Does it exist a general form for adaptation/learning algorithms?

I.D. Landau
CNRS, GIPSA-LAB, Grenoble, France

In collaboration with:
T. Airimitoaie, B. Vau and G. Buche

International Carpathian Control Conference, 2022
Sinaia, May 30, 2022 (virtual)
Adaptation /Learning Algorithms = *The Fiddler’s Paradise!*
Booming of the number of papers on « adaptation/learning » algorithms in the last 12 years! (however some algorithms are very old: goes back to 1965 - 1975)
Booming of the number of papers on « adaptation/learning » algorithms in the last 12 years! (however some algorithms are very old: goes back to 1965 -1975)

With memory

- Gradient algorithm(s)
- Conjugate gradients
- Nesterov algorithm
- Back momentum
- Average gradients
- Integral + Proportional
- Integral +Proportional+ Derivative
- ...........
- ...........

Without memory

- Leakage algorithm
- ............
Booming of the number of papers on « adaptation/learning » algorithms in the last 12 years! (however some algorithms are very old: goes back to 1965 -1975)

**With memory**

- Gradient algorithm(s)
- Conjugate gradients
- Nesterov algorithm
- Back momentum
- Average gradients
- Integral + Proportional
- Integral +Proportional+ Derivative
- ............
- ............

**Without memory**

- Leakage algorithm
- ............

- **Number of algorithms have been re-discovered (often without reference to the original).**
- Many algorithms differs just by the way that the equations are written.
- Absence of comparisons.
- Atentive reading allows to reduce significantly the number of truly « original » algorithms.
- Absence of serious theoretical analysis (most of the cases).
- The stability issues seldom discussed (most of the cases)
What should you remember from this talk?

\[ \hat{\theta}(t+1) = \hat{\theta}(t) + \text{correcting term} \]

Vector of parameters

\[ \hat{\theta}(t+1) = \frac{1}{1 - q^{-1}} \text{correcting term} \]

Integrator

- New estimated parameter vector = the correcting term filtered by an « integrator »
- Gradient algorithm: the correcting term is the « gradient » of the criterion to be minimized with minus sign
What should you remember from this talk?

\[ \hat{\theta}(t + 1) = \hat{\theta}(t) + \text{correcting term} \]

- New estimated parameter vector = the correcting term filtered by an « integrator »
- Gradient algorithm: the correcting term is the « gradient » of the criterion to be minimized with minus sign
- All the other algorithms: just replace the « integral » filter by a filter with poles and zeros (one can generate an infinite number of adaptation/learning algorithms!)
What should you remember from this talk?

\[
\dot{\theta}(t + 1) = \dot{\theta}(t) + \text{correcting term}
\]

- New estimated parameter vector = the correcting term filtered by an « integrator »
- Gradient algorithm: the correcting term is the « gradient » of the criterion to be minimized with minus sign
- All the other algorithms: just replace the « integral » filter by a filter with poles and zeros (one can generate an infinite number of adaptation/learning algorithms!)
- Stability of adaptation/learning algorithms is a very important issue.
- **Stability** of the adaptation/learning alg. for a large range of adaptation gains/learning rates requires that the filter acting on the gradient be characterized by a **positive real transfer function** (phase lag \(\leq 90^\circ\) for all freq)
- Some of the algorithms can improve the performance of the gradient algorithm.
An example in Adaptive Feedorward Noise Attenuation

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
An example in Adaptive Feedforward Noise Attenuation

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
An example in Adaptive Feedforward Noise Attenuation

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
• Revisiting the gradient algorithm – Feedback interpretation and stability issues
• A general form for adaptation/learning algorithms
• Stability issues for high and low adaptation gains/learning rates (slow and fast adaptation/learning)
• The ARIMA 2 (I+P+DF) algorithm (new)
• A review of existing adaptation/learning algorithms
• Comparisons – Simulation results
• Comparisons – Real time results (adaptive active noise control)
• Concluding remarks
Discrete time plant model (unknown parameters))

\[ y(t + 1) = -a_1 y(t) + b_1 u(t) = \theta^T \phi(t) \]

\[ \theta^T = [a_1, b_1] \quad \text{Parameter vector} \quad ; \quad \phi(t)^T = [-y(t), u(t)] \quad \text{Vector of attributes} \]
Algorithms for parameter estimation

Discrete time plant model (unknown parameters))

\[ y(t + 1) = -a_1 y(t) + b_1 u(t) = \theta^T \phi(t) \]

\[ \theta^T = [a_1, b_1] \quad \text{Parameter vector} \quad \phi(t)^T = [-y(t), u(t)] \]

Adjustable prediction model (a priori)

\[ \hat{y}_o(t + 1) = \hat{y}(t + 1|\hat{\theta}(t)) = -\hat{a}_1(t) y(t) + \hat{b}_1(t) u(t) = \hat{\theta}(t)^T \phi(t) \]

\[ \theta(t)^T = [\hat{a}_1(t), \hat{b}_1(t)] \quad \text{Vector of adjustable parameters} \]

