SISTEM NEURO-FUZZY (ANFIS)

Tutorial ini cuma akan menerangkan konsep dasar pengontrolan dengan ANFIS,mudah2an bisa dimengerti sehingga tergambar jelas algoritmanya secara step-by-step.
Referensi:

[1] Anton Adi S,Sstudi dan Penerapan Model Neuro-Fuzzy Dalam Prakiraan Cuaca,S1 Jurusan Teknik Fisika ITB, 2000
[2] J.S.R. Jang, C.T. Sun, and E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice-Hall International, 1997

Apa itu ANFIS ?

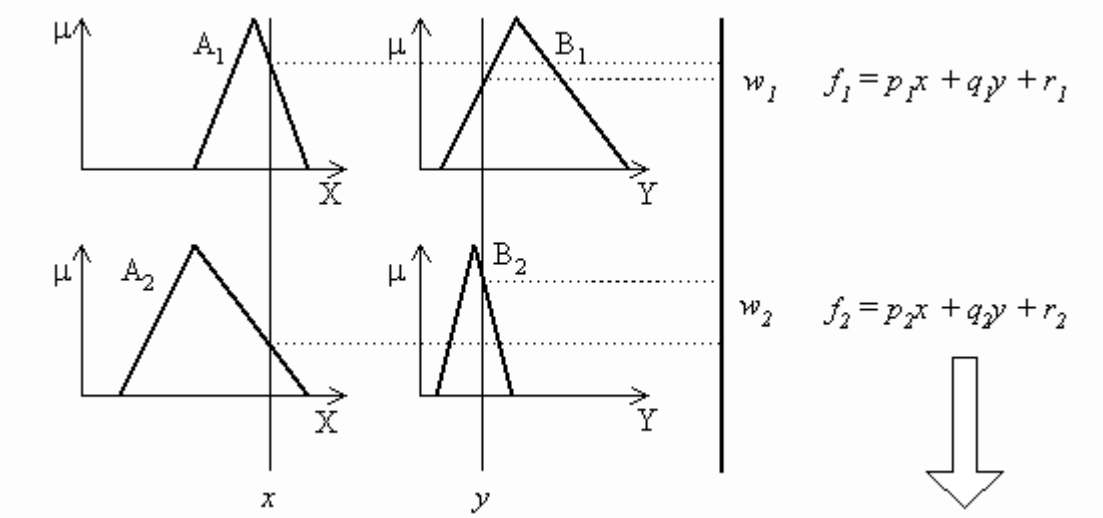
Adaptive Neuro-Fuzzy Inference System (ANFIS) adalah penggabungan mekanisme fuzzy inference system yang digambarkan dalam arsitektur jaringan syaraf. Sistem inferensi fuzzy yang digunakan adalah sistem inferensi fuzzy model Tagaki-Sugeno-Kang (TSK) orde satu dengan pertimbangan kesederhanaan dan kemudahan komputasi.

Nah ini salah satu contoh ilustrasi mekanisme inferensi fuzzy TSK orde satu dengan dua masukan x dan y (Gambar 1). Basis aturan dengan dua aturan fuzzy if-then seperti dibawah ini :

Rule 1 : if x is A1 and y is B1 then f1 = p1x + q1y + r1
premis consequent

Rule 2 : if x is A2 and y is B2 then f2 = p2x + q2y + r2
premis consequent

Input : x dan y.
Consequent-nya adalah f



$$f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \overline{w}_1 f_1 + \overline{w}_2 f_2$$

Gambar 1 Sistem inferensi fuzzy TSK dua masukan dengan dua aturan

Gimana cara kerjanya?

Cara kerjanya seperti sistem FIS biasa cuma cara perhitungannya (algoritmanya) yang beda.
Jelasnya gambar diatas memperlihatkan suatu masukan crisp (tidak fuzzy) x dan y, hmmm.. gini deh supaya jelas misalnya kita ingin mengontrol kecepatan motor listrik dengan mengatur tegangannya, jadi x itu pengukuran harga variabel yang dikontrol yaitu kecepatan pada saat ke t, dan misalnya y pengukuran pada saat ke t+1 sedangkan f adalah nilai tegangan yang diberikan sebagai sinyal kontrol. Harga x dan y tersebut jelaskan bukan fuzzy. Lalu nilai x dan y tersebut dipetakan pada funggsi keanggotaannya.
Dalam gambar diatas tiap-tiap input tersebut dibagi jadi 2 fungsi keanggotaan, x dibagi dalam A1 dan A2 anggap misalnya A1 menyatakan small dan A2 menyatakan big. Begitu juga y dibagi dalam fungsi keanggotaan B1 yang menyatakan small dan B2 yang menyatakan big.

