เอกสารอ้างอิง

1. Doebelin, E. O., System Modeling and Response, John Wiley & Son, N.Y., 1980.
2. Chen, C. T., One-dimensional Digital Signal Processing, Marcel Dekker, N.Y., 1979.
3. Desilva, C. W. and S. S. Palusamy, "Experimental Model Analysis - A Modeling and Design Tool", Mechanical Engineering, June, 56-65, 1984.
4. Cowley P., "The Accuracy of Frequency Response Function Measurements Using FFT-Based Analyzers With Transient Excitation", ASME Trans. on Vibration, Acoustics, Stress, and Realiabity in Design, 108, 44-49, 1986.
5. Eykhoff, P., System Identification, John Wiley & Sons, N.Y., 1974.
6. Hsia, T. C., System Identification, Lexington Books, Massachusetts, 1977.
7. Fritzen, C. T., "Identification of Mass, Damping, and Stiffness Matrices of Mechanical Systems", ASME Trans. on Vibration, Acoustics, Stress, and Reliability in Design, 108, 9-16, 1986.
8. Lin, P. L. and Y. C. Wu, "Identification of Multi-Input Multi-Output Linear Systems From Frequency Response Data", ASME Trans. on Dynamic Systems, Measurement, and Control, 104, 58-64, 1982.
9. Okata K., Modern Control Engineering, Prentice-Hall., Englewood Cliffs, N.J., 1970.

ภาคผนวก

ภาคผนวก ก

การเปรียบเทียบวิธีการประมาณค่าพารามิเตอร์ ระหว่างวิธีตัวแปรอินสทรูเมนทอลที่ใช้ในการวิจัยกับวิธีกำลังสองน้อยที่สุด

การเปรียบเทียบวิธีการประมาณค่าพารามิเตอร์ระหว่างวิธีตัวแปรอินสทรูเนทอล (Instrumental Variable Method) กับวิธีกำลังสองน้อยที่สุด (Least Squares Method) จะแสดงให้เห็นโดยการทดลองประมาณค่าพารามิเตอร์ของระบบจำลองเชิงกลแบบเชิงเส้น (Simulated Linear Mechanical System) เพื่อตรวจดูผลของการรบกวน (Noise) ที่มีต่อวิธีการ ระบบจำลองที่ใช้ เป็นระบบเชิงกล 2 องศาอิสระ แสดงในรูป ก-1 ประกอบด้วยมวล, สปริง และแดมเปอร์ตามลำดับ โดยกฎของนิวตันและวิธีการในบทที่ 2 จะได้ทรานเฟอร์ฟังก์ชัน

$$H_3 = (4\times10^{-3}\, s + 0.2) / (s^4 + 4.4\, s^3 + 2.124\times10^3\, s^2 + 4.2\times10^3\, s + 2\times10^5)$$

$$= X_2(s) / F_1(s) \qquad (ก-1)$$

เมื่อ H_3 มีอินพุทเป็นแรง f_1 (ให้ $f_2 = 0$) และเอ้าพุทเป็นการขจัด x_2 หรือได้ค่าพารามิเตอร์

$b_o = 0.2$

$b_1 = 4\times10^{-3}$

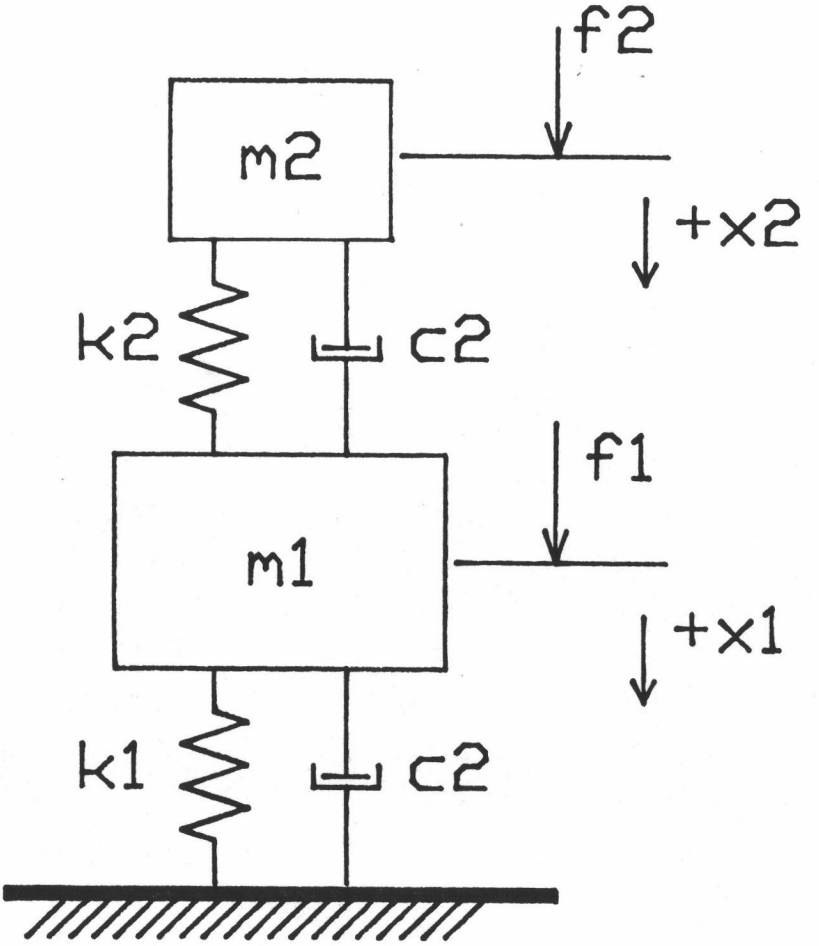
$a_o = 2\times10^5$

$a_1 = 4.2\times10^3$

$a_2 = 2.124\times10^3$

$a_3 = 4.4$

จากสมการ ก-1 นำมาคำนวณหาผลตอบความถี่ โดยให้ $s = jw$ และใช้วิธีตัวแปรอินสทรูเมนทอล ประมาณค่าพารามิเตอร์ โดยสมมติว่าได้ทราบโครงสร้างของทรานสเฟอร์ฟังก์ชันหรือออเดอร์ (Order) ของระบบแล้ว ผลที่ได้



$m1 = 500$ kg, $m2 = 100$ kg

$k1 = 1 \times 10^6$ N/m, $k2 = 1 \times 10^4$ N/m

$c1 = 1000$ Ns/m, $c2 = 200$ Ns/m

รูป ก-1 ระบบจำลองที่ใช้ทดสอบวิธีตัวแปรอินสทรูเมนทอล

ITER	PARAMETERS, N/S = 0					
	b0	b1	a0	a1	a2	a3
(LS)0	0.199999	0.004000	199999.8	4200.002	2124.000	4.400002
1	0.200000	0.003999	200000.6	4200.008	2124.000	4.400004
2	0.199999	0.004000	199998.8	4199.992	2123.999	4.399996
3	0.199999	0.004000	199999.9	4199.996	2123.999	4.399997
4	0.199999	0.004000	199998.9	4199.997	2123.999	4.399999
5	0.200000	0.003999	200000.4	4200.008	2124.000	4.400004
6	0.199999	0.004000	199999.3	4199.996	2123.999	4.399998
7	0.199999	0.004000	199999.5	4199.991	2123.999	4.399995
8	0.199999	0.004000	199999.7	4200.002	2124.000	4.400001
9	0.200000	0.004000	199999.4	4200.004	2124.000	4.400002
10	0.199999	0.004000	199999.5	4199.999	2123.999	4.400000

ตาราง ก-1 ค่าพารามิเตอร์ของทรานสเฟอร์ฟังก์ชัน H_3 ที่ประมาณโดยวิธีตัวแปรอินสทรูเมนทอล เมื่อ N/S = 0 (ค่าที่ลำดับขั้นที่ศูนย์ เป็นค่าที่ได้จากวิธีกำลังสองน้อยที่สุด)