Prediction error (a priori) \[ \varepsilon^o(t + 1) = y(t + 1) - \hat{y}_o(t + 1) = \varepsilon^o(t + 1, \hat{\theta}(t)) \]

Adjustable prediction model (a posteriori)

\[ \hat{y}(t + 1) = \hat{y}(t + 1|\hat{\theta}(t + 1)) = -\hat{a}_1(t + 1) y(t) + \hat{b}_1(t + 1) u(t) = \hat{\theta}(t + 1)^T \phi(t) \]

Prediction error (a posteriori): \[ \varepsilon(t + 1) = y(t + 1) - \hat{y}(t + 1) \]

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Discrete time plant model (unknown parameters))

\[ y(t + 1) = -a_1 y(t) + b_1 u(t) = \theta^T \phi(t) \]

\[ \theta^T = [a_1, b_1] \quad \text{Parameter vector} \quad ; \quad \phi(t)^T = [-y(t), u(t)] \]

Adjustable prediction model (a priori)

\[ \hat{y}^o(t + 1) = \hat{y}(t + 1 \mid \hat{\theta}(t)) = -\hat{a}_1(t) y(t) + \hat{b}_1(t) u(t) = \hat{\theta}(t)^T \phi(t) \]

\[ \theta(t)^T = [\hat{a}_1(t), \hat{b}_1(t)] \quad \text{Vector of adjustable parameters} \]

Prediction error (a priori)

\[ \epsilon^o(t + 1) = y(t + 1) - \hat{y}^o(t + 1) = \epsilon^o(t + 1, \hat{\theta}(t)) \]

Adjustable prediction model (a posteriori)

\[ \hat{y}(t + 1) = \hat{y}(t + 1 \mid \hat{\theta}(t + 1)) = -\hat{a}_1(t + 1) y(t) + \hat{b}_1(t + 1) u(t) = \hat{\theta}(t + 1)^T \phi(t) \]

Prediction error (a posteriori):

\[ \epsilon(t + 1) = y(t + 1) - \hat{y}(t + 1) \]

Parameter adaptation algorithm (learning algorithm)

\[ \hat{\theta}(t + 1) = \hat{\theta}(t) + \Delta \hat{\theta}(t + 1) = \hat{\theta}(t) + f\left(\hat{\theta}(t), \phi(t), \epsilon^o(t + 1)\right) \]
**PAA – Gradient algorithm (steepest descent)**

Criterion to be minimized (objective):

\[
\min_{\hat{\theta}(t+1)} J(t+1) = \left[ \epsilon(t+1) \right]^2
\]

Gradient strategy: \((*)\) \(\hat{\theta}(t+1) = \hat{\theta}(t) - F \frac{\delta J(t+1)}{\delta \hat{\theta}(t+1)} \quad F = \alpha I \quad (\alpha > 0) \quad (I = \text{unit matrix})
\]

\((\alpha - \text{adaptation gain/learning rate})

Other criteria can be considered

---

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
PAA – Gradient algorithm (steepest descent)

Criterion to be minimized (objective):

\[
\min_{\hat{\theta}(t+1)} J(t+1) = [\varepsilon(t+1)]^2
\]

**Gradient strategy:**

\[
(*) \quad \hat{\theta}(t + 1) = \hat{\theta}(t) - F \frac{\delta J(t + 1)}{\delta \hat{\theta}(t + 1)}
\]

\[
F = \alpha I \quad (\alpha > 0) \quad (I = \text{unit matrix})
\]

(\(\alpha\) – adaptation gain/learning rate)

The Gradient of \(J\):

(with respect to \(\theta\))

\[
\frac{1}{2} \nabla_{\theta} J = \frac{1}{2} \frac{\partial J(t+1)}{\partial \hat{\theta}(t+1)} = \frac{\partial \varepsilon(t+1)}{\partial \varepsilon(t+1)} \varepsilon(t + 1) = -\phi(t) \varepsilon(t + 1)
\]

\[
\varepsilon(t + 1) = y(t + 1) - \hat{y}(t+1) = y(t + 1) - \hat{\theta}^T(t + 1) \phi(t)
\]

\[
\frac{\partial \varepsilon(t+1)}{\partial \hat{\theta}(t+1)} = -\phi(t)
\]

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
PAA – Gradient algorithm (steepest descent)