Dari pemetaan tersebut x dan y sudah jadi variabel fuzzy yang masing-masing punya nilai m small dan big tertentu. x mempunyai nilai mA1 dan mA2 sedangkan y punya nilai mB1 dan mB2. Nilai masing-masing pasangan input tersebut lalu diagregasi dengan operasi T-norm, misalnya operasi ini adalah operasi AND. Jadi w1 = (mA1 AND mA2) sedangkan w2 = (mB1 AND mB2).

Dari basis aturan yang udah dibuat kita tau

```
if w=w1 then f1 = p1x + q1y + r1
if w=w2 then f2 = p2x + q2y + r2
```

Nah jadi sekarang kita punya hasil akhir f1 dan f2. Ini merupakan nilai output sinyal kontrol, yaitu tegangan. Perhatikan kita telah loncat dari domain input x dan y (kecepatan) ke domain output f (tegangan). Tapi itu nilai p1, q1, r1, p2, q2, dan r2 dari mana, siapa yang nentuin? Itu namanya parameter konsekuen yang ditentukan dengan nilai awal tertentu dan akan berubah dengan pembelajaran (algoritma belajar). Pada bagian pembelajaran parameter konsekuen hal ini akan dibicarakan lebih detail. Sekarang yang penting kita udah punya f1 dan f2. Selanjutnya dari nilai f1 dan f2 ini kita perlu mendapatkan satu nilai tegangan sebagai sinyal kontrol. Nah nilai akhir tersebut dihitung dengan persamaan:

$$f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \overline{w}_1 f_1 + \overline{w}_2 f_2$$

Ini namanya defuzzyfikasi. Rumus tersebut sebenarnya diperoleh dari salah satu metode defuzzyfikasi yaitu metode rata-rata tengah

(centroid). Selesai kaaan.. sekarang kita sudah punya harga tegangan output kontroler yang harus diberikan ke sistem yang kita kontrol. Cuma sekarang masalahnya adalah perhitungan dan pembelajaran parameter premis dan konsekuennya kaan...(?)

Sekarang kita kembali ke fungsi keanggotaan input.

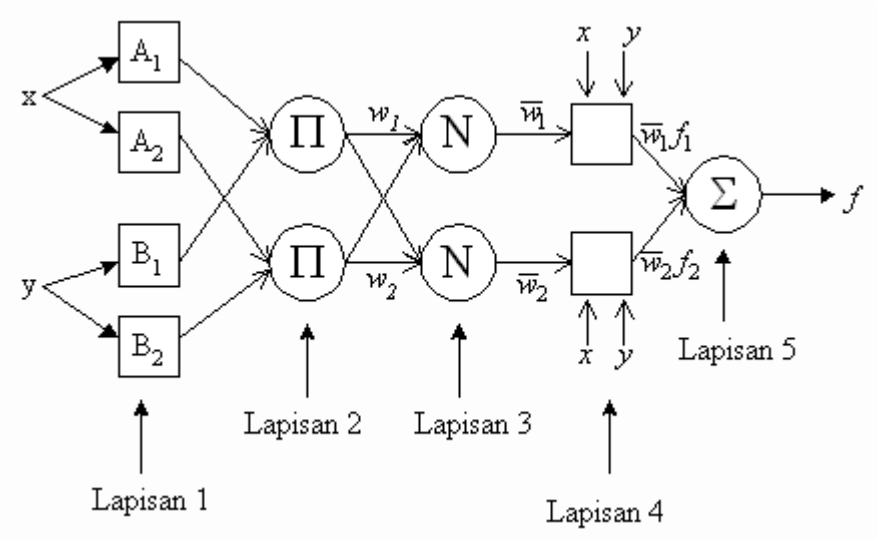
Biasanya fungsi keanggotaan fuzzy input (premis) yang digunakan adalah fungsi Generalized-Bell yang dirumuskan

$$gbell(x,a,b,c) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}$$

Nah, fungsi Generalized-Bell pertama-tama akan dipakai sebagai fungsi keanggotaan dari masukan, dan kita tentukan sekehendak kita parameter awal a, b, c dan jumlah himpunan fuzzy input. Nantinya parameter premis a,b,c akan diubah dengan cara pembelajaran.