แสดงในตาราง ก-1 จะเห็นว่าค่าที่ได้จากวิธีกำลังสองน้อยที่สุดและจากลำดับขั้นต่างๆของวิธีตัวแปรอินสทรูเมนทอลมิได้แตกต่างกัน เนื่องจากข้อมูลผลตอบความถี่นี้ไม่มีการรบกวน ทำให้เมตริก W ที่สร้างขึ้นมาจากโมเดลช่วยยังคงเป็นเมตริก A เดิมอยู่ (System Matrix) และจะสังเกตุเห็นว่าค่าตัวเลขที่ต่างไปบ้างเล็กน้อย ซึ่งมิได้แสดงแนวโน้มของค่าพารามิเตอร์ไปทางใดทางหนึ่งนั้น เป็นผลจากการตัดทอนตัวเลข (Truncation Error) ในการการคำนวณแบบดิจิตอลของเครื่องคอมพิวเตอร์ และจากการใช้ข้อมูลแบบไม่ต่อเนื่อง (Discrete Data)

ต่อไปใส่ค่ารบกวน (Noise) เข้าไปในส่วนจริงและส่วนจินตภาพ (Real and Imaginary Part) ของข้อมูลผลตอบความถี่ที่สร้างขึ้น โดยค่ารบกวนที่ใช้ มีการแจกแจงแบบเก๊า (Gaussian Distribution) และมีค่า

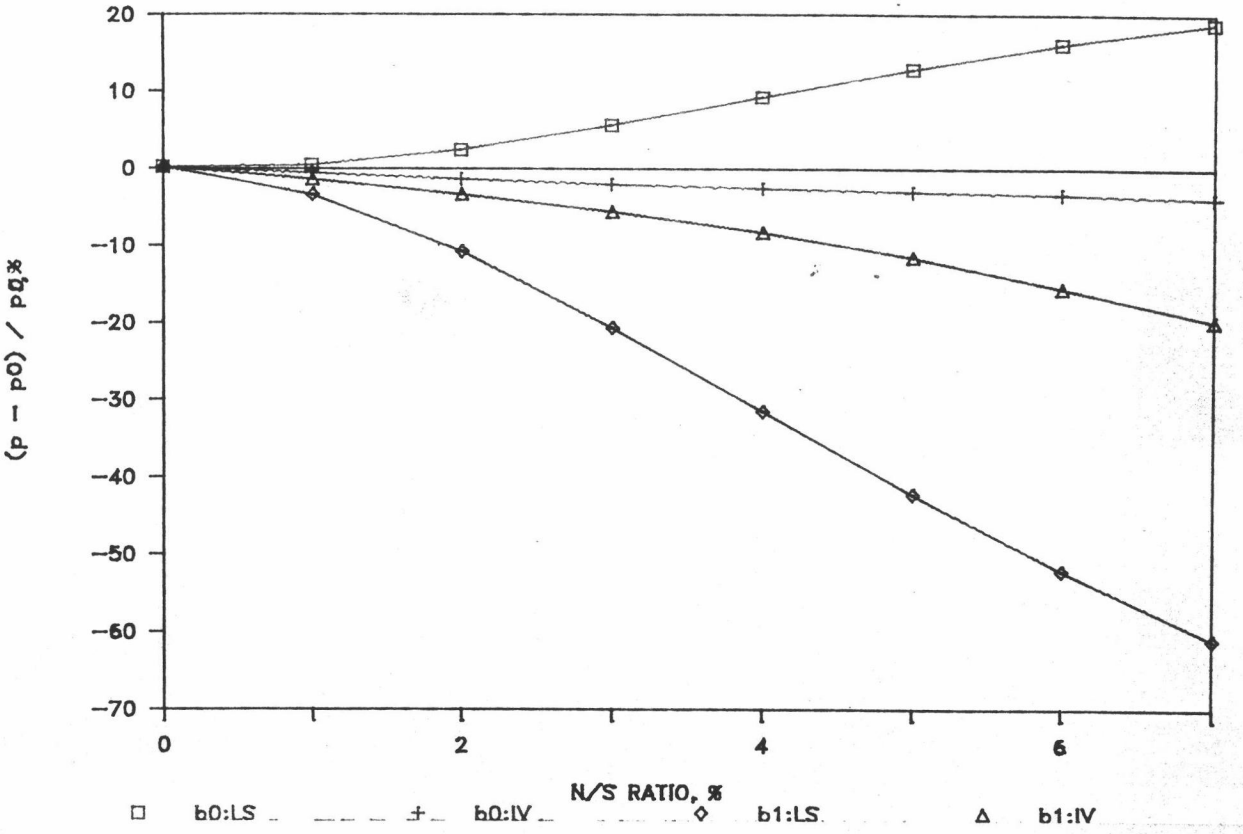
เฉลี่ย (mean) เป็นศูนย์ ขนาดของการรบกวนสามารถคำนวณออกมาเป็นอัตราส่วน N/S (Noise per Signal Ratio) โดย

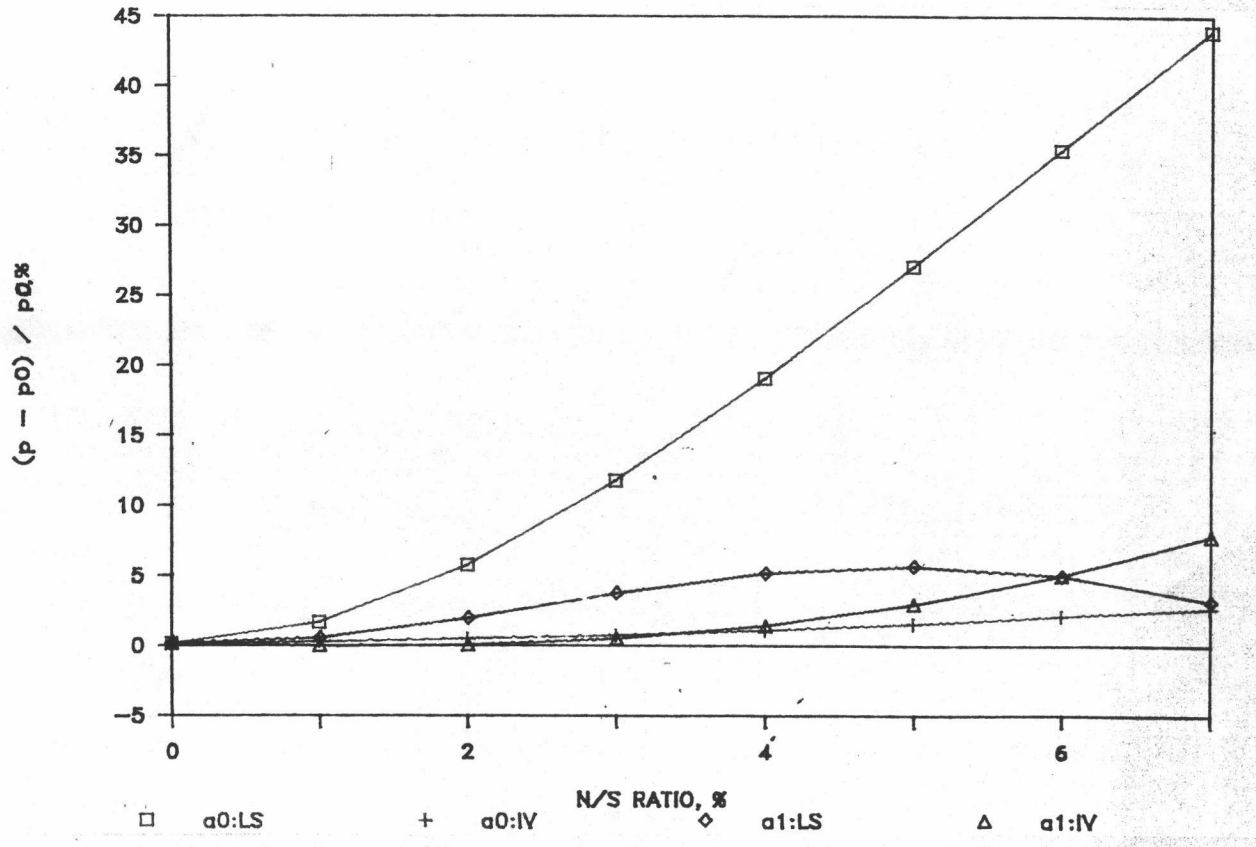
$$N/S = \left(\sum_{i=1}^{t} |n_i|^2 \Big/ \sum_{i=1}^{t} \{Re^2[h(jw_i)] + Im^2[h(jw_i)]\} \right)^{1/2} \qquad (ก-2)$$

