Novel Adaptive filtering Algorithms for Bilinear Sparse systems

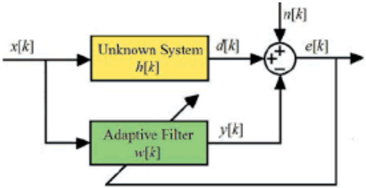
Battu Sri Charan
Supervisor: Prof. Mrityunjoy Chakraborty

May 3, 2021

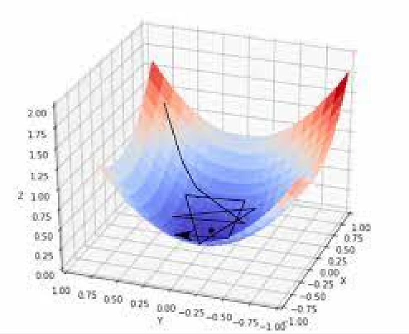
IIT Kharagpur
sricharanbattu@gmail.com

- Filter coefficients are unknown
- Filter coefficients need to be estimated
- Input data and the output is available
- Improvises the estimates from the error of the previous estimate

Fig. 1. ADAPTIVE FILTER STRUCTURE FOR SYSTEM IDENTIFICATION



Fuente: Autores



- Follows Steep Gradient descent for estimation
- Minimizes a loss function $\mathcal{L}$(usually expectation of squared error + some additional term based on more information about the system)

▷ $\underline{w}(i+1) = \underline{w}(i) - \mu.\frac{\partial \mathcal{L}}{\partial \underline{w}}$

Some rules about the Performance of Algorithm

- More the noise, more is the misalignment
- Higher the value of μ, more is misalignment, but faster is the convergence.
- Higher the number of coefficients, more time does it take to converge and more is the misalignment for same μ
- More information about the system implies less achievable misalignment
- We have two main types of algorithms : LMS and NLMS

LMS and NLMS

LMS

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i) + \mu.\underline{\mathrm{x}}(i).e(i)$$

NLMS

$$\underline{\mathrm{w}}(i+1) = \underline{\mathrm{w}}(i+1) + \mu\frac{\underline{\mathrm{X}}(i).e(i)}{\underline{\mathrm{X}}^T(i).\underline{\mathrm{X}}(i)}$$

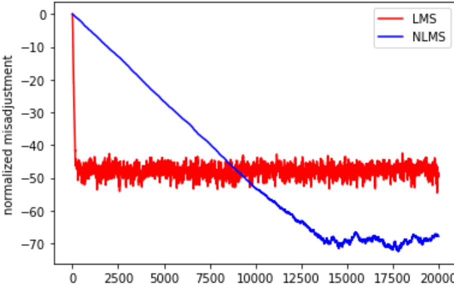
LMS and NLMS

LMS

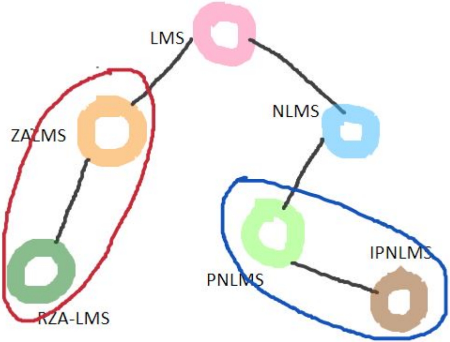
$$\underline{w}(i+1) = \underline{w}(i) + \mu.\underline{x}(i).e(i)$$

NLMS

$$\underline{w}(i+1) = \underline{w}(i+1) + \mu\frac{\underline{x}(i).e(i)}{\underline{x}^T(i).\underline{x}(i)}$$