Criterion to be minimized (objective):

$$\min_{\theta(t+1)} J(t+1) = [\varepsilon(t+1)]^2$$

**Gradient strategy:**

$$\hat{\theta}(t+1) = \hat{\theta}(t) - F \frac{\delta J(t+1)}{\delta \hat{\theta}(t+1)}$$

$$F = \alpha I \quad (\alpha > 0) \quad (I = \text{unit matrix})$$

$$(\alpha – 	ext{adaptation gain/learning rate})$$

The Gradient of $J$:

$$\frac{1}{2} \nabla_{\theta} J = \frac{1}{2} \frac{\partial J(t+1)}{\partial \hat{\theta}(t+1)} = \frac{\partial \varepsilon(t+1)}{\partial \hat{\theta}(t+1)}$$

$$\varepsilon(t+1) = -\phi(t)\varepsilon(t+1)$$

$$\varepsilon(t+1) = y(t+1) - \hat{y}(t+1) = y(t+1) - \hat{\theta}^T(t+1)\phi(t)$$

$$\frac{\partial \varepsilon(t+1)}{\partial \hat{\theta}(t+1)} = -\phi(t)$$

$$\hat{\theta}(t+1) = \hat{\theta}(t) + F\phi(t)\varepsilon(t+1)$$

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
PAA – Gradient algorithm (steepest descent)

Criterion to be minimized (objective):
\[
\min_{\theta(t+1)} J(t+1) = \left[ \varepsilon(t+1) \right]^2
\]

Gradient strategy: (*) \( \hat{\theta}(t+1) = \hat{\theta}(t) - F \frac{\delta J(t+1)}{\delta \theta(t+1)} \)  
\[ F = \alpha I \quad (\alpha > 0) \quad (I = \text{unit matrix}) \]

\( \alpha \) – adaptation gain/learning rate

The Gradient of \( J \):  
(\text{with respect to} \ \theta)  
\[
\frac{1}{2} \nabla_{\theta} J = \frac{1}{2} \frac{\partial J(t+1)}{\partial \theta(t+1)} = \frac{\partial \varepsilon(t+1)}{\partial \theta(t+1)} \varepsilon(t+1) = -\phi(t)\varepsilon(t+1)
\]

\( \varepsilon(t+1) = y(t+1) - \hat{y}(t+1) = y(t+1) - \hat{\theta}^T(t+1)\phi(t) \)

\[
(*) \quad \hat{\theta}(t+1) = \hat{\theta}(t) + F\phi(t)\varepsilon(t+1)
\]

\( \varepsilon(t+1) = \frac{\varepsilon^0(t+1)}{1 + \phi(t)^T F\phi(t)} \)

\( \hat{\theta}(t+1) = \hat{\theta}(t) + \frac{F\phi(t)\varepsilon^0(t+1)}{1 + \phi(t)^T F\phi(t)} \)

Stable for any \( F > 0 \) (\( \alpha > 0 \))

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Even for the estimation of the parameters of a static model, the adaptation/learning algorithm introduces a dynamic feedback

\[ y(t + 1) = bu(t) \quad \text{Unknown model} \]

\[ \hat{y}(t + 1) = \hat{b}(t + 1)u(t) \quad \text{Adjustable predictor model (a posteriori)} \]

\[ \epsilon(t + 1) = y(t + 1) - \hat{y}(t + 1) = \left[ b - \hat{b}(t + 1) \right]u(t) \quad (*) \quad \text{a posteriori Prediction error} \]

\[ \hat{b}(t + 1) = \hat{b}(t) + \alpha u(t) \epsilon(t + 1); \quad \alpha > 0 \quad (** \text{) Adaptation/learning alg.} \]
Parameter adaptation/learning algorithms: a feedback interpretation

Even for the estimation of the parameters of a static model, the adaptation/learning algorithm introduces a dynamic feedback

\[ y(t + 1) = bu(t) \quad \text{Unknown model} \]

\[ \hat{y}(t + 1) = \hat{b}(t + 1)u(t) \quad \text{Adjustable predictor model (a posteriori)} \]

\[ \varepsilon(t + 1) = y(t + 1) - \hat{y}(t + 1) = [b - \hat{b}(t + 1)]u(t) \quad (*) \quad \text{a posteriori Prediction error} \]

\[ \hat{b}(t + 1) = \hat{b}(t) + \alpha u(t)\varepsilon(t + 1); \quad \alpha > 0 \quad (**) \quad \text{Adaptation/learning alg.} \]

\[ \tilde{b}(t + 1) = \hat{b}(t + 1) - b \quad \text{Parameter error} \]

\[ \varepsilon(t + 1) = -\tilde{b}(t + 1)u(t) \quad (*) \]

\[ \tilde{b}(t + 1) = \tilde{b}(t) + \alpha u(t)\varepsilon(t + 1) \quad (**) \quad \text{Feedback system} \]
Parameter adaptation/learning algorithms: a feedback interpretation