เมื่อ t เป็นจำนวนข้อมูล Re[.] และ Im[.] แสดงส่วนจริงและส่วนจินตภาพตามลำดับ ค่ารบกวนที่ใช้นั้นสร้างจากฟังก์ชันแรนดอมเจนเนอเรเตอร์ (Random Generator) ในตัวแปรภาษาที่ใช้

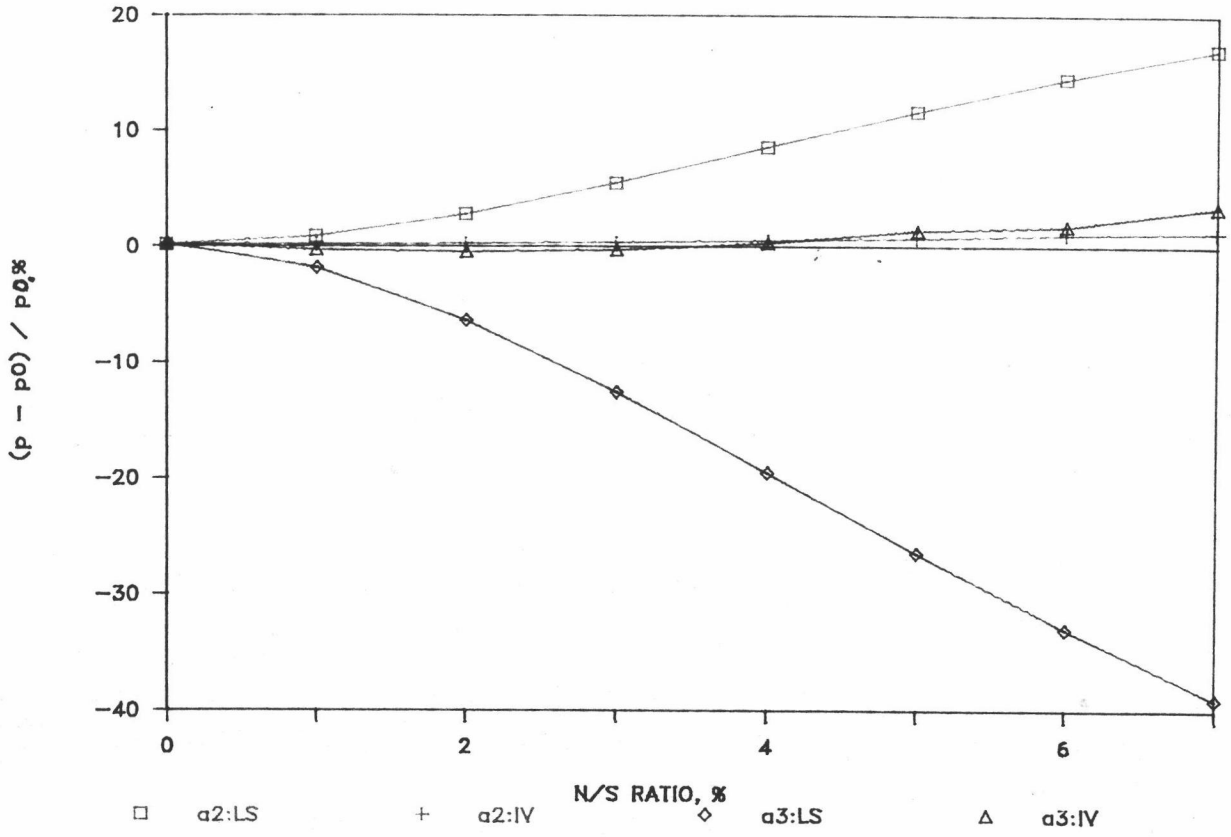
คำตอบของการประมาณค่าพารามิเตอร์ของระบบที่มีการรบกวนนี้ได้มาจากค่าเฉลี่ยของการประมาณ 10 หน แต่ละหนใช้ค่ารบกวนต่างชุดกันไปโดยการเปลี่ยนค่าซีดนัมเบอร์ (Seed Number) ของแรนดอมเจนเนอเรเตอร์ ทั้งนี้เนื่องจากการรบกวนนั้นเป็นกระบวนการความน่าจะเป็น ค่ารบกวนแต่ละชุดจึงเป็นโอกาสหนึ่งที่จะเกิดขึ้นเท่านั้นซึ่งมีลักษณะที่แตกต่างกันไป ดังนั้นการใช้ค่าเฉลี่ยจากการประมาณหลายๆหน จึงทำให้มั่นใจได้ว่าได้ค่าที่ดีที่สุด การประมาณค่า ใช้ข้อมูลผลตอบความถี่ช่วงตั้งแต่ 0 ถึง 10 Hz จำนวน 60 ค่า ซึ่งครอบคลุมความถี่ธรรมชาติทั้งสองโหมดของระบบ การที่ใช้ช่วงความถี่เท่านี้จะให้ผลดีกว่าการใช้ช่วงความถี่ที่มากเกินไป เนื่องจากที่ช่วงความถี่ที่เกินจากนี้ไปค่าของทรานส์เฟอร์ฟังก์ชันจะมีค่าต่ำ ทำให้เปอร์เซนต์ N/S ในช่วงนั้นเพิ่มสูงขึ้นมาก น้ำหนัก (Weight) ของการประมาณจะตกไปยังช่วงความถี่ส่วนนั้นมาก ซึ่งไม่ใช่ส่วนสำคัญที่สนใจ ทำให้ค่าพารามิเตอร์ที่ได้ผิดพลาดมากขึ้น หรือการประมาณอาจจะไม่คอนเวอร์จได้

รูป ก-2 แสดงผลของขนาดของค่ารบกวน (N/S ratio) ที่มีต่อไบแอสของพารามิเตอร์แต่ละตัว จะเห็นว่าเมื่อ N/S เพิ่มขึ้น ขนาดของไบแอสก็จะเพิ่มขึ้นด้วย เมื่อเปรียบเทียบวิธีตัวแปรอินสทรูเมนทอล กับวิธีกำลังสองน้อยที่สุดจะพบว่าวิธีตัวแปรอินสทรูเมนทอลทำให้อัตราการเพิ่มของค่าไบแอสน้อยกว่าวิธีกำลังสองน้อยที่สุดมาก และแม้ว่าค่าไบแอสจะไม่หมดไปทีเดียว แต่ก็อยู่ในเกณฑ์ที่ยอมรับได้ ค่าไบแอสที่ยังหลงเหลืออยู่นี้เป็นผลเนื่องมาจากเมตริก W ที่





รูป ก-2 แสดงค่าไบแอสของพารามิเตอร์ ที่ N/S ค่าต่างๆ



รูป ก-2 (ต่อ)