Gimana struktur ANFIS koq sampai dibilang neural?

Struktur ANFIS yang menggambarkan sistem fuzzy TSK seperti yang digambarin di Gambar 1 diatas bisa digambarin dalam diagram blok atau disebut arsitektur jaringan syaraf feedforward seperti ini:



Gambar 2 Struktur ANFIS

Nah keliatan kan kalo seperti jaringan syaraf (neural-network).
Pada gambar 2 terlihat sistem neuro-fuzzy terdiri atas lima lapisan dengan fungsi yang berbeda untuk tiap lapisannya. Tiap lapisan terdiri atas beberapa simpul yang dilambangkan dengan kotak atau lingkaran. Lambang kotak menyatakan simpul adaptif artinya nilai parameternya bisa berubah dengan pembelajaran dan lambang lingkaran menyatakan simpul nonadaptif yang nilainya tetap.

Ooo.. jadi berlapis-lapis kaya kue lapis ya. Jelasin dong tiap lapisannya?

Bagian ini bisa dilewat, tanpa harus kehilangan kotinuitas (masih bakalan ngerti koq tanpa baca bagian yang ini). Baiklah tanpa proses editing sedikitpun lapisan2 tersebut dijelaskan Anton[1] sebagai berikut:

Lapisan 1. Semua simpul pada lapisan ini adalah simpul adaptif (parameter dapat berubah) dengan fungsi simpul :

$$O_{1,i} = \mu_{A_i}(x), \quad \text{untuk } i = 1, 2, \text{ atau}$$
$$O_{1,i} = \mu_{B_{i-2}}(y), \quad \text{untuk } i = 3, 4 \tag{1}$$

dengan x dan y adalah masukan pada simpul i, Ai (atau Bi-2) adalah fungsi keanggotaan masing-masing simpul. Simpul O1,i berfungsi untuk menyatakan derajat keanggotaan tiap masukan terhadap himpunan fuzzy A dan B. Fungsi keanggotaan yang dipakai adalah jenis generalized bell (gbell). Parameter a, b, c, pada fungsi keanggotaan gbell dinamakan parameter premis yang adaptif.

Lapisan 2. Semua simpul pada lapisan ini adalah nonadaptif (parameter tetap). Fungsi simpul ini adalah mengalikan setiap sinyal masukan yang datang. Fungsi simpul :

$$O_{2,i} = w_i = \mu_{A_i}(x).\mu_{B_i}(y), \quad i = 1, 2 \tag{2}$$

Tiap keluaran simpul menyatakan derajat pengaktifan (firing strength) tiap aturan fuzzy. Fungsi ini dapat diperluas apabila bagian premis memiliki lebih dari dua himpunan fuzzy. Banyaknya simpul pada lapisan ini menunjukkan banyaknya aturan yang dibentuk. Fungsi perkalian yang digunakan adalah interpretasi kata hubung and dengan menggunakan operator t-norm.

Lapisan Setiap simpul pada lapisan ini adalah simpul nonadaptif yang menampilkan fungsi derajat pengaktifan ternomalisasi (normalized firing strength) yaitu rasio keluaran simpul ke-i pada lapisan sebelumnya terhadap seluruh keluaran lapisan sebelumnya, dengan bentuk fungsi simpul:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{3}$$

Apabila dibentuk lebih dari dua aturan, fungsi dapat diperluas dengan membagi wi dengan jumlah total w untuk semua aturan.

Lapisan 4. Setiap simpul pada lapisan ini adalah simpul adaptif dengan fungsi simpul :

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i) \tag{3}$$

dengan adalah derajat perngaktifan ternormalisasi dari lapisan 3 dan parameter p, q, r menyatakan parameter konsekuen yang adaptif.

Lapisan 5. Pada lapisan ini hanya ada satu simpul tetap yang fungsinya untuk menjumlahkan semua masukan. Fungsi simpul :

$$O_{5,i} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{3}$$

Jaringan adaptif dengan lima lapisan tersebut ekivalen dengan sistem inferensi fuzzy TSK.

Gimana Proses Belajar pada ANFIS ?