Sparse Algorithms



- **LMS**
  - ZA-LMS
  - RZA-LMS
- **NLMS**
  - PNLMS
  - IPNLMS

Zero Attractor Class

Zero Attractor Class

▷ ZA-LMS

- Minimizes Mean Squared Error + L_1 norm of the filter
- $\mathcal{L} = \frac{1}{2}e^2 + \gamma||\underline{w}||_1$
- $\underline{w}(i+1) = \underline{w}(i) + \mu\underline{x}(i)e(i) - \rho.sgn(\underline{w}(i))$ where $\rho = \gamma.\mu$
- It tend to shrink the coefficients, but the active taps are also effected

Zero Attractor Class

- **ZA-LMS**
  - Minimizes Mean Squared Error + L_1 norm of the filter
  - $\mathcal{L} = \frac{1}{2}e^2 + \gamma||\underline{w}||_1$
  - $\underline{w}(i+1) = \underline{w}(i) + \mu\underline{x}(i)e(i) - \rho.sgn(\underline{w}(i))$ where $\rho = \gamma.\mu$
  - It tend to shrink the coefficients, but the active taps are also effected
- **RZA-LMS**
  - To shield the active taps from the influence of zero attraction, Re-weighted ZA-LMS is proposed.
  - $\mathcal{L} = \frac{1}{2}e^2 + \gamma' \sum_{i=1}^{n} \ln(1 + \frac{|w_i|}{\epsilon'})$
  - $\underline{w}(i+1) = \underline{w}(i) + \mu.\underline{x}(i)e(i) - \rho.\frac{sgn(\underline{W}(i))}{1+\epsilon.|\underline{W}(i)|}$, where $\rho = \mu.\gamma'/\epsilon'$,$\epsilon = 1/\epsilon'$

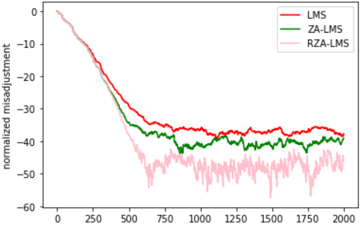
Proportionate Class

- Step size for each coefficient varies based on its magnitude. coefficients with large magnitudes get larger step sizes.
- $\underline{w}(i+1) = \underline{w}(i) + \frac{\mu G(i)\underline{x}(i)e(i)}{\underline{x}^T(i)G(i)\underline{x}(i)+\delta_{PNLMS}}$ where
  - $G(i) = diag\{g_0(i), g_1(i), ..., g_n(i)\}$
  - $g_l(i) = \frac{\gamma_l(i)}{\sum_{j=1}^{n} \gamma_j(i)} \quad 0 \leq l \leq n$
  - $\gamma_l(i) = max\{\rho.max[\delta_p, |w_0(i)|, |w_1(i)|, ..., |w_n(i)|], |w_l(i)|\}$
    $0 \leq l \leq n$
- The above is PNLMS algorithm. An improved version on IPNLMS performs even better than PNLMS.