Even for the estimation of the parameters of a static model, the adaptation/learning algorithm introduces a dynamic feedback

\[ y(t + 1) = bu(t) \quad \text{Unknown model} \]

\[ \hat{y}(t + 1) = \hat{b}(t + 1)u(t) \quad \text{Adjustable predictor model (a posteriori)} \]

\[ \varepsilon(t + 1) = y(t + 1) - \hat{y}(t + 1) = [b - \hat{b}(t + 1)]u(t) \quad (*) \quad \text{a posteriori Prediction error} \]

\[ \hat{b}(t + 1) = \hat{b}(t) + \alpha u(t)\varepsilon(t + 1); \quad \alpha > 0 \quad (**) \quad \text{Adaptation/learning alg.} \]

\[ \tilde{b}(t + 1) = \hat{b}(t + 1) - b \quad \text{Parameter error} \]

\[ \varepsilon(t + 1) = -\tilde{b}(t + 1)u(t) \quad (*) \]

\[ \tilde{b}(t + 1) = \tilde{b}(t) + \alpha u(t)\varepsilon(t + 1) \quad (**) \]

Feedback system

There is an associated asymptotic stability problem

(Will the system be as. stable for all values of the adaptation gain/learning rate \( \alpha > 0 \) ?)

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Parameter adaptation/learning algorithms: a feedback interpretation

\[ \hat{\theta}(t+1) = \hat{\theta}(t) + F\phi(t)\varepsilon(t+1) \]

\[ \tilde{\theta}(t) = \hat{\theta}(t) - \theta \]

\[ \phi(t)^T \hat{\theta}(t+1) = \phi(t)^T \hat{\theta}(t) + \phi(t)^TF\phi(t)\varepsilon(t+1) \]

\[ \varepsilon(t+1) = y(t+1) - \hat{y}(t+1) = [\theta - \hat{\theta}(t+1)]^T \phi(t) = -\tilde{\theta}^T \phi(t) \]
Parameter adaptation/learning algorithms: a feedback interpretation

\[ \hat{\theta}(t+1) = \hat{\theta}(t) + F\phi(t)\varepsilon(t+1) \]
\[ \tilde{\theta}(t) = \hat{\theta}(t) - \theta \]

\[ \phi(t)^T \tilde{\theta}(t+1) = \phi(t)^T \hat{\theta}(t) + \phi(t)^T F\phi(t)\varepsilon(t+1) \]
\[ \varepsilon(t+1) = y(t+1) - \tilde{y}(t+1) = \left[\theta - \hat{\theta}(t+1)\right]^T \phi(t) = -\tilde{\theta}^T \phi(t) \]

Since the equivalent feedforward path is characterized by a strictly positive real transfer function (it is strictly passive) the feedback system is Globally Asymptotically Stable for all the equivalent feedback paths which are passive.

(based on V. M. Popov – Hyperstability, 1962)
An old result (1972)

The system (input \( u(t) \) – output \( y(t) \)) is passive provided that \( H(z^{-1}) \) is a positive real transfer function

\[
\sum_{0}^{t_1} y(t, \tau) u(t) \geq -\gamma^2 \; ; \; \gamma^2 < \infty \; ; \; \forall \; t \geq 0 \; (\tau \leq t)
\]

**Positive real transfer function** : stable, \(-90^\circ \leq \text{input-output phase lag/advance} \leq 90^\circ\)
A Transfer Operator interpretation of the Gradient Algorithm

\[
\hat{\theta}(t+1) = \hat{\theta}(t) + F \phi(t) \varepsilon(t+1)
\]

\[F = \alpha I \quad (\alpha > 0)\]

\[
\hat{\theta}(t+1) = \hat{\theta}(t) - F \nabla_{\theta} J(t+1)
\]

Gradient of J with respect to \(\theta\)

\[
\hat{\theta}(t+1) = \frac{1}{1-q^{-1}} \alpha \phi(t)\varepsilon(t+1)
\]
A Transfer Operator interpretation of the Gradient Algorithm

\[ \hat{\theta}(t+1) = \hat{\theta}(t) + F \phi(t) \epsilon(t+1) \]

\[ F = \alpha I \quad (\alpha > 0) \]

Gradient of \( J \) with respect to \( \theta \)

- It is an « integrator » filter
- It is a passive system
- Characterized by a positive real transfer function
- \(-90^\circ \leq \text{input-output phase lag/advance} \leq 90^\circ\)
A Transfer Operator interpretation of the Gradient Algorithm

\[
\hat{\theta}(t+1) = \hat{\theta}(t) + F \phi(t) \varepsilon(t+1)
\]

\[
F = \alpha I \quad (\alpha > 0)
\]

\[
\hat{\theta}(t+1) = \hat{\theta}(t) - F \nabla_{\theta} J(t+1)
\]

Gradient of J with respect to \(\theta\)

- It is an « integrator » filter
- It is a passive system
- Characterized by a positive real transfer function
- \(-90^\circ \leq \text{input-output phase lag/advance} \leq 90^\circ\)

Adaptation gain/learning rate

\[
\hat{\theta}(t+1) = \frac{1}{1-q^{-1}} \alpha [-\nabla_{\theta} J(t+1)]
\]

MIMO diagonal transfer operator with identical terms (\(H^{ii}(q^{-1}) = \frac{1}{1-q^{-1}}\) for gradient alg.)