Pada struktur ANFIS (gambar 2), simpul adaptif terdapat pada lapisan pertama dan keempat. Simpul pada lapisan pertama mengandung parameter premis yang nonlinier sedangkan pada lapisan keempat mengandung parameter konsekuen yang linier. Nah untuk memperbaharui parameter2 itu atau dalam kata lain si jaringan saraf itu belajar, maka perlu metoda atau algoritma untuk itu. Metoda pembelajaran jaringan syaraf tiruan banyak macamnya, dan masing2 punya kelebihan dan kekurangan.
Disini kita akan dijelaskan pembelajaran hibrid untuk ANFIS. Artinya penggunaan/penyatuan dua metoda pembelajaran pada ANFIS. Pembelajaran hibrid terdiri atas dua bagian yaitu arah maju (forward pass) dan arah mundur (backward pass).
Pada arah maju, parameter premis dibuat tetap. Dengan menggunakan metode Recursive Least Square Estimator (RLSE), parameter konsekuen diperbaiki berdasarkan pasangan data masukan-keluaran. Metode RLSE dapat diterapkan karena parameter konsekuen yang diperbaiki adalah parameter linier. Metode RLSE akan mempercepat proses belajar hibrid. Kemudian setelah parameter konsekuen didapatkan, data masukan dilewatkan jaringan adaptif kembali dan hasil keluaran jaringan adaptif ini dibandingkan dengan keluaran yang sebenarnya.

Pada arah mundur, parameter konsekuen dibuat tetap. Kesalahan yang terjadi antara keluaran jaringan adaptif dan keluaran sebenarnya dipropagasikan balik dengan menggunakan gradient descent untuk memperbaiki parameter premis. Pembelajaran ini dikenal sebagai Algoritma Backpropagation-error.
Satu tahap arah pembelajaran maju-mundur dinamakan satu epoch. Tabel 1 menerangkan proses pembelajaran hibrid ANFIS.

Tabel 1 Proses pembelajaran hibrid ANFIS

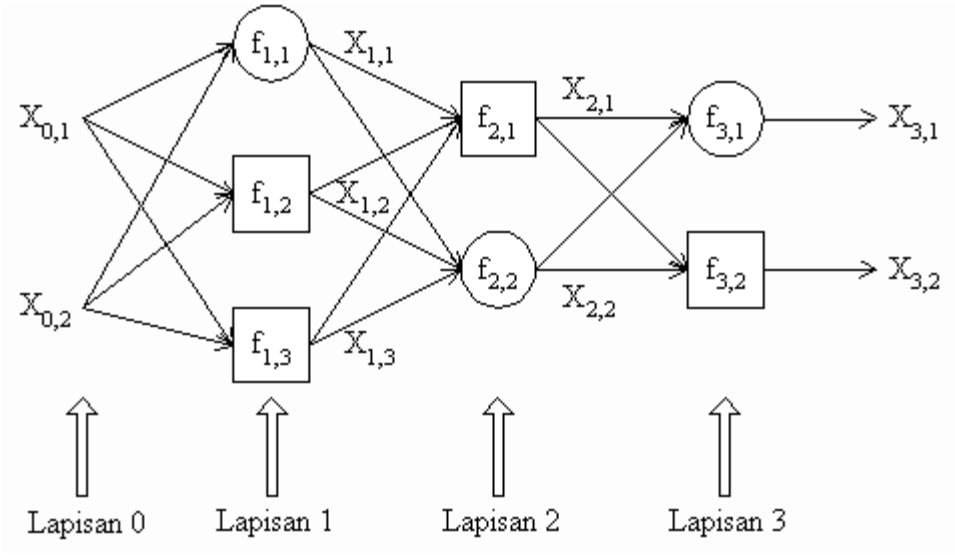
	Arah maju	Arah mundur
Parameter premis	Tetap	Gradient descent
Parameter konsekuen	RLSE	Tetap
Sinyal	Keluaran simpul	Laju kesalahan

Proses Pembelajaran Backpropagation-error untuk Parameter Premis

Parameter premis adalah parameter adaptif dengan proses pembelajarannya menggunakan metode belajar sistem jaringan syaraf feedforward dengan gradient descent. Misalkan sebuah sistem jaringan adaptif dinyatakan dengan L lapisan dan lapisan ke-l (l = 0, 1, ..., L; l = 0 menyatakan lapisan masukan)mempunyai N(l) simpul. Keluaran dan fungsi simpul ke-i [i = 1, ..., N(l)]pada lapisan ke-l dinyatakan dengan xl,i dan fl,i seperti tampak pada gambar Keluaran simpul merupakan fungsi dari sinyal yang masuk dan parameter sistem, maka diperoleh

$$x_{l,i} = f_{l,i}(x_{l-1,1}, \ldots x_{l-1,N(l-1)}, \alpha, \beta, \gamma, \ldots) \quad (6)$$

dengan a, b, g adalah parameter simpul.