Performance of Sparse Algorithms

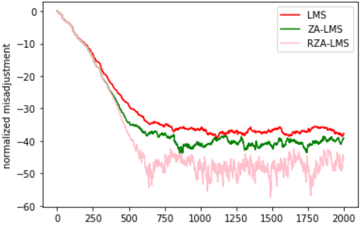
Performance of Sparse Algorithms

- Sparse : 5 of 64

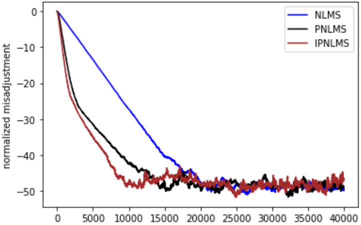


Performance of Sparse Algorithms

- Sparse : 5 of 64

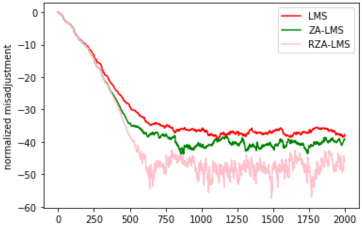


- Sparse : 5 of 64

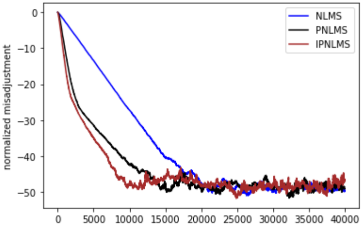


Performance of Sparse Algorithms

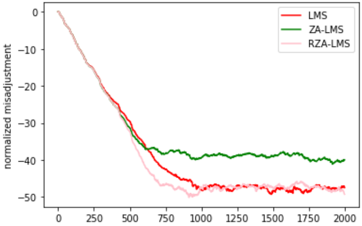
- Sparse : 5 of 64



- Sparse : 5 of 64

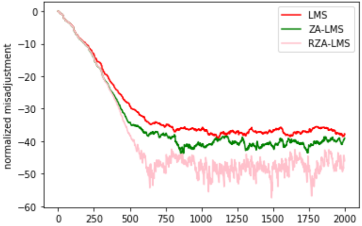


- Semisparse : 32 of 64

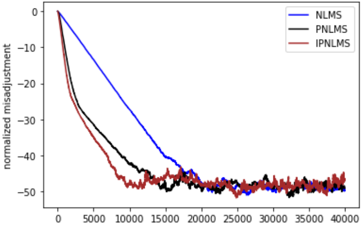


Performance of Sparse Algorithms

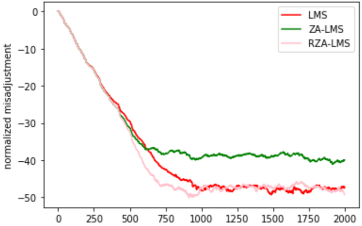
- Sparse : 5 of 64



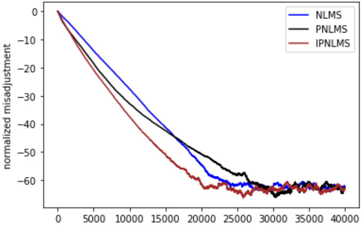
- Sparse : 5 of 64



- Semisparse : 32 of 64



- Semi sparse : 32 of 64



Bilinear Systems

- Bilinear systems are modelled by this equation.

$$d(i) = \underline{h}^T X(i) \underline{g} + s(i) = y(i) + s(i)$$

where

- $[X(i)]_{LxM} = [\underline{x}_1(i), \underline{x}_2(i), ..., \underline{x}_M(i)]$
- $\underline{x}_m(i) = [x_m(i), x_m(i-1), ..., x_m(i-L+1)] \;\; \forall 1 \leq m \leq M$

- $vec[X(i)] = [\ \underline{x}_1^T(i), \underline{x}_2^T(i), ..., \underline{x}_m^T(i)\ ]^T = \widetilde{\underline{x}}(i)$
  - It can be derived that $y(i) = \underline{f}^T \widetilde{\underline{x}}(i)$,where $\underline{f} = \underline{g} \otimes \underline{h}$

- There could be a scaling ambiguity in identifying $\underline{h}$ and $\underline{g}$ as $\frac{1}{\eta}\underline{f}$ and $\eta\underline{g}$ also gives same output

Bilinear Systems and Notations for Adaptive algorithms

- We typically have two adaptive equations, one for $\underline{\mathrm{h}}$ and other for $\underline{\mathrm{g}}$.
- While updating the equation for one filter, Mathematical results are interpreted as if the other filter is constant
- We define the following terms for repeated usage in Adaptive algorithms for bilinear systems.

$$e_{\underline{\hat{\mathrm{g}}}}(i+1) = d(i+1) - \underline{\hat{\mathrm{h}}}^T(i)\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{g}}}}(i+1)$$
$$e_{\underline{\hat{\mathrm{h}}}}(i+1) = d(i+1) - \underline{\hat{\mathrm{g}}}^T(i)\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1)$$

where

$$\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{g}}}}(i+1) = [\underline{\hat{\mathrm{g}}}(i) \otimes I_L]^T\underline{\widetilde{\mathrm{x}}}(i+1)$$
$$\underline{\widetilde{\mathrm{x}}}_{\underline{\hat{\mathrm{h}}}}(i+1) = [I_M \otimes \underline{\hat{\mathrm{h}}}(i)]^T\underline{\widetilde{\mathrm{x}}}(i+1)$$

