figure: Blind Source Separator

Blind Speech Separation (De Leon and LeBlanc)

As an application of BSS we consider the separation of linear, fixed, nonconvolutive mixtures of speech signals. This problem is difficult due to the nonstationarity of speech.

Fundamental Property

We have observed that the process of mixing speech signals together decreases the kurtosis as compared to the kurtosis of the individual speech signals. Here kurtosis is defined as

$$\kappa = \frac{E[x^4]}{\{E[x^2]\}^2}.$$

Property Restoration Approach

Therefore, we adjust **W** so as to maximize kurtosis of the output signals and therefore restore the original HOS property . As it turns out, maximizing kurtosis often leads to a separation of speech signals from mixtures.

Algorithm

In order to maximize kurtosis, we use SD but with a positive corrective step (we normally use a negative step to as to minimize as cost function)

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \nabla \kappa_{\mathbf{y}}$$

which leads to

$$\begin{aligned}
\mathbf{y}(n) &= \mathbf{W}(n)\mathbf{x}(n); \text{ Separate} \\
\hat{\sigma}_i^2(n) &= \lambda_2 \hat{\sigma}_i^2(n-1) + (1-\lambda_2)x_i^2(n); \text{ AR estimate for input variance} \\
\hat{r}_{12}(n) &= \lambda_2 \hat{r}_{12}(n-1) + (1-\lambda_2)x_1(n)x_2(n); \text{ AR estimate for input corr} \\
\alpha_i &= 4y_i^3(n) \\
\beta_i &= -W_{i1}(n)\hat{r}_{12}(n)x_1(n) - W_{i2}(n)\hat{\sigma}_2^2(n)x_1(n) + \\
&\quad W_{i1}(n)\hat{\sigma}_1^2(n)x_2(n) + W_{i2}(n)\hat{r}_{12}(n)x_2(n) \\
\gamma_i &= [W_{i1}^2(n)\hat{\sigma}_1^2(n) + 2W_{i1}(n)W_{i2}(n)\hat{r}_{12}(n) + W_{i2}^2(n)\hat{\sigma}_2^2(n)]^{-3} \\
\mathbf{C}(n) &= \begin{bmatrix} -\alpha_1\beta_1\gamma_1 W_{12}(n) & \alpha_1\beta_1\gamma_1 W_{11}(n) \\ -\alpha_2\beta_2\gamma_2 W_{22}(n) & \alpha_2\beta_2\gamma_2 W_{21}(n) \end{bmatrix}; \text{ Build correction matrix} \\
\mathbf{W}(n+1) &= \mathbf{W}(n) + \frac{\tilde{\mu}}{\|\mathbf{C}(n)\|_F^2}\mathbf{C}(n); \text{ Update separation matrix}
\end{aligned}$$

Examples.