Gambar 3 Model jaringan syaraf feedforward

Misalkan sejumlah P pasangan data untuk proses belajar jaringan adaptif, selanjutnya dapat didefinisikan pengukuran kesalahan pada data latih ke-p ( ) adalah jumlah kuadrat kesalahan.

$$E_p = \sum_{k=1}^{N(L)} (d_k^p - x_{L,k}^p)^2 \quad (7)$$

dengan d_k^p : komponen ke-k dari vektor keluaran yang diinginkan dan $x_{L,k}^p$ adalah vektor keluaran aktual yang dihasilkan sistem jaringan adaptif dengan masukan dari vektor masukan ke-p dari P data belajar. Tujuan dari sistem adaptif adalah untuk meminimumkan pengukuran kesalahan pada persamaan (7) dengan mengubah parameter-parameter adaptif.

Dengan mendefinisikan sinyal kesalahan $\varepsilon_{l,i}$ sebagai ordered derivative terhadap keluaran simpul ke-i, lapisan ke-l, maka ordered derivative dinotasikan dengan :

$$\varepsilon_{l,i} = \frac{\partial^+ E_p}{\partial x_{l,i}} \quad (8)$$

Sinyal kesalahan untuk simpul keluaran ke-i (pada lapisan L) dapat dihitung langsung dengan:

$$\varepsilon_{L,i} = \frac{\partial^+ E_p}{\partial x_{L,i}} = \frac{\partial E_p}{\partial x_{L,i}} \quad (9)$$

Jika pengukuran kesalahan seperti yang didefinisikan pada persamaan (7) maka persamaan (9) menjadi :

$$\varepsilon_{L,i} = -2(d_i^p - x_{L,i}^p) \quad (10)$$

Untuk simpul dalam pada lapisan l posisi ke-i, sinyal kesalahan dapat diperoleh menggunakan aturan rantai.

(11)

$$\varepsilon_{l,i} = \underbrace{\frac{\partial^+ E_p}{\partial x_{l,i}}}_{\substack{\text{sinyal kesalahan} \\ \text{pada lapisan } l}} = \sum_{m=1}^{M(l+1)} \underbrace{\frac{\partial^+ E_p}{\partial x_{l+1,m}}}_{\substack{\text{sinyal kesalahan} \\ \text{pada lapisan } l+1}} \frac{\partial f_{l+1,m}}{\partial x_{l,i}} = \sum_{m=1}^{N(l+1)} \varepsilon_{l+1,m} \frac{\partial f_{l+1,m}}{\partial x_{l,i}}$$

dengan $0 \le l \le L-1$. Sinyal kesalahan simpul dalam, pada lapisan ke-l dapat dinyatakan sebagai kombinasi linier dari sinyal kesalahan simpul pada lapisan ke (l+1). Jadi untuk menghitung sinyal kesalahan pada simpul ke-i lapisan ke-l (l < L), pertama digunakan persamaan (9) untuk mendapatkan sinyal kesalahan pada lapisan keluaran kemudian persamaan (11) secara iteratif sampai mencapai lapisan yang diinginkan. Prosedur diatas disebut penjalaran balik (backpropagation) karena sinyal kesalahan dihitung secara mundur dari lapisan keluaran hingga lapisan masukan.

Vektor gradien didefinisikan sebagai ordered derivative dari pengukuran kesalahan terhadap tiap parameternya. Jika a adalah parameter simpul ke-I lapisan ke-l, maka diperoleh :

$$\frac{\partial^+ E_p}{\partial \alpha} = \frac{\partial E_p}{\partial x_{l,i}} \frac{\partial f_{l,i}}{\partial \alpha} = \varepsilon_{l,i} \frac{\partial f_{l,i}}{\partial \alpha}$$

(12)

Jika a merupakan parameter yang ada pada beberapa simpul maka persamaan (12) menjadi :

$$\frac{\partial^+ E_p}{\partial \alpha} = \sum_{x^* \in S} \frac{\partial^+ E_p}{\partial x^*} \frac{\partial f^*}{\partial \alpha}$$

(13)

dengan S merupakan himpunan simpul yang berisikan a sebagai parameter, sedangkan x* dan f* adalah keluaran dan fungsi dari simpul yang bersangkutan.