Some Assumptions and Rules

Some Assumptions and Rules

▷ **Assumptions :**

- The temporal part $\underline{h}$ of filter is longer than the spatial part $\underline{g}$
- The temporal part is usually sparse and the spatial part is dense

Some Assumptions and Rules

▷ **Assumptions :**

- The temporal part $\underline{h}$ of filter is longer than the spatial part $\underline{g}$
- The temporal part is usually sparse and the spatial part is dense

▷ **Rules :**

- The performance rules specified for linear systems apply for bilinear systems also
- Algorithms for both filters either converge together or diverge together
- If an algorithm performs well for one filter, it also performs well for the other filter
- Bilinear systems can also be modelled as linear systems with $\underline{f}$ as filter but it takes too long to converge. Hence, bilinear algorithms are favoured.

Bilinear Algorithms

LMS-BF

$$\underline{\hat{h}}(i+1) = \underline{\hat{h}}(i) + \mu_{\underline{\hat{h}}}.\underline{\widetilde{x}}_{\underline{\hat{g}}}(i+1).e_{\underline{\hat{g}}}(i+1)$$
$$\underline{\hat{g}}(i+1) = \underline{\hat{g}}(i) + \mu_{\underline{\hat{g}}}.\underline{\widetilde{x}}_{\underline{\hat{h}}}(i+1).e_{\underline{\hat{h}}}(i+1)$$

- The other Algorithms also follow similarly.
- Sparsity is usually present in $\underline{h}$. Hence, the sparse versions of algorithms are applied to first updation step of $\underline{h}$
- The Algorithms studied for Bilinear filters are : LMS, NLMS, PNLMS and IPNLMS.

Hard Thresholding For Linear Filters

- One of the Novel methods used is Hard Thresholding
- After the LMS step in each iteration, Take only those values whose magnitudes exceed a certain specified threshold and make others 0.

Hard Thresholding For Linear Filters

- One of the Novel methods used is Hard Thresholding
- After the LMS step in each iteration, Take only those values whose magnitudes exceed a certain specified threshold and make others 0.

▷ **Issues of the Algorithm :**

- If the threshold is too large, the Algorithm may not even begin
- If the threshold is too low, No difference between LMS and thresholding
- The threshold is dfficult to specify as we don't know the nature of the coefficients.

Adaptive Hard Thesholding : Mean

- Make Threshold Adaptive
- A natural adaptive threshold : Mean of the magnitudes of current coefficients
- Works better than Hard Thresholding

Adaptive Hard Thesholding : Mean

- Make Threshold Adaptive
- A natural adaptive threshold : Mean of the magnitudes of current coefficients
- Works better than Hard Thresholding

▷ **Issues of Mean Threshold :**

- A large variation in magnitudes of the active coefficients suppress some crucial coefficients
- Only works for very sparse filters even if the variation is less

Adaptive Hard Thresholding : Weighted Mean

- Instead of Mean, take Weighted Means.
- A natural weighted mean is exponential weighted mean
- Give higher weights for lower coefficients
- Performs much better than Mean threshold

Adaptive Hard Thresholding : Weighted Mean

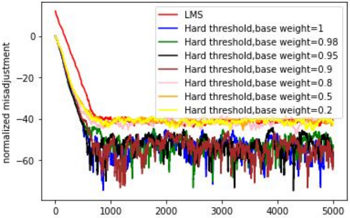
- Instead of Mean, take Weighted Means.
- A natural weighted mean is exponential weighted mean
- Give higher weights for lower coefficients
- Performs much better than Mean threshold

▷ **Issues with weighted mean:**
  - Higher weights perform similar to Mean threshold
  - Lower weights perform similar to LMS
  - More number of active coefficients need lower weights
  - Less number of active coefficients perform better with higher weights.
  - We need more information on degree of sparsity for choosing weights