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Main Result

\[ \hat{\theta}(t+1) = H_{PAA}(q^{-1})\alpha[ - \nabla_{\theta} J(t+1)] \]

\[ H_{PAA} = \begin{bmatrix} H_{11} & H_{ii} \\ H_{ii} & H_{nn} \end{bmatrix} \]

Objective: \( \lim_{t \to \infty} \varepsilon(t+1) = 0 \) for any initial conditions \( \theta(0), \varepsilon(0) \)

The global asymptotic stability is assured for any positive adaptation gain/learning rate provided that the H_{ii} operators are characterized by a positive real transfer function with a pole at z=1 (for memory)
Main Result

\[ \hat{\theta}(t + 1) = H_{PA\text{A}}(q^{-1})\alpha[-\nabla_\theta J(t + 1)] \]

\[ H_{PA\text{A}} = \begin{bmatrix} H_{11} \\ H_{ii} \\ H_{nn} \end{bmatrix} \]

Objective: \( \lim_{t \to \infty} \epsilon(t + 1) = 0 \) for any initial conditions \( \theta(0), \epsilon(0) \)

The global asymptotic stability is assured for any positive adaptation gain/learning rate provided that the \( H_{ii} \) operators are characterized by a positive real transfer function with a pole at \( z=1 \) (for memory)

\[ H_{ii}(q^{-1}) = \frac{1 + c_1q^{-1} + c_2q^{-2} + \ldots + c_{nc}q^{-nc}}{(1 - q^{-1})(1 - d_1'q^{-1} - d_2'q^{-2} - \ldots - d_{nd}'q^{-nd}')} \]

\[ = \frac{C(q^{-1})}{(1 - q^{-1})D'(q^{-1})} = \frac{C(q^{-1})}{D(q^{-1})} = \frac{1 + c_1q^{-1} + c_2q^{-2} + \ldots + c_{nc}q^{-nc}}{1 - d_1q^{-1} - d_2q^{-2} - \ldots - d_{nd}q^{-nd}} \]

\[ d_i = (d_i' - d_{i-1}'); i = 1, \ldots, n_D; d_0' = -1, d_{nd}' = 0 \]

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
ARIMA2 (I+P+DF) Algorithm (new)

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})} \]

\[ \hat{\theta}(t + 1) = d_1 \hat{\theta}(t) + d_2 \hat{\theta}(t - 1) + F[\phi(t)\varepsilon(t + 1) + c_1 \phi(t - 1)\varepsilon(t) + c_2 \phi(t - 2)\varepsilon(t - 1)] \]

\( d_1 = (1 + d'_1); \quad d_2 = -d'_1 \)

(to assure the presence of the integrator)
ARIMA2 (I+P+DF) Algorithm (new)

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})} \]

\[
\hat{\theta}(t + 1) = d_1 \hat{\theta}(t) + d_2 \hat{\theta}(t - 1) + F[\phi(t)\varepsilon(t + 1) + c_1 \phi(t - 1)\varepsilon(t) + c_2 \phi(t - 2)\varepsilon(t - 1)]
\]

\[ d_1 = (1 + d'_1); \quad d_2 = -d'_1 \] (to assure the presence of the integrator)

If we want to guarantee stability for any value of the adaptation gain/learning rate:

The weights: \(d'_1, c_1, c_2\) should be chosen such that:

\(H^{ii}\) be characterized by a positive real transfer function
ARIMA2 (I+P+DF) Algorithm (new)

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} \]

\[ = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})} \]

\[ \hat{\theta}(t + 1) = d_1 \hat{\theta}(t) + d_2 \hat{\theta}(t - 1) + F[\phi(t)\epsilon(t + 1) + c_1 \phi(t - 1)\epsilon(t) + c_2 \phi(t - 2)\epsilon(t - 1)] \]

\[ d_1 = (1 + d'_1); \quad d_2 = -d'_1 \]

(to assure the presence of the integrator)

If we want to guarantee stability for any value of the adaptation gain/learning rate:

The weights: \( d'_1, c_1, c_2 \) should be chosen such that:

\( H^{ii} \) be characterized by a **positive real** transfer function

Define:

\[ \delta = \frac{1 + c_1 + c_2}{1 - d'_1}; \quad \gamma = \frac{d'_1 c_1 + d'_2 + c_2}{d'_1 - 1} \]

**PR conditions** (B. Vau):

\[ -1 < d'_1 < 1; \quad 0 \leq \delta \leq 2; \quad -1 \leq d'_1 - \frac{\gamma}{1 - \delta/2} \leq 1 \]
This algorithm can be interpreted as an *Integral + Proportional + Filtered Derivative* adaptation algorithm:

\[
H_{ii}(q^{-1}) = \frac{\alpha_I}{1 - q^{-1}} + \alpha_P \frac{(1 - q^{-1})}{(1 - d'_1 q^{-1})} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})}
\]