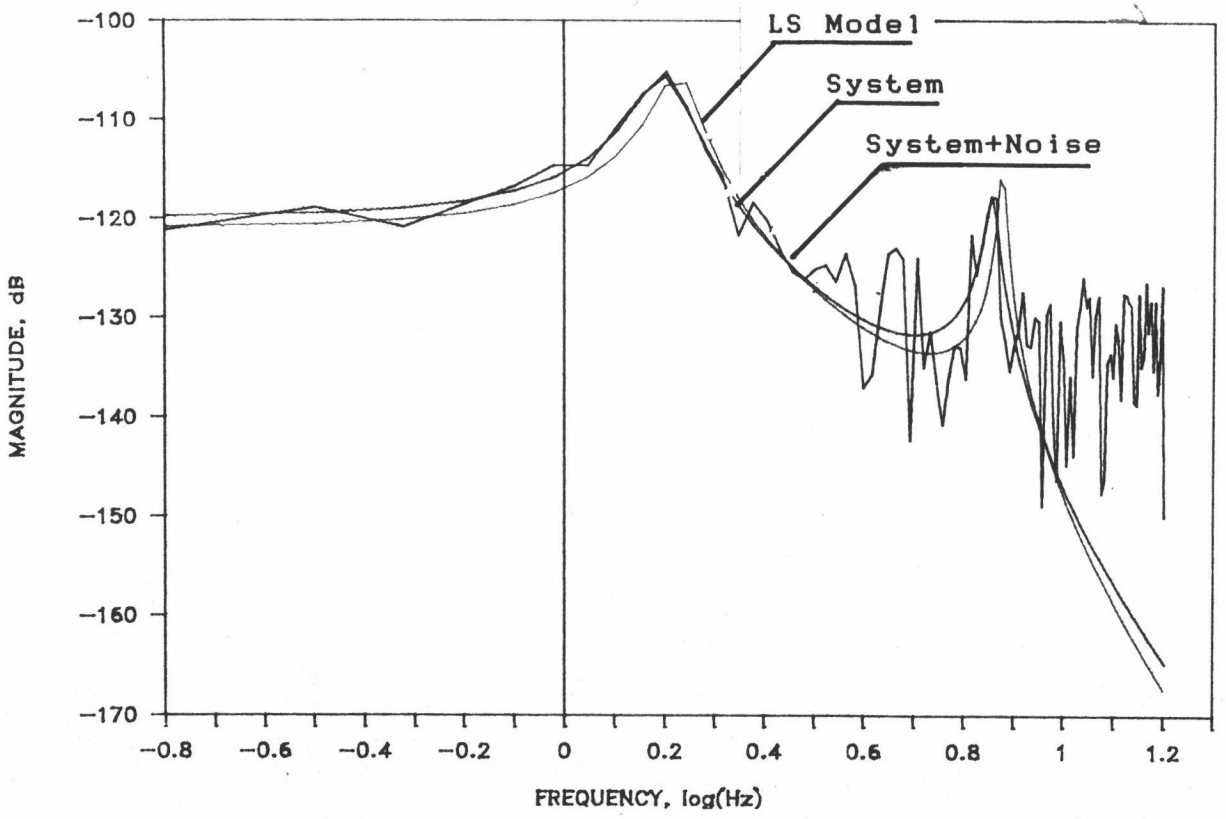
ได้สร้างขึ้นตามวิธีตัวแปรอินสทรูเมนทอลนั้นเป็นเมตริกที่มีคุณสมบัติโดยประมาณตามสมการ (4-29) และ (4-30) เท่านั้น ถึงแม้ว่าจะมีการคำนวณหาเมตริก W ซ้ำเป็นลำดับขั้น (iteration) ซึ่งทำให้เมตริก W มีคุณสมบัติใกล้เคียงกับสมการทั้งสองมากขึ้นแล้วก็ตาม ก็จะยังมีไบแอสเหลืออยู่ดังเหตุผลข้างต้น และจะสังเกตุเห็นว่าถ้า N/S มากขึ้น ค่าไบแอสก็จะมากขึ้นด้วย จากเหตุผลที่ว่าเมื่อ N/S มากขึ้น การสร้างเมตริก W ด้วยวิธีดังกล่าวจะทำให้คุณสมบัติของมันผิดไปจากสมการ (4-29) และ (4-30) มากขึ้นนั่นเอง พิจารณารูป ก-2 ที่ค่า N/S มากกว่า 6 เปอร์เซนต์ ค่าไบแอสของพารามิเตอร์ a_1 เมื่อใช้วิธีตัวแปรอินสทรูเมนทอลกลับมีค่ามากกว่าวิธีกำลังสองน้อยที่สุด ซึ่งน่าจะหมายความว่าวิธีการจะใช้ไม่ได้ผลเมื่อค่า N/S เพิ่มขึ้นถึงระดับหนึ่ง ในการทดสอบพบว่าค่าพารามิเตอร์เริ่มไม่คอนเวอร์จเมื่อ N/S มีค่าประมาณ 8 เปอร์เซนต์ แสดงให้เห็นว่าวิธีการจะใช้ได้ผลเมื่อ N/S อยู่ในระดับไม่มากเกินไปซึ่งระดับที่ว่านี้ขึ้นอยู่กับคุณสมบัติของระบบนั้นเองด้วย

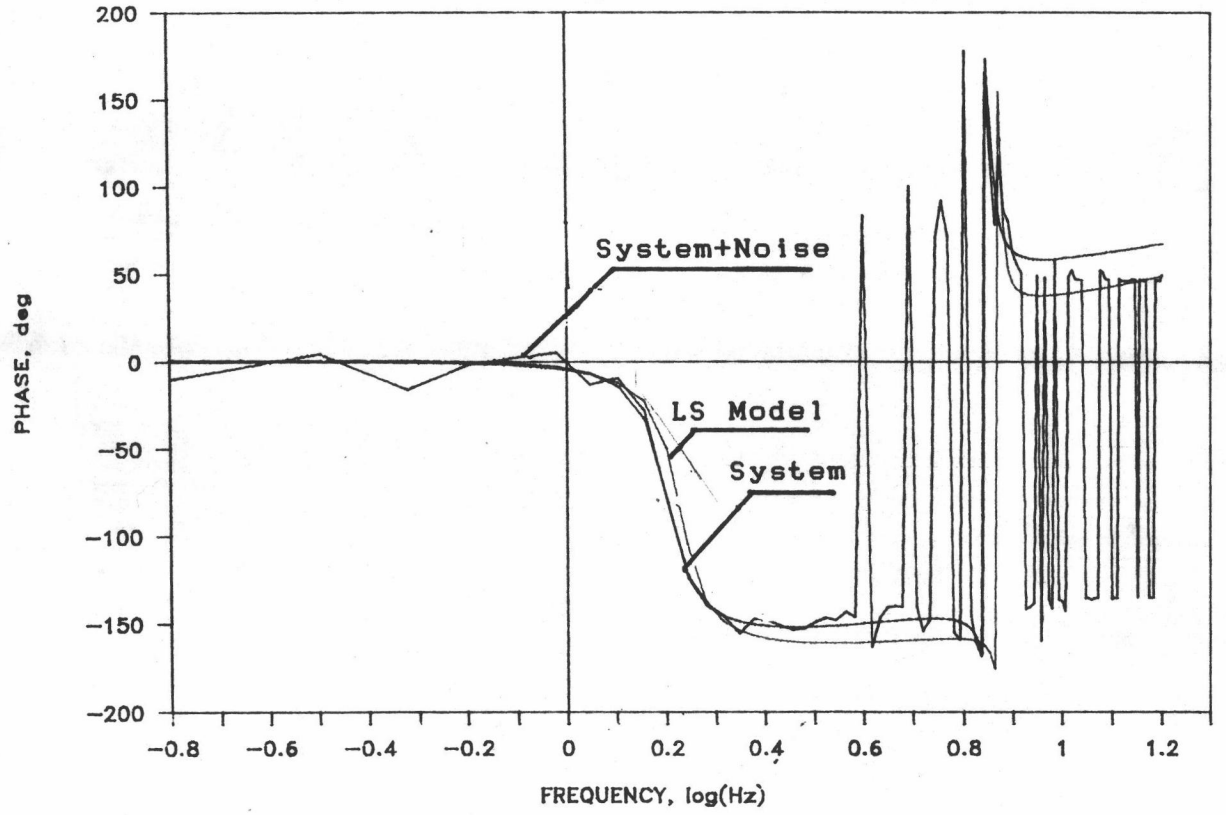
PARA-METER	p_o	$\hat{p}_{LS}$	$\left\|\frac{p_{LS}-p_o}{p_o}\right\|$ %	$\hat{p}_{IV}$ Iter = 5	$\left\|\frac{p_{IV}-p_o}{p_o}\right\|$ %
b_0	2.000×10^{-1}	2.253×10^{-1}	12.67	1.935×10^{-1}	3.25
b_1	4.000×10^{-3}	2.300×10^{-3}	42.5	3.530×10^{-3}	11.75
a_0	2.000×10^{5}	2.539×10^{5}	26.93	2.028×10^{5}	1.4
a_1	4.200×10^{3}	4.431×10^{3}	5.5	4.314×10^{3}	2.7
a_2	4.400	3.228	26.64	4.448	1.09