Performance of Adaptive Hard Thresholding

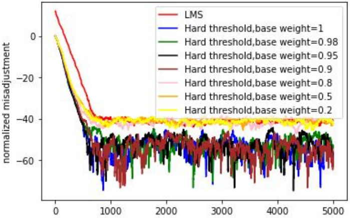
Performance of Adaptive Hard Thresholding

3/32 active taps



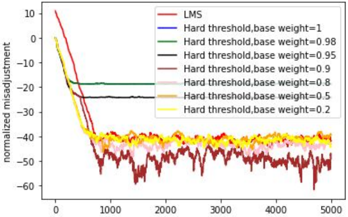
Performance of Adaptive Hard Thresholding

3/32 active taps



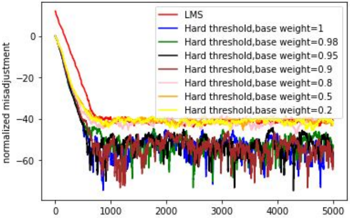
6/32 active taps

0.09545642007692401



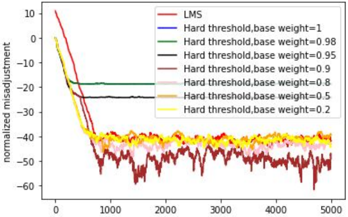
Performance of Adaptive Hard Thresholding

3/32 active taps

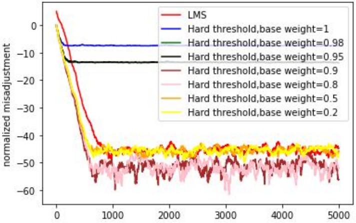


6/32 active taps

0.09545642007692401

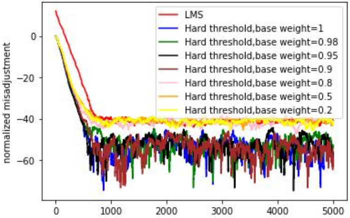


12/32 active taps



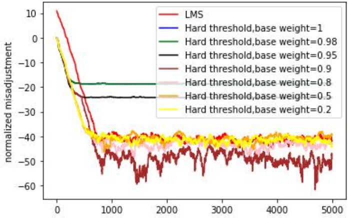
Performance of Adaptive Hard Thresholding

3/32 active taps

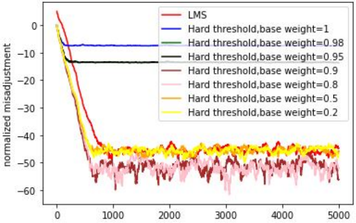


6/32 active taps

0.0954564207692401

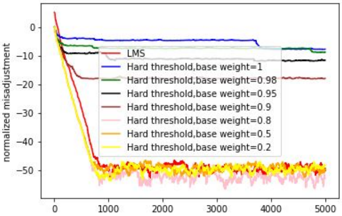


12/32 active taps



20/32 active taps

0.11242837176692722



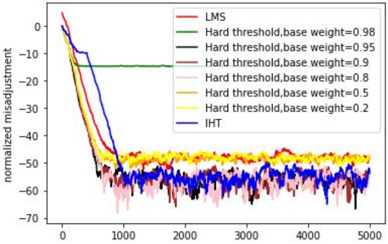
Iterative Hard thresholding(IHT)

- Inspired from IHT in compressed sensing
- When information on degree of sparsity is available, IHT is better than weighted mean threshold
- Only a specified number(say k) coefficients are retained(those with k largest magnitudes) and others are set to 0
- k should be at least the number of active taps
- Performance much better than LMS always when k is at least the number of active taps and performs worse than LMS in other cases
- It is highly reliable, given the extra information is available