Bringing (*) to a common denominator, one gets:

\[
c_1 = -\frac{\alpha_I d'_1 - \alpha_P (1 + d'_1) - 2\alpha_D}{\alpha_T} \quad c_2 = \frac{d'_1 \alpha_P + \alpha_D}{\alpha_T} \quad \alpha_T = \alpha_I + \alpha_P + \alpha_D
\]
This algorithm can be interpreted as an **Integral + Proportional + Filtered Derivative** adaptation algorithm.

\[
H_{ii}(q^{-1}) = \frac{\alpha_I}{1 - q^{-1}} + \alpha_P + \alpha_D \frac{(1 - q^{-1})}{(1 - d'q^{-1})} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'q^{-1})} \tag{*}
\]

Bringing (*) to a common denominator, one gets:

\[
c_1 = \frac{-\alpha_I d'_1 - \alpha_P (1 + d'_1) - 2 \alpha_D}{\alpha_T} \quad c_2 = \frac{d'_1 \alpha_P + \alpha_D}{\alpha_T} \quad \alpha_T = \alpha_I + \alpha_P + \alpha_D
\]

Conversely: given \( c_1, c_2, d'_1 \) one gets:

\[
\alpha_I = \frac{1 + c_1 + c_2}{1 - d'_1} \quad \alpha_P = -\frac{c_1 + c_2 (2 - d'_1) + d'_1}{(1 - d'_1)^2} \quad \alpha_D = c_2 - \alpha_P d'_1 \quad \alpha_T = \alpha_I + \alpha_P + \alpha_D = 1
\]

The new value of the estimated parameters is a weighted sum (with minus sign) of:

- the integral of the gradient,
- the gradient and its filtered derivative

\[
\hat{\theta}(t + 1) = H_{PAA}(q^{-1}) \alpha [-\nabla_{\theta} J(t + 1)]
\]
Is the PR condition necessary when operating with small adaptation gain/learning rate?

**Answer:** No. There are however signal dependent relaxed conditions to be fulfilled.

![Graph showing phase response](image)

H\textsubscript{ii} can be « non positive real » in some frequency regions provided that the regions where it is « positive real » exceed those where it is not PR.

( the averaged input/output energy should be positive)

An example:

Assume that the excitation signal cover the region from 0.1 f/fs to 0.4 f/fs with an almost constant energy.

The system H2 is non PR from 0.1f/fs to 0.17f/fs and PR from 0.17 to 0.4.f/fs. It will works for small adaptation gains.

One can say that is « PR in the average »
Review of PALA algorithms

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})} \]

IMA

- Integral + Proportional: \( c_1 \neq 0; c_2 = 0; d'_1 = 0 \)

- Int. + Prop. + Derivative: \( c_1 \neq 0; c_2 \neq 0; d'_1 = 0 \)

- Averaged gradient: \( c_1 \neq 0; c_2 \neq 0; d'_1 = 0 \) (\( c_i \neq 0 \))
Review of PALA algorithms

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})} \]

IMA

- Integral + Proportional: \( c_1 \neq 0; c_2 = 0; d'_1 = 0 \)
- Int. + Prop. + Derivative: \( c_1 \neq 0; c_2 \neq 0; d'_1 = 0 \)
- Averaged gradient: \( c_1 \neq 0; c_2 \neq 0; d'_1 = 0 \) (\( c_i \neq 0 \))

ARI

- Conjugate gradients: \( c_1 = 0; c_2 = 0; d'_1 \neq 0 \)
- Nesterov Algorithm: \( c_1 = 0; c_2 = 0; d'_1 \neq 0 \)
- Momentum back propagation: \( c_1 = 0; c_2 = 0; d'_1 \neq 0 \)
  \[ \alpha' = \alpha(1 - d'_1) \]

Leakage algorithm: \( H^{ii}(q^{-1}) = \frac{1}{1 - \sigma q^{-1}} \); \( 0 < \sigma < 1 \)
Review of PALA algorithms

\[
H^ii(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{1 - d_1 q^{-1} - d_2 q^{-2}} = \frac{1 + c_1 q^{-1} + c_2 q^{-2}}{(1 - q^{-1})(1 - d'_1 q^{-1})}
\]

IMA

- Integral + Proportional: \( c_1 \neq 0; c_2=0; d'_1=0 \)
- Int. + Prop. + Derivative: \( c_1 \neq 0; c_2 \neq 0; d'_1=0 \)
- Averaged gradient: \( c_1 \neq 0; c_2 \neq 0; d'_1=0 \) (\( c_i \neq 0 \))