ตาราง ก-2 ค่าพารามิเตอร์ของทรานสเฟอร์ฟังก์ชัน H_3 ที่ประมาณโดยวิธีตัวแปรอินสทรูเมนทอล เมื่อ N/S = 0.05 (ค่าที่ลำดับขั้นที่ศูนย์ เป็นค่าที่ได้จากวิธีกำลังสองน้อยที่สุด)

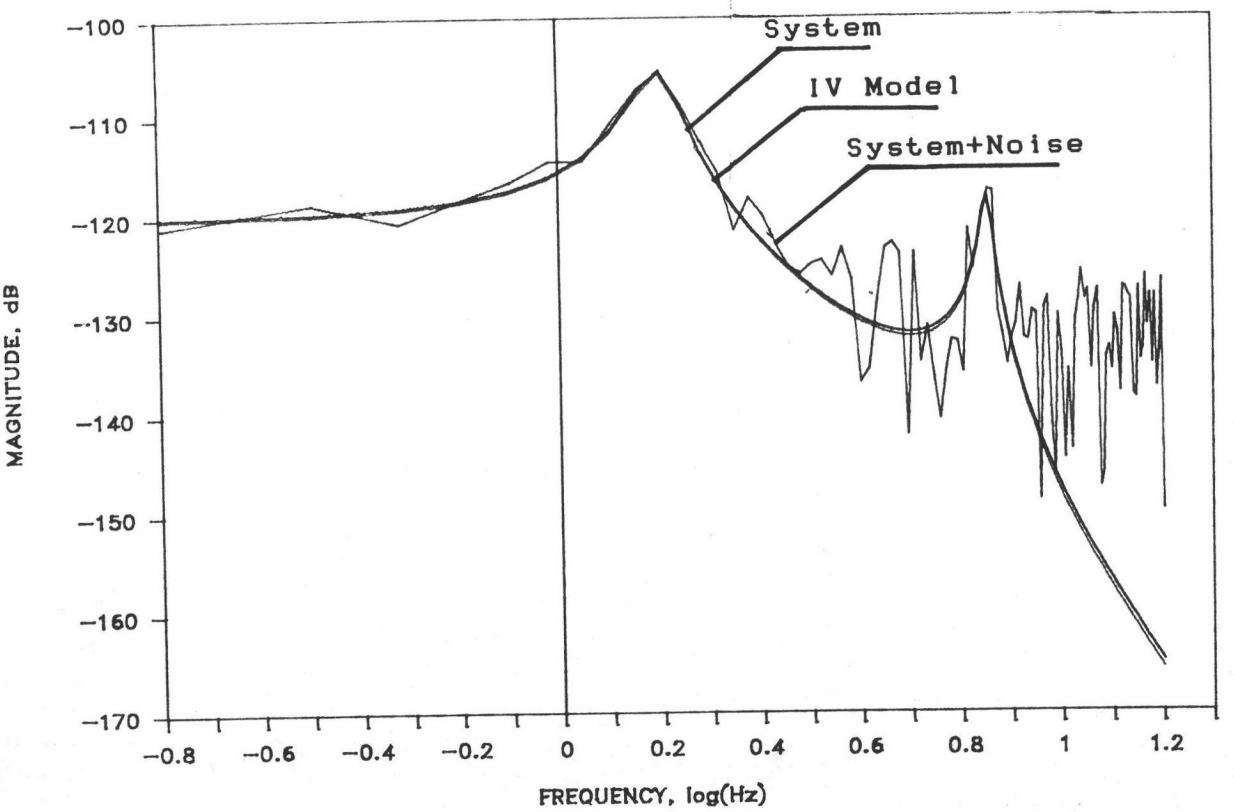
ตาราง ก-2 แสดงค่าพารามิเตอร์ที่ประมาณได้โดยวิธีตัวแปรอินสทรูเมนทอล เมื่อกำหนดให้ N/S เท่ากับ 5 เปอร์เซนต์ เปรียบเทียบกับวิธีกำลังสองน้อยที่สุด เห็นได้ชัดว่าวิธีตัวแปรอินสทรูเมนทอลให้คำตอบที่ใกล้เคียงกับค่าพารามิเตอร์จริงมากกว่ามาก อย่างไรก็ตามจะสังเกตุเห็นว่าค่าพารามิเตอร์ b_1 ของทั้งสองวิธีให้ค่าประมาณที่ผิดไปจากค่าจริงมากซึ่งเป็นผลเนื่องมาจากพารามิเตอร์ b_1 มีค่าเล็กมากเมื่อเทียบกับพารามิเตอร์ตัวอื่นๆ

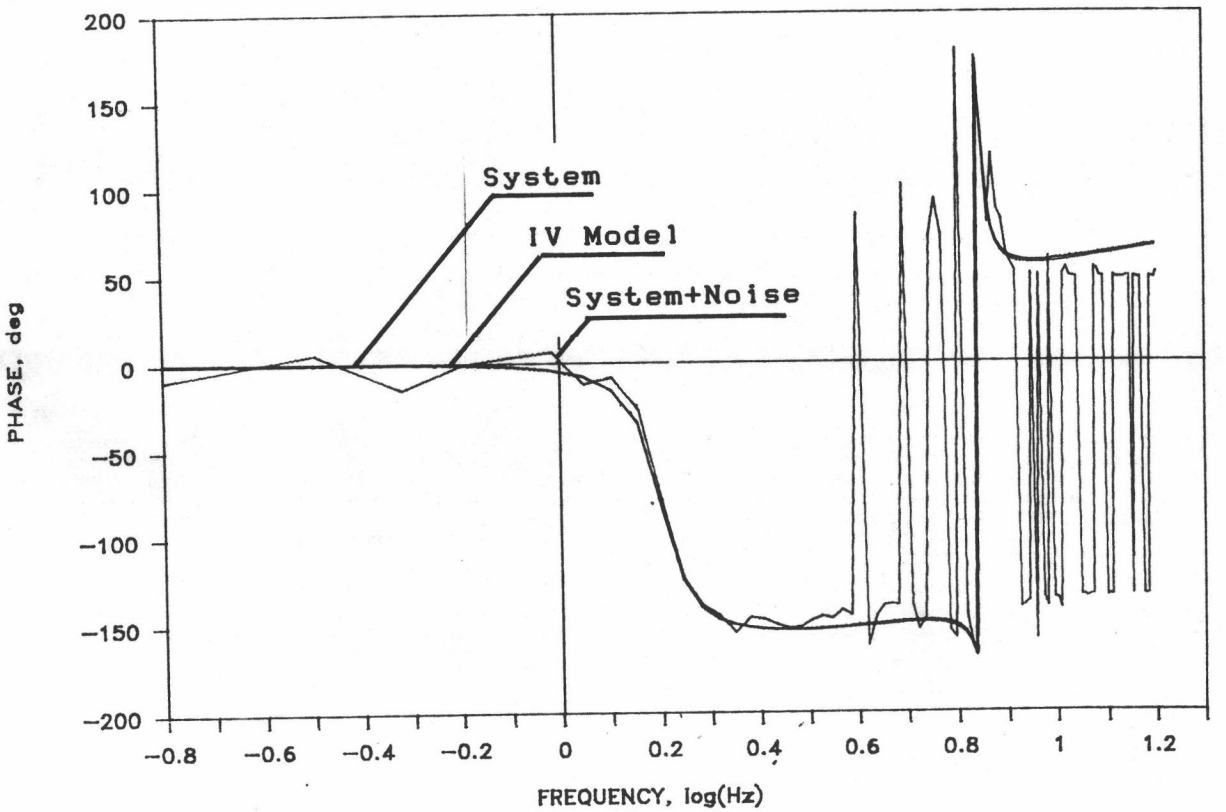
รูป ก-3 และ ก-4 แสดงทรานสเฟอร์ฟังก์ชันของระบบจำลองที่สร้างขึ้นประกอบด้วยทรานสเฟอร์ฟังก์ชันจริง ทรานส์เฟอร์ฟังก์ชันที่มีการรบกวน และโมเดลที่ได้จากการประมาณค่าพารามิเตอร์ของทั้งสองวิธี โดยกำหนดให้ N/S เท่ากับ 5 เปอร์เซนต์ ซึ่งย้ำให้เห็นว่าวิธีตัวแปรอินสทรูเมนทอลสามารถให้