Turunan masing-masing secara keseluruhan terhadap pengukuran kesalahan akan menghasilkan :

$$\frac{\partial^+ E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial^+ E_p}{\partial \alpha}$$

(14)

Dengan metode gradient simple steepest descent, persamaan untuk memperbaiki parameter a adalah :

$$\Delta \alpha = -\eta \frac{\partial^+ E}{\partial \alpha}$$

(15)

dengan h adalah laju proses belajar (learning rate) yang dinyatakan dengan:

$$\eta = \frac{\kappa}{\sqrt{\sum_{\alpha} \left(\frac{\partial E}{\partial \alpha} \right)^2}}$$

(16)

dan k adalah ukuran langkah (step size) yang dapat diubah untuk mempercepat konvergensi.

Proses Pembelajaran dengan RLSE untuk Parameter Konsekuen

Kita balik lagi ke gambar 2, jika nilai dari parameter premis tetap maka keluaran keseluruhannya dapat dinyatakan dengan kombinasi linier dari parameter konsekuen [7]

$$\begin{aligned} f &= \overline{w}_1 f_1 + \overline{w}_2 f_2 \\ &= \overline{w}_1 (p_1 x + q_1 y + r_1) + \overline{w}_2 (p_2 x + q_2 y + r_2) \\ &= (\overline{w}_1 x) p_1 + (\overline{w}_1 y) q_1 + (\overline{w}_1) r_1 + (\overline{w}_2 x) p_2 + (\overline{w}_2 y) q_2 + (\overline{w}_2) r_2 \end{aligned} \tag{17}$$

Pada persamaan (17) terlihat parameter-parameter bagian konsekuen merupakan parameter linier terhadap keluaran sistem. Jika sejumlah N data belajar diterapkan pada persamaan (17), didapat :

$$\begin{gathered} (\overline{w}_1 x)_1 p_1 + (\overline{w}_1 y)_1 q_1 + (\overline{w}_1)_1 r_1 + (\overline{w}_2 x)_1 p_2 + (\overline{w}_2 y)_1 q_2 + (\overline{w}_2)_1 r_2 = y_1 \\ \vdots \\ \vdots \\ (\overline{w}_1 x)_n p_1 + (\overline{w}_1 y)_n q_1 + (\overline{w}_1)_n r_1 + (\overline{w}_2 x)_n p_2 + (\overline{w}_2 y)_n q_2 + (\overline{w}_2)_n r_2 = y_n \end{gathered} \tag{18}$$

jika dinyatakan dengan persamaan matriks, berbentuk :

$$\mathbf{A}\theta = \mathbf{y} \tag{19}$$

dengan q merupakan vektor 6x1 yang elemen-elemennya merupakan himpunan parameter konsekuen, y merupakan vektor keluaran yang elemen-elemennya N buah data keluaran sistem. Penyelesaian terbaik untuk q adalah meminimumkan $\|\mathbf{A}\theta = \mathbf{y}\|^2$, dengan teori LSE (Least Square Estimator) didapat q* :

$$\theta^* = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y} \tag{20}$$

q* dihitung dengan rumus Recsursive LSE (RLSE). Selanjutnya persamaan (20) menjadi :

$$\begin{cases} \theta_{i+1} = \theta_i + \mathbf{P}_{i+1} \mathbf{a}_{i+1} (y_{i+1}^T - \mathbf{a}_{i+1}^T \theta_i) \\ \mathbf{P}_{i+1} = \mathbf{P}_i - \dfrac{\mathbf{P}_i \mathbf{a}_{i+1} \mathbf{a}_{i+1}^T \mathbf{P}_i}{1 + \mathbf{a}_{i+1}^T \mathbf{P}_i \mathbf{a}_{i+1}}, \quad i = 0,1,\ldots,P-1 \end{cases} \tag{21}$$

dengan $\mathbf{a}_i^T$ adalah vektor baris dari matriks A pada persamaan (19), yi adalah elemen ke-i dari y. Pi disebut matriks kovariansi yang didefinisikan dengan :

$$\mathbf{P}_i = (\mathbf{A}^T \mathbf{A})^{-1} \tag{22}$$