Performance of IHT

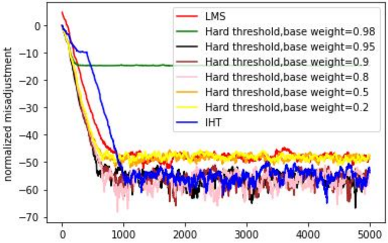
Performance of IHT

8/32 active

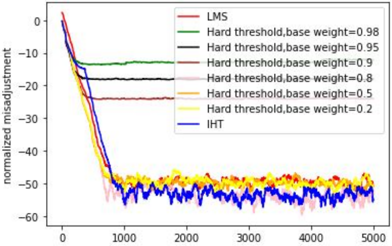


Performance of IHT

8/32 active

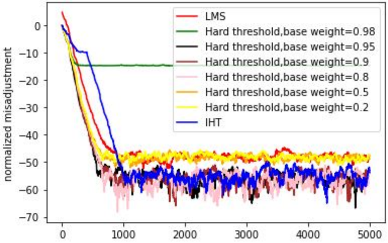


16/32 active

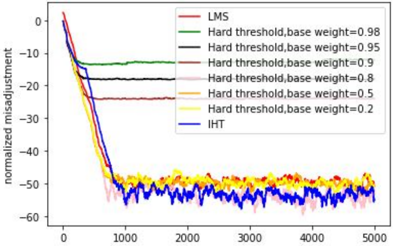


Performance of IHT

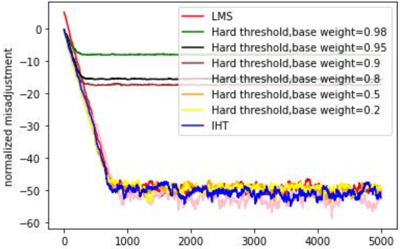
8/32 active



16/32 active

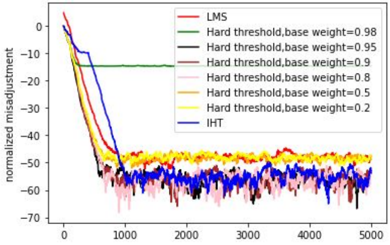


24/32 active

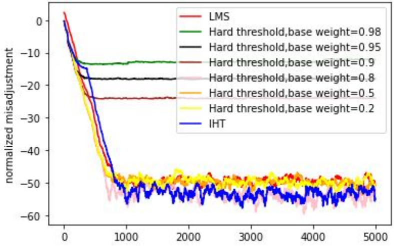


Performance of IHT

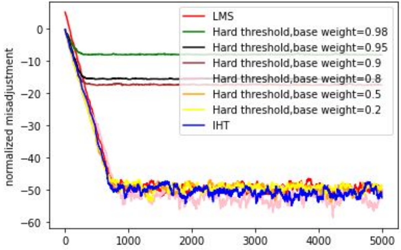
8/32 active



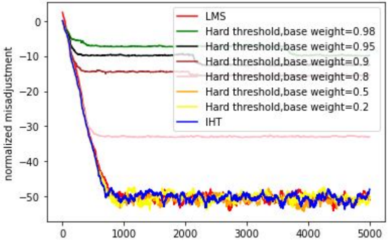
16/32 active



24/32 active



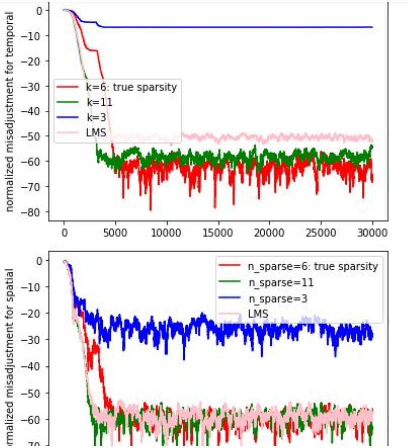
32/32 active



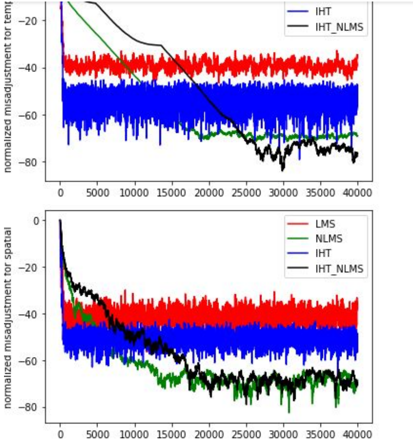
Thresholding for Bilinear Systems

- Thresholding Algorithms are applied for the temporal filter in Bilinear case
- The Performance behaviour and the issues observed are similar to those in linear filters
- Convergence Analysis for Bilinear systems is a little more complex than for linear filters.