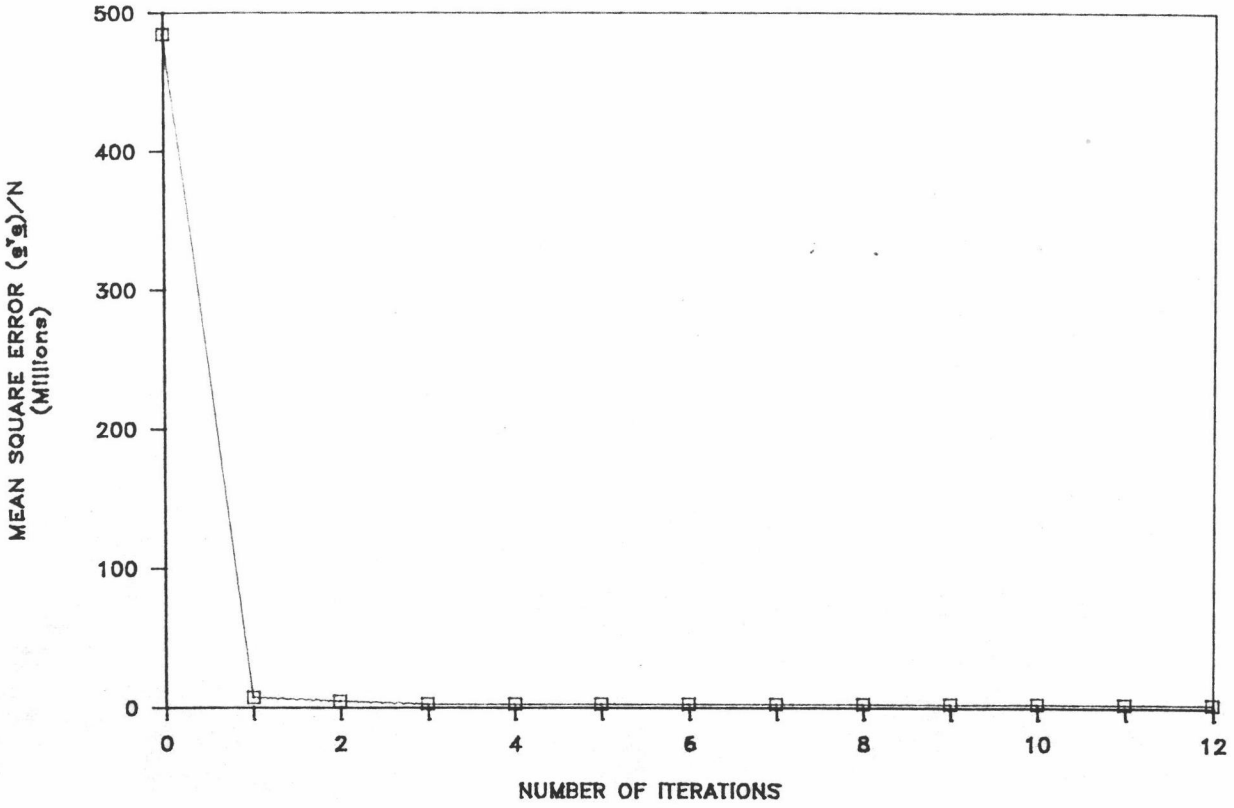


รูป ก-3 แสดงโมเดลที่ได้จากวิธีกำลังสองน้อยที่สุด เปรียบเทียบกับทรานสเฟอร์ฟังก์ชัน H_3 จริง





รูป ก-4 แสดงโมเดลที่ได้จากวิธีตัวแปรอินสทรูเมนทอล เปรียบเทียบกับทรานสเฟอร์ฟังก์ชัน H_3 จริง



รูป ก-5 แสดงความเร็วในการคอนเวอร์จของวิธีตัวแปรอินสทรูเมนทอล

โมเดลที่ใกล้เคียงกับระบบจริงมากกว่าวิธีกำลังสองน้อยที่สุดเมื่อระบบมีการรบกวน และจากรูป ก-5 แสดงให้เห็นความเร็วในการคอนเวอร์จของวิธีนี้จากลำดับดับขั้นที่ศูนย์ (Iteration 0) ซึ่งเป็นค่าเฉลี่ยความผิดพลาดกำลังสอง (Mean Square Error, $\underline{e}^T\underline{e}$) ของวิธีกำลังสองน้อยที่สุด ลงไปเข้าใกล้ค่าศูนย์อย่างรวดเร็วภายในไม่กี่ลำดับขั้น

ภาคผนวก ข

ตัวโปรแกรมคอมพิวเตอร์ที่ใช้ในการวิจัย

โปรแกรมคอมพิวเตอร์ที่ใช้ในการวิจัย เป็นโปรแกรมที่เขียนขึ้นด้วยภาษาซี (C Language) ใช้งานบนเครื่องไมโครคอมพิวเตอร์ขนาด 16 บิท โดยใช้ตัวแปรภาษา Optimizing C-86 ของบริษัท Computer Innovation ประเทศสหรัฐอเมริกา ประกอบด้วยโปรแกรมวิเคราะห์เสปคตรัมและโปรแกรมประมาณค่าพารามิเตอร์

ข-1 โปรแกรมวิเคราะห์เสปคตรัม ประกอบด้วยตัวโปรแกรมย่อยที่สำคัญดังนี้ (ดูบทที่ 3 หัวข้อ 3.4 ประกอบ)

```
window(w_type, ax, n_dat)
int w_type, n_dat;
float ax[];
{
    switch (w_type) {
        case 0:
            break;
        case 1:
            w_hanning(ax, n_dat);
            break;
        case 2:
            w_flattop(ax, n_dat);
            break;
        case 3:
            w_hamming(ax, n_dat);
            break;
    }
}
```

โปรแกรม ข-1 โปรแกรมย่อยสำหรับสร้างวินโดว์ให้กับข้อมูล (Windowing)

```
w_hanning(ax, n)
/*
    hanning window
*/
int n;
float ax[];
{
    extern double cos();
    int i;
    float fi, fn, w;

    fn = (float) n;
    for ( i = 1; i <= n; i++) {
        fi = (float) i;
        w = 0.5 * (1.0 - cos(2.0 * PI * (fi - 1.0) / (fn - 1.0)));
        ax[i] *= w;
    }
}

w_hamming(ax, n)
/*
    hamming window
*/
int n;
float ax[];
{
    extern double cos();
    int i;
    float fi, fn, w;

    fn = (float) n;
    for ( i = 1; i <= n; i++) {
        fi = (float) i;
        w = 0.8 + (0.46 * cos(2.0 * PI * (fi - 1.0) / (fn - 1.0)));
        ax[i] *= w;
    }
}
```

โปรแกรม ข-1 (ต่อ)

```
Mean_adj(a, n_dat)
/*
    Mean Adjustment  of Time Domain Response of Random Signal
    Arguments:
        a  input/output array
        n_dat  number of data
*/
float a[];
int n_dat;
{
    int i;
    float mean;

    mean = 0.0;
    for (i = 1; i <= n_dat; i++)
        mean += a[i];
    mean /= (float) n_dat;
    for (i = 1; i <= n_dat; i++) {
        a[i] -= mean;
    }
}
```

โปรแกรม ข-2 โปรแกรมย่อยสำหรับปรับค่าเฉลี่ยของข้อมูลให้เป็นศูนย์

```
fft(ax, ay, m)
/*
    function to carry out Fast Fourier Transform
    using decimation-in-time algorithm
    Arguments:
        ax  real input/output array
        ay  imaginary input/output array
        m   order of input/output array
    Oct,1986
*/
int m;
float ax[], ay[];
{
    extern double sin(), cos();
    int i, j, l, n, le, le1, ip;
    float ux, uy, wx, wy, tx, ty, tempx, tempy;

    bitrev(ax, m);
    n = 1;
    for (i = 1; i <= m; i++) n *= 2;
    for (l = 1; l <= m; l++) {
        le = 1;
        for (i = 1; i <= l; i++) le *= 2;
        le1 = le / 2;
        ux = 1.0;
        uy = 0.0;
        wx = cos(PI / (float) le1);
        wy = -sin(PI / (float) le1);
        for (j = 1; j <= le1; j++) {
            for (i = j; i <= n; i += le) {
                ip = i + le1;
                tx = ax[ip] * ux - ay[ip] * uy;
                ty = ay[ip] * ux + ax[ip] * uy;
                ax[ip] = ax[i] - tx;
                ay[ip] = ay[i] - ty;
                ax[i] = ax[i] + tx;
                ay[i] = ay[i] + ty;
            }
            tempx = ux * wx - uy * wy;
            tempy = uy * wx + ux * wy;
            ux = tempx;
            uy = tempy;
        }
    }
}
```

โปรแกรม ข-3 โปรแกรมย่อยสำหรับทำ FFT

```
bitrev(a, m)
/*
    function to carry out bit reversal
    for Fast Fourier Transform calculation
    Arguments:
       a input/output array of data sequence
       m order of data sequence
*/
int m;
float a[];
{
    int n, i, j, mv2, nm1, k;
    float t;

    n = 1;
    for (i = 1; i <= m; i++) n *= 2;
    mv2 = n / 2;
    nm1 = n - 1;
    j = 1;
    for (i = 1; i <= nm1; i++) {
        if (i < j) {
            t = a[j];
            a[j] = a[i];
            a[i] = t;
        }
        k = mv2;
        while (k < j) {
            j -= k;
            k /= 2;
        }
        j += k;
    }
}
```

โปรแกรม ข-3 (ต่อ)

```
main()
{
    FILE *fpin, *fpout, *fopen();
    extern double sqrt(), atan2();
    char fin_name[12], fout_name[12];
    int i, j, k, n, nn, order, n_dat, n_ens, f_col, a_col, b_col, w_type,
        ;
    float ax[TMAX], ay[TMAX], bx[TMAX], by[TMAX], tx[FMAX], ty[FMAX],
        time[TMAX], psd_aa[FMAX], psd_bb[FMAX], txx[FMAX], tyy[FMAX], ttt,
        psd_a, psd_b, psd_ab, phs_a, phs_b, t_mag, t_phs,
        prd, freq_step, temp, coh;

    fft_menu(&n_dat, &n_ens, &f_col, &a_col, &b_col, &prd, &w_type,
         &bias, &order);
    n = 1;
    for (i = 0; i < order; i++) n *= 2;
    for (i = 0; i <= n / 2; i++) {
        txx[i] = tyy[i] = 0.0;
        tx[i] = ty[i] = 0.0;
        psd_aa[i] = psd_bb[i] = 0.0;
    }
    for (k = 1; k <= n_ens; k++) {
        printf("\n                Ensemble %d  Filename ", k);
        scanf("%s", fin_name);
        while ((fpin = fopen(fin_name, "r")) == NULL) {
            printf("                    Can't Open Input File: %s\n", fin_name);
            printf("                    Ensemble %d  Filename ", k);
            scanf("%s", fin_name);
        }
        for (i = 1; i <= n; i++) {
            ax[i] = ay[i] = 0.0;
            bx[i] = by[i] = 0.0;
        }
        for (i = 1; i <= n_dat; i++) {
            for (j = 1; j <= f_col; j++) {
                if (j == a_col)
                    fscanf(fpin, "%f", &ax[i]);
                else if (j == b_col)
                    fscanf(fpin, "%f", &bx[i]);
                else
                    fscanf(fpin, "%f", &temp);
            }
        }
        fclose(fpin);
```

โปรแกรม ข-4 โปรแกรมหลักสำหรับคำนวณหาทรานสเฟอร์ฟังก์ชัน

```
        window(w_type, ax, n_dat);
        if (mean == 1)
                adj(ax, n_dat);
        fft(ax, ay, order);
        if (b_col != 0) {
            window(w_type, bx, n_dat);
            if (mean == 1)
                    adj(bx, n_dat);
            fft(bx, by, order);
        }
        nn = n / 2;
        for (i = 1; i <= nn; i++) {
            ax[i] *= prd;
            ay[i] *= prd;
            psd_a = (ax[i] * ax[i] + ay[i] * ay[i]);
            psd_aa[i] += psd_a;
            if (b_col != 0) {
                bx[i] *= prd;
                by[i] *= prd;
                psd_bb[i] += (bx[i] * bx[i] + by[i] * by[i]);
                txx[i] += (ax[i] * bx[i] + ay[i] * by[i]);
                tyy[i] += (by[i] * ax[i] - ay[i] * bx[i]);
            }
            else {
                tx[i] += ax[i];
                ty[i] += ay[i];
            }
        }
    }
    for (i = 1; i <= n / 2; i++) {
        psd_aa[i] /= (float) n_ens;
        if (b_col != 0) {
            psd_bb[i] /= (float) n_ens;
            txx[i] /= (float) n_ens;
            tyy[i] /= (float) n_ens;
            ttt = txx[i] * txx[i] + tyy[i] * tyy[i];
            tx[i] = psd_bb[i] * txx[i] / ttt;
            ty[i] = psd_bb[i] * tyy[i] / ttt;
        }
        else {
            tx[i] /= (float) n_ens;
            ty[i] /= (float) n_ens;
        }
    }
    printf("\n                OUTPUT FILENAME ");
    scanf("%s", fout_name);
    while ((fpout = fopen(fout_name, "w")) == NULL) {
        printf("                    Can't Open Output File: %s\n", fout_name);
        printf("                    Output Filename ");
        scanf("%s", fout_name);
    }
```

โปรแกรม ข-4 (ต่อ)

```
    for (i = 2; i <= nn; i++) {
        t_phs = atan2(ty[i], tx[i]);
        if (b_col != 0) {
            coh = (txx[i] * txx[i] + tyy[i] * tyy[i]) / psd_aa[i] / psd_bb[i];
            t_mag = sqrt(tx[i] * tx[i] + ty[i] * ty[i]);
            fprintf(fpout, "%e %e %e %e %e %e\n",
                (((float) i) - 1.0) * freq_step,
                tx[i], ty[i], t_mag, t_phs, coh);
        }
        else {
            temp = ((float) n) * prd / 2.0;
            psd_a = psd_aa[i] / temp;
            fprintf(fpout, "%e %e %e %e %e\n",
                (((float) i) - 1.0) * freq_step,
                tx[i], ty[i], psd_a, t_phs);
        }
    }
    fclose(fpout);
}
```