ARI

- Conjugate gradients: \( c_1 = 0; c_2=0; d'_1 \neq 0 \)
- Nesterov Algorithm: \( c_1 = 0; c_2=0; d'_1 \neq 0 \)
- Momentum back propagation: \( c_1 = 0; c_2=0; d'_1 \neq 0 \)
  \( \alpha' = \alpha(1 - d'_1) \)

Leakage algorithm: \( H^ii(q^{-1}) = \frac{1}{1 - \sigma q^{-1}} ; 0 < \sigma < 1 \)

ARIMA2 can be viewed as a combination of I+P+D and Conjugate gradients

**Challenge**: Find a parameter adaptation/learning algorithm which does not have an ARMA structure.
Estimation of the parameters of: 

\[ S = \frac{q^{-2} + 0.5q^{-3}}{1 - 1.5q^{-1} + 0.7q^{-2}} \]  

(Input: PRBS)

Performance indices:

\[ J_\varepsilon(N) = \sum_{t=0}^{N} \varepsilon^2(t + 1) \]

\[ D^2(t) = \left\{ [\theta - \hat{\theta}(t)]^T [\theta - \hat{\theta}(t)] \right\} \]

\[ J_D(N) = \sum_{t=0}^{N} D^2(t) \]

\[ \alpha = 0.1 \quad \text{(adaptation gain)} \]

Simulation Results

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PR</th>
<th>c_1</th>
<th>c_2</th>
<th>d'_1</th>
<th>J_D(N)</th>
<th>J_\varepsilon(N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integral (gradient)</td>
<td>Y</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>51.65</td>
<td>13.32</td>
</tr>
<tr>
<td>Conj.Gr/Nest.</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>37.15</td>
<td>12.09</td>
</tr>
<tr>
<td>I+P+D (\alpha_P = -2\alpha_D)</td>
<td>N</td>
<td>0</td>
<td>0.99</td>
<td>0</td>
<td>34.58</td>
<td>11.95</td>
</tr>
<tr>
<td>I+P</td>
<td>Y</td>
<td>0.667</td>
<td>0</td>
<td>0</td>
<td>41.41</td>
<td>12.45</td>
</tr>
<tr>
<td>ARIMA 2</td>
<td>N</td>
<td>-0.5</td>
<td>0.4</td>
<td>0.7</td>
<td>26.62</td>
<td>9.67</td>
</tr>
</tbody>
</table>
Simulation results – Stability issues

I+P alg.

ARIMA2 alg.

PR in the average

Unstable!

Stable!

α = adaptation gain

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Simulation results under the « positive real » constraint

- The improvement in performance is less significant
- Small differences in performance between various algorithms
- Are these weights values the best?
Simulation results – Imperfect matching

\[ S = \frac{q^{-2} + 0.5q^{-3}}{1 - 1.5q^{-1} + 0.7q^{-2}} \]

\[ \hat{S} = \frac{\hat{b}_2 q^{-2}}{1 + \hat{a}_1 q^{-1} + \hat{a}_2 q^{-2}} \]

The prediction (adaptation) error and the parametric distance will no go to zero!

Simulation results for different models:
- ARIMA2
- I+P
- I+P+D
- Conj. grad.

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Experimental Results: Adaptive Feedforward Noise Attenuation

- Broad band noise disturbance (70 – 170 Hz)
- Youla-Kucera parametrized feedforward compensator
- Q- adjustable FIR filter (60 parameters)
- Adaptation gain/learning rate: $\alpha = 0.2$

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Experimental Results: Adaptive Feedforward Noise Attenuation

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Experimental Results: Adaptive Feedforward Noise Attenuation

I.D. Landau & al. "Does it exists a general form for adaptation/learning algorithms"
Experimental Results: Adaptive Feedforward Noise Attenuation

I. D. Landau & al. "Does it exist a general form for adaptation/learning algorithms"
A new concept: *Dynamic adaptation gain/learning rate* (frequency dependend adaptation gain/learning rate)

\[
\hat{\theta}(t+1) = H_{PAA}(q^{-1})\alpha[-\nabla_{\theta} J(t+1)] \\
H_{PAA} = \begin{bmatrix}
H_{11} & H_{ii} \\
H_{ii} & H_{nn}
\end{bmatrix}
\]

\[
H^{ii}(q^{-1}) = \frac{C(q^{-1})}{(1-q^{-1})D'(q^{-1})}
\]

\[
\hat{\theta}(t+1) = \hat{\theta}(t) + \alpha \frac{C(q^{-1})}{D'(q^{-1})}[-\nabla_{\theta} J(t+1)]
\]

Dynamic adaptation gain/learning rate

If \(C\) and \(D'\) have all the zeros inside the unit circle ➔ average gain=0!
Concluding Remarks

• The « gradient » classical strategy can be generalized (the integrator is replaced by an ARIMA filter)

• Stability of the adaptive/learning scheme for any magnitude of the adaptation gain/learning rate requires that this embedded filter be characterized by a positive real transfer function