Variation of misalignment with chosen k



IHT-BF with LMS and NLMS



Issues with Zero Attractor for Bilinear Systems

- The Zero attractor algorithm is very much sensitive the zero attractor parameter
- Sometimes, if the parameter is large, Divergence results followed by a periodic convergence. This is an interesting phenomena, which needs to be examined
- The misalignment decreases and then increases and then decreases and so on
- If the value is taken low, the above problem can be eliminated
- But too low values again result in normal LMS algorithm
- Hence the parameter must be chosen carefully. ZA-LMS in itself is very dangerous for Bilinear filters

Proportionate + Zero Attractor?

- The problem with ZA-LMS is that only zero taps converge faster because of shrinkage
- The problem with PNLMS is that only active taps converge faster
- In a sense, they are complementary to each other

Proportionate + Zero Attractor?

- The problem with ZA-LMS is that only zero taps converge faster because of shrinkage
- The problem with PNLMS is that only active taps converge faster
- In a sense, they are complementary to each other

Why not combine them both to get a faster convergence?

Combination of PNLMS and ZA-LMS

The combined algorithm of PNLMS and ZA-LMS for Bilinear filters is as follows :

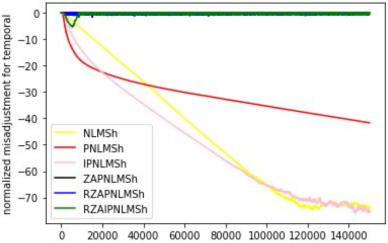
$$\underline{\hat{h}}(i+1) = \underline{\hat{h}}(i) + \frac{\mu_{\underline{\hat{h}}}.G(i).\underline{\widetilde{x}}_{\underline{\hat{g}}}(i+1).e_{\underline{\hat{g}}}(i+1)}{\underline{\widetilde{x}}^{T}_{\underline{\hat{g}}}(i+1).G(i).\underline{\widetilde{x}}_{\underline{\hat{g}}}(i+1)+\delta_{\underline{\hat{h}}_{PNLMS}}} - \beta.sgn\{\underline{\hat{h}}(i)\}$$

$$\underline{\hat{g}}(i+1) = \underline{\hat{g}}(i) + \frac{\mu_{\underline{\hat{g}}}.\underline{\widetilde{x}}_{\underline{\hat{h}}}(i+1).e_{\underline{\hat{h}}}(i+1)}{\underline{\widetilde{x}}^{T}_{\underline{\hat{h}}}(i+1)\underline{\widetilde{x}}_{\underline{\hat{h}}}(i+1)+\delta_{\underline{\hat{g}}_{PNLMS}}}$$