โปรแกรม ข-4 (ต่อ)

ข-2 โปรแกรมประมาณค่าพารามิเตอร์ ประกอบด้วยตัวโปรแกรมย่อยที่สำคัญดังนี้ (ดูบทที่ 4 หัวข้อ 4.4 ประกอบ)

```
transp(a, ndat, n, m, at)
/*
     transpose of a matrix ( for iv program only !)
     a = input matrix
     at = output matrix
*/
int ndat, n, m;
float a[][PMAX], at[][NMAX2];
{
     extern int ndat2;
     int i, j;

     for(i = 1; i <= ndat2; i++){
          for(j = 0; j <= n + m; j++)
               at[j][i] = a[i][j];
     }
}

amulti(at, a, ndat, n, m, ata)
/*
     matrix multiplication ( for iv program only !)
     [ata] = [at][a]
*/
int ndat, n, m;
float at[][NMAX2], a[][PMAX], ata[][PMAX];
{
     extern int ndat2;
     int i, j, k;

     for(i = 0; i <= n + m; i++){
          for(k = 0; k <= n + m; k++){
               ata[i][k] = 0;
               for(j = 1; j <= ndat2; j++)
                    ata[i][k] += at[i][j] * a[j][k];
          }
     }
}

bmulti(at, b, ndat, n, m, atb)
/*
     matrix multiplication ( for iv program only !)
     [ata] = [at][a]
*/
int ndat, n, m;
float at[][NMAX2], b[], atb[];
{
     extern int ndat2;
     int i, j;

     for(i = 0; i <= n + m; i++){
         atb[i] = 0;
         for(j = 1; j <= ndat2; j++)
             atb[i] += at[i][j] * b[j];
     }
}
```

โปรแกรม ข-5 โปรแกรมย่อยสำหรับทำการคำนวณพื้นฐานของเมตริก

```
aform(freq, rdat, idat, ndat, n, m, a)
/*
    function to form A-matrix
*/
int ndat, n, m;
float freq[], rdat[], idat[], a[][PMAX];
{
    extern int ndat2;
    int i, j, k;
    float sign, temp, ftemp;

    for (i = 1; i <= ndat2; i++) {
        for (j = 0; j <= m + n; j++)
            a[i][j] = 0;
    }
    for (i = 1; i <= ndat2; i += 2) {
        k = (i + 1) / 2;
        ftemp = 1;
        sign = -1;
        for (j = 0; j <= n; j += 2) {
            sign = -sign;
            a[i][j] = sign * ftemp;
            ftemp *= freq[k] * freq[k];
        }
    }
    for (i = 2; i <= ndat2; i += 2) {
        k = (i + 1) / 2;
        ftemp = freq[k];
        sign = -1;
        for (j = 1; j <= n; j += 2) {
            sign        ;
            a[i][j] = sign        ;
            ftemp *= freq[k] * freq[k];
        }
    }
    for (i = 1; i <= ndat2; i += 2) {
        k = (i + 1) / 2;
        ftemp = 1;
        sign = -1;
        for (j = n + 1; j <= m + n; j += 2) {
            a[i][j] = sign * rdat[k] * ftemp;
            ftemp *= freq[k] * freq[k];
            sign = -sign;
        }
```

โปรแกรม ข-6 โปรแกรมย่อยสำหรับจัดรูปเมตริก A

```
            ftemp = freq[k];
            sign = -1;
            for (j = n + 2; j <= m + n; j += 2) {
                sign = -sign;
                a[i][j] = sign * idat[k] * ftemp;
                ftemp *= freq[k] * freq[k];
            }
        }
        for (i = 2; i <= ndat2; i += 2) {
            k = (i + 1) / 2;
            ftemp = 1;
            sign = -1;
            for (j = n + 1; j <= m    ; j += 2) {
                a[i][j] = sign * idat[k] * ftemp;
                sign = -sign;
                ftemp *= freq[k] * freq[k];
            }
            ftemp = freq[k];
            sign = -1;
            for (j = n + 2; j <= m + n; j += 2) {
                a[i][j] = sign * rdat[k] * ftemp;
                ftemp *= freq[k] * freq[k];
                sign = -sign;
            }
        }
    }
```

โปรแกรม ข-6 (ต่อ)

```
bform(freq, rdat, idat, ndat, m, b)
/*
    routine to form 'b' vector
    Arguments :
    input-      rdat = real parts of frequency response data
                idat = imaginary parts of frequency response data
                ndat = number of frequency response data
    output-     b = b-vector
*/
int ndat, m;
float freq[], rdat[], idat[], b[];
{
    extern int ndat2;
    int k, kk, code, sign;
    float temp[NMAX];

    code = 0;
    for (kk = 1; kk <= m; kk++) {
        code += 1;
        if (code > 4)
            code = 1;
    }
    for (kk = 1; kk <= ndat; kk++) {
        temp[kk] = 1.0;
        for (k = 1; k <= m; k++)
            temp[kk] *= freq[kk];
    }
    sign = -1;
    if ((code ==  ) | (code ==  )) {
        if (code == 1)
            sign *= -1;
        for (k = 1; k <= ndat2; k += 2) {
            kk = (k + 1) / 2;
            b[k] = -idat[kk] * temp[kk] * sign;
        }
        for (k = 2; k <= ndat2; k += 2) {
            kk = (k + 1) / 2;
            b[k] = rdat[kk] * temp[kk] * sign;
        }
    }
    if ((code ==  ) | (code ==  )) {
        if (code ==  )
            sign *= -1;
        for (k = 1; k <= ndat2; k += 2) {
            kk = (k + 1) / 2;
            b[k] = -rdat[kk] * temp[kk] * sign;
        }
        for (k = 2; k <= ndat2; k += 2) {
            kk = (k + 1) / 2;
            b[k] = -idat[kk] * temp[kk] * sign;
        }
    }
}
```

โปรแกรม ข-7 โปรแกรมย่อยสำหรับจัดรูปเวคเตอร์ b

```
solve(wta, p, wtb, n, m)
/*
    program to solve linear systems of equations, Ax = b,
    using Gauss elimination method with partial pivotting.
    ( for iv program only! )
*/
int n, m;
float wta[][PMAX], wtb[], p[];
{
    int i, j, k, imax;
    float a[PMAX][PMAX], b[PMAX], mul[PMAX], amax, atemp, btemp, axtemp;

    for (i = 0; i <= n + m; i++) {
        b[i] = wtb[i];
        for (j = 0; j <= n + m; j++)
            a[i][j] = wta[i][j];
    }
/* forward elimination */
    for (k = 0; k <= n + m; k++) {
            /* find i of amax, imax */
        amax = a[k][k];
        imax = k;
        for (i = k + 1; i <= n + m; i++) {
            if (a[i][k] > amax) {
                amax = a[i][k];
                imax = i;
            }
        }
            /* interchange */
        for (j = k; j <= n + m; j++) {
            atemp = a[imax][j];
            a[imax][j] = a[k][j];
            a[k][j] = atemp;
        }
        btemp = b[imax];
        b[imax] = b[k];
        b[k] = btemp;
            /* multiplier calculation */
        for (i = k + 1; i <= n + m; i++) {
            mul[i] = -a[i][k] / a[k][k];
            b[i] += mul[i] * b[k];
            for (j = k; j <= n + m; j++)
                a[i][j] += mul[i] * a[k][j];
        }
    }
/* back substitution */
    for (k = n + m; k >= 0; k--) {
        axtemp = 0.;
        for (j = k + 1; j <= n + m; j++)
            axtemp += a[k][j] * p[j];
        p[k] = (b[k] - axtemp)/a[k][k];
    }
}
```

โปรแกรม ข-8 โปรแกรมย่อยสำหรับหาคำตอบของระบบสมการเชิงเส