

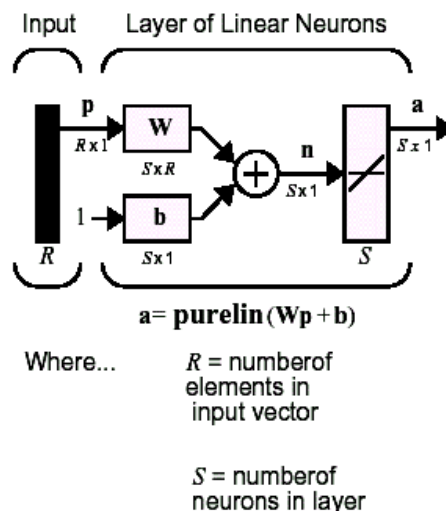
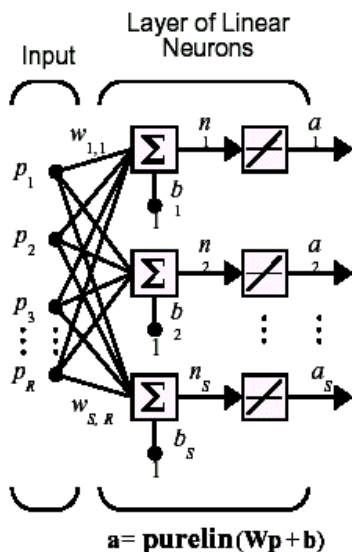
## BAB VI

### METODE BELAJAR WIDROW-HOFF

- Aturan belajar LMS (*Least Mean Squares*) lebih efektif dari aturan belajar perseptron.
- Aturan belajar LMS atau Widrow-Hoff meminimisasikan *mean square error*, sehingga menggeser batasan keputusan sejauh yang bisa dilakukan dari pola latihan

#### 6.1 Jaringan ALDALINE

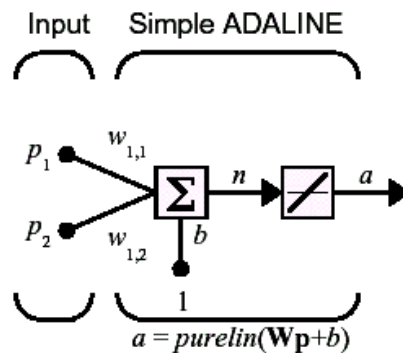
- Jaringan ADALINE (*Adaptive Linear Neuron*) mirip dengan perseptron, tapi ADALINE menggunakan fungsi transfer linier alih-alih *hard-limiting*. Dengan demikian, keluarannya bisa beragam (perseptron hanya 0 atau 1).
- ADALINE dan perseptron hanya bisa menyelesaikan masalah yang bersifat bebas linier.
- *Adaptive Linear Neuron* merespons perubahan lingkungannya pada saat beroperasi
- Jaringan ADALINE di bawah ini memiliki satu lapisan dengan  $S$  neuron yang terhubung melalui vektor matriks  $\mathbf{W}$ .



- Jaringan ini kadang-kadang disebut MADALINE (**M**any **A**DALINEs)
- Aturan Widrow-Hoff hanya dapat melatih jaringan linier satu lapis. Ini tidak sepenuhnya merupakan kelemahan, karena jaringan linier satu lapispun bisa memiliki kemampuan yang sama dengan jaringan linier berlapis banyak. Untuk setiap jaringan linier berlapis banyak terdapat ekuivalen berupa jaringan linier satu lapis.

### ADALINE tunggal (Single ADALINE)

- ADALINE dengan dua masukan. Diagram jaringan ini adalah sbb. :

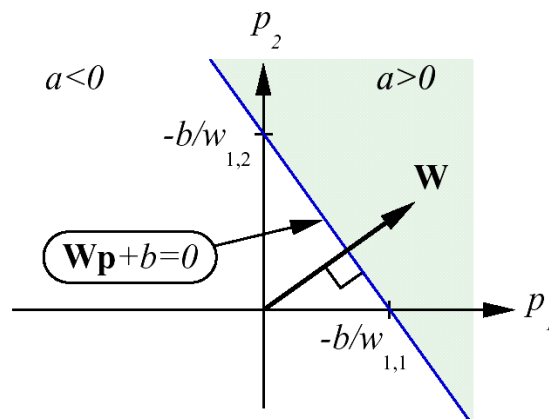


Bobot matriks **W** hanya memiliki satu baris. Keluaran neuron adalah

$$a = \text{purelin}(n) = \text{purelin}(\mathbf{Wp} + b) = \mathbf{Wp} + b \quad \text{atau}$$

$$a = w_{1,1} p_1 + w_{1,2} p_2 + b$$

- Seperti halnya perseptron, ADALINE memiliki *decision boundary* (batasan keputusan) yang ditentukan oleh vektor-vektor input yang menghasilkan input jaringan  $\eta$  bernilai 0. Dengan  $\eta = 0$ , diperoleh persamaan  $\mathbf{Wp} + b = 0$ , yang akan menghasilkan *decision boundary* sbb. :



## 6.2 Mean Square Error

- Seperti juga aturan belajar perseptron, algoritma *least mean square* (LMS) adalah contoh pembelajaran tersupervisi, dimana aturan belajar dijalankan dengan bantuan satu set contoh hubungan masukan-keluaran yang diinginkan :

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \dots, \{\mathbf{p}_Q, \mathbf{t}_Q\}$$

Dengan kata  $\mathbf{p}_q$  sebagai masukan jaringan dan  $\mathbf{t}_q$  sebagai target keluaran.

- Jika keluaran jaringan berbeda dengan target, dilakukan perhitungan *mse* (*mean square error*) terhadap perbedaan keluaran dan target :

$$mse = \frac{1}{Q} \sum_{k=1}^Q e(k)^2 = \frac{1}{Q} \sum_{k=1}^Q (t(k) - a(k))^2$$

- Algoritma LMS mengatur nilai bobot dan bias ADALINE agar menghasilkan *mse* sekecil mungkin.

## 6.3 Algoritma Least Mean Square (LMS)

- Algoritma belajar LMS atau Widrow-Hoff berbasis pada pendekatan prosedur *steepest-descent* (penurunan tercepat).
- Menurut Widrow-Hoff, *mse* dapat diestimasi melalui kwadrat kesalahan (*squared error*) pada setiap iterasi. Derivatif pertama dari *squared error* terhadap bobot dan bias pada iterasi ke- $k$  adalah :

$$\frac{\partial e^2(k)}{\partial w_{1,j}} = 2e(k) \frac{\partial e(k)}{\partial w_{1,j}}$$

dengan  $j = 1, 2, \dots, R$  dan

$$\frac{\partial e^2(k)}{\partial b} = 2e(k) \frac{\partial e(k)}{\partial b}$$

Derivasi parsial untuk *error* adalah :

$$\frac{\partial e(k)}{\partial w_{1,j}} = \frac{\partial [t(k) - a(k)]}{\partial w_{1,j}} = \frac{\partial}{\partial w_{1,j}} [t(k) - (\mathbf{W}\mathbf{p}(k) + b)] \quad \text{atau}$$

$$\frac{\partial e(k)}{\partial w_{1,j}} = \frac{\partial}{\partial w_{1,j}} \left[ t(k) - \left( \sum_{i=1}^R w_{1,i} p_i(k) + b \right) \right]$$

Di sini  $p_i(k)$  adalah elemen ke-i dari vektor input pada iterasi ke-k.

Demikian juga

$$\frac{\partial e(k)}{\partial w_{1,j}} = -p_j(k)$$

dapat disederhanakan menjadi :

$$\frac{\partial e(k)}{\partial w_{1,j}} = -p_j(k) \text{ dan } \frac{\partial e(k)}{\partial b} = -1$$

Akhirnya dilakukan perubahan matriks bobot dan bias sbb. :

$$2\alpha e(k) p(k) \text{ dan } 2\alpha e(k)$$

Ini merupakan basis algoritma belajar Widrow-Hoff (LMS).

- Persamaan di atas dapat dikembangkan untuk neuron majemuk :

$$\mathbf{W}(k+1) = \mathbf{W}(k) + 2\alpha e(k) \mathbf{p}^T(k)$$

$$\mathbf{b}(k+1) = \mathbf{b}(k) + 2\alpha e(k)$$

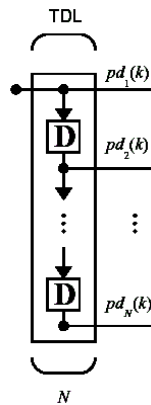
Kesalahan / galat (error)  $\mathbf{e}$  dan bias  $\mathbf{b}$  adalah vektor, dan  $\alpha$  adalah *learning rate*. Jika  $\alpha$  besar, proses belajar berlangsung cepat, tetapi  $\alpha$  yang terlalu besar akan menimbulkan ketidakstabilan dan galatpun akan semakin tinggi.

## 6.4 Adaptive Filtering

- ADALINE pada saat ini merupakan salah satu jaringan neural yang paling banyak digunakan. Salah satu aplikasi utamanya adalah *adaptive filtering*.

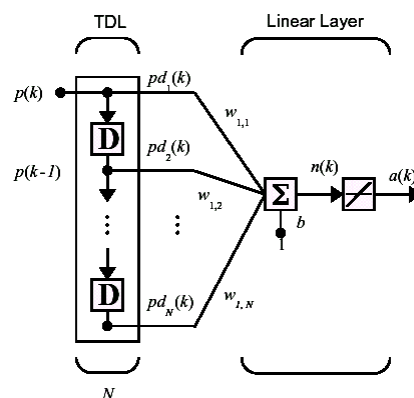
### Tapped Delay Line

Untuk dapat memanfaatkan jaringan ADALINE secara penuh, diperlukan satu komponen baru, yaitu *tapped delay line*. Pada gambar di bawah ini, sinyal masuk dari kiri, melalui  $N - 1$  buah *delay*. Keluaran dari *tapped delay line* (TDL) berupa vektor  $N -$  dimensi, yang terbentuk dari sinyal masukan pada saat itu, sinyal masukan sebelumnya, dll.



### Filter adaptif

TDL dapat digabungkan dengan ADALINE, membentuk filter adaptif sebagai berikut :

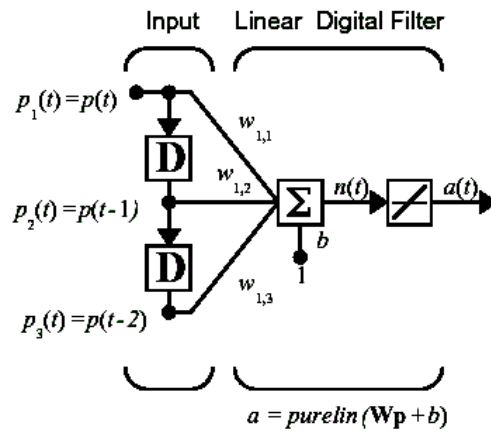


Keluaran filter ini adalah :

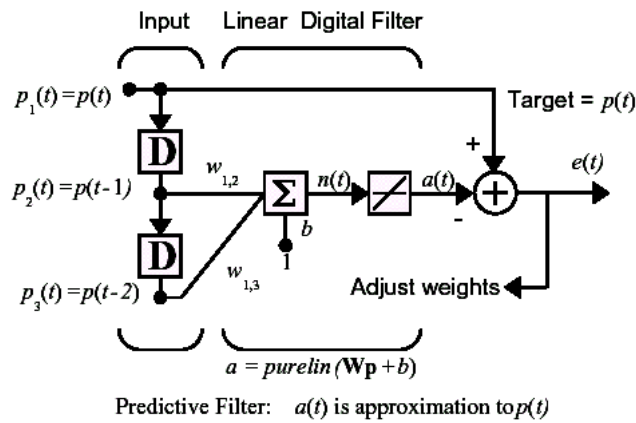
$$a(k) = \text{purelin}(\mathbf{W}\mathbf{p} + b) = \sum_{i=1}^R w_{1,i} a(k - i + 1) + b$$

Pada bidang *signal processing*, filter ini dikenal sebagai *Finite Impulse Response* (FIR).

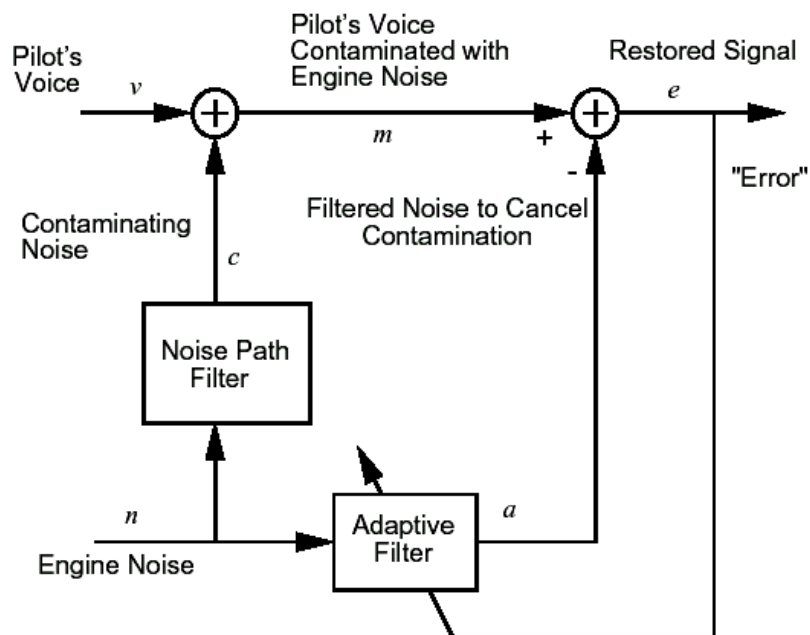
### Contoh Filter Adaptif



### Contoh Filter (adaptif) prediktif

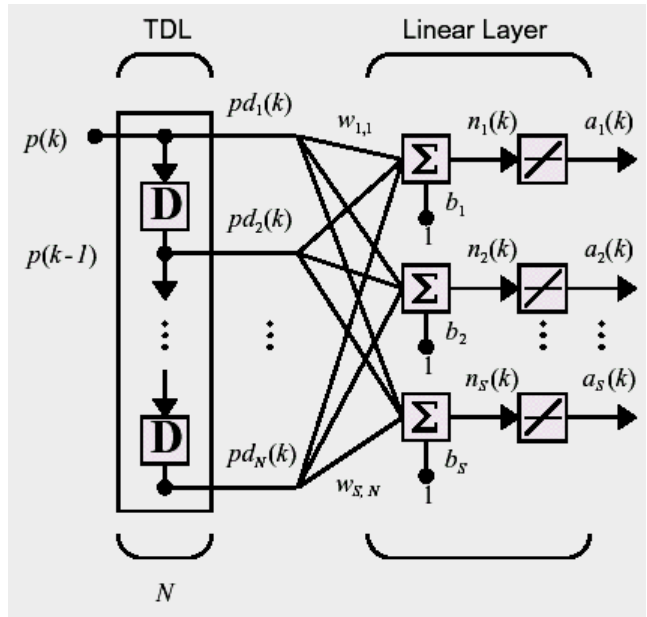


### Contoh Noise Cancellation

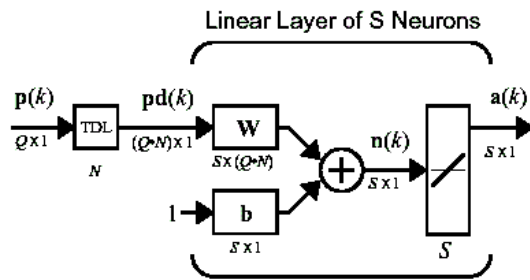


Adaptive Filter Adjusts to Minimize Error. This removes the engine noise from contaminated signal, leaving the pilot's voice as the "error."

### Filter Adaptif Neuron Majemuk



### Bentuk ringkas Filter Adaptif



### Specific Small Adaptive Filter

#### Abbreviated Notation