• For slow adaptation/learning the above condition can be relaxed

• The algorithms using an ARIMA filter can improve the performance of the « gradient » algorithm

• The optimal choice of the coefficients of this embedded filter is a partial open problem
Concluding Remarks

- The « gradient » classical strategy can be generalized (the integrator is replaced by an ARIMA filter)

- Stability of the adaptive/learning scheme for any magnitude of the adaptation gain/learning rate requires that this embedded filter be characterized by a positive real transfer function

- For slow adaptation/learning the above condition can be relaxed

- The algorithms using an ARIMA filter can improve the performance of the « gradient » algorithm

- The optimal choice of the coefficients of this embedded filter is a partial open problem

- The results hold also for « continous time » formulation of adaptive/learning systems

- When using « approximations » of the gradient as « correcting » term, an additional SPR condition has to be satisfied for stability of the adaptive/learning system
Thank you for your attention!
Implementation

\[ H^{ii}(q^{-1}) = \frac{1 + c_1 q^{-1} + c_2 q^{-2} + \ldots + c_{nC} q^{-n_C}}{1 - d_1 q^{-1} - d_2 q^{-2} - \ldots - d_{n_D} q^{-n_D}} = \frac{C(q^{-1})}{D(q^{-1})} \]

\[
\hat{\theta}(t + 1) = d_1 \hat{\theta}(t) + d_2 \hat{\theta}(t - 1) + \ldots + d_{n_D} \hat{\theta}(t - n_D) \\
+ F[\phi(t)\varepsilon(t + 1) + c_1 \phi(t - 1)\varepsilon(t) + c_2 \phi(t - 2)\varepsilon(t - 1) \\
+ \ldots + c_{n_C} \phi(t - n_C)\varepsilon(t - n_C + 1)]
\]

Gradient algorithm \((d_1 = 1)\)
Estimation of the parameters of:

\[ S = \frac{q^{-2} + 0.5q^{-3}}{1 - 1.5q^{-1} + 0.7q^{-2}} \]  

(Input: PRBS)

Performance indices:

\[ J_\varepsilon(N) = \sum_{t=0}^{N} \varepsilon^2(t + 1) \]

\[ D^2(t) = \{[\theta - \hat{\theta}(t)]^T [\theta - \hat{\theta}(t)]\} \]

\[ J_D(N) = \sum_{t=0}^{N} D^2(t) \]

\[ \alpha = 0.1 \]  (adaptation gain)

\[
\begin{array}{|c|c|c|c|c|c|c|}
\hline
\text{Algorithm} & \text{PR} & c_1 & c_2 & d'_1 & J_D(N) & J_\varepsilon(N) \\
\hline
\text{Integral (gradient)} & Y & 0 & 0 & 0 & 51.65 & 13.32 \\
\text{Conj. Gr/Nest..} & N & 0 & 0 & 0 & 37.15 & 12.09 \\
\text{I+P+D} (\alpha_P = -2\alpha_D) & N & 0 & 0.99 & 0 & 34.58 & 11.95 \\
\text{I+P} & Y & 0.667 & 0 & 0 & 41.41 & 12.45 \\
\text{ARIMA 2} & N & -0.5 & 0.4 & 0.7 & 26.62 & 9.67 \\
\hline
\end{array}
\]

\( (\alpha_I=1; \alpha_P=-2.66, \alpha_D=2) \)
Simulation results under the « positive real » constraint

- The improvement in performance is less significant
- Small differences in performance between various algorithms
- Are these weights values the best?

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PR</th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$d'_1$</th>
<th>$J_D(N)$</th>
<th>$J_e(N)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integral</td>
<td>Y</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>51.65</td>
<td>13.32</td>
</tr>
<tr>
<td>Conj.Gr/Nest..</td>
<td>Y</td>
<td>0.1</td>
<td>0.333</td>
<td>0.333</td>
<td>42.16</td>
<td>11.99</td>
</tr>
<tr>
<td>I+P+D</td>
<td>Y</td>
<td>0.667</td>
<td>0</td>
<td>0</td>
<td>42.91</td>
<td>12.04</td>
</tr>
<tr>
<td>I+P</td>
<td>Y</td>
<td>0</td>
<td>0.33</td>
<td>0</td>
<td>41.41</td>
<td>12.45</td>
</tr>
<tr>
<td>I+P+D/Av.Gr</td>
<td>Y</td>
<td>0.0989</td>
<td>0.0789</td>
<td>0.22</td>
<td>41.96</td>
<td>11.99</td>
</tr>
<tr>
<td>ARIMA 2</td>
<td>Y</td>
<td>0.408</td>
<td>-0.032</td>
<td>0.2</td>
<td><strong>40.59</strong></td>
<td>12.39</td>
</tr>
</tbody>
</table>

($\alpha_1=1; \alpha_p=-0.5; \alpha_D=0.081$)