Performance of Combined algorithm

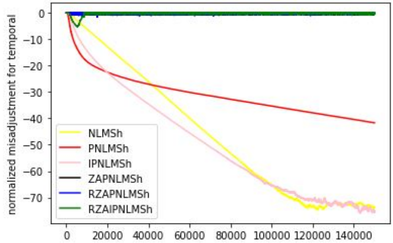
Performance of Combined algorithm

β=4e-5

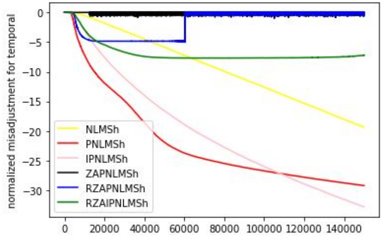


Performance of Combined algorithm

β=4e-5

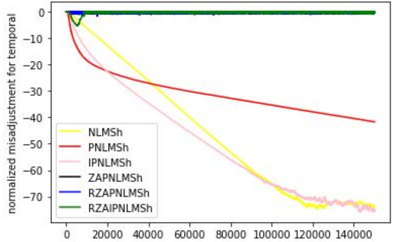


β=4e-6

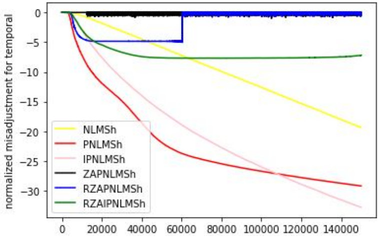


Performance of Combined algorithm

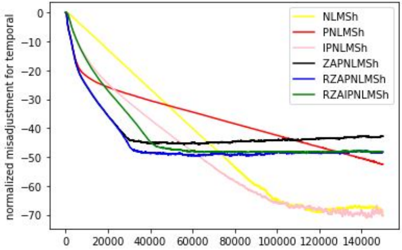
β=4e-5



β=4e-6

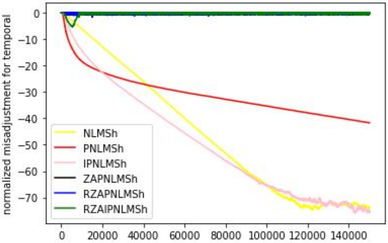


β=4e-7

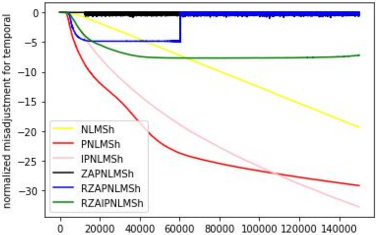


Performance of Combined algorithm

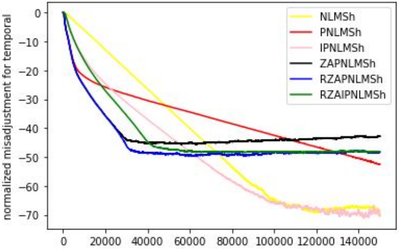
β=4e-5



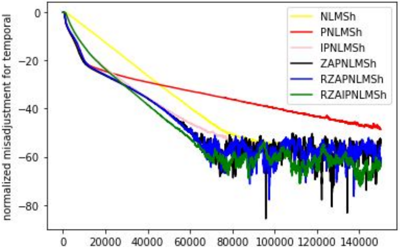
β=4e-6



β=4e-7



β=4e-8



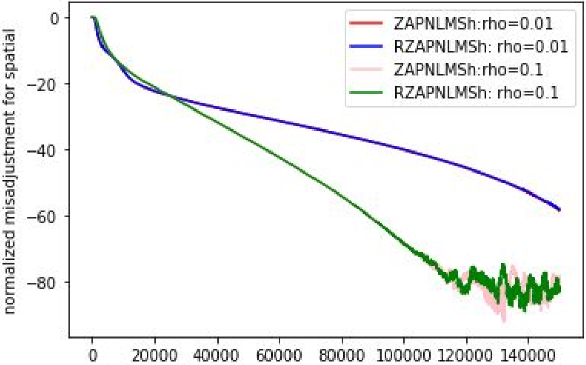
Improvement of Misalignment of Combined Algorithm

Improvement of Misalignment of Combined Algorithm

- The misalignment of the combined Algorithm can be improved by increasing the ρ parameter in PNLMS algorithm.
- When stable state is reached, the active taps need not have higher step sizes any longer. To restrict the step size at this stage, ρ is useful

Improvement of Misalignment of Combined Algorithm

- The misalignment of the combined Algorithm can be improved by increasing the ρ parameter in PNLMS algorithm.
- When stable state is reached, the active taps need not have higher step sizes any longer. To restrict the step size at this stage, ρ is useful



Thanks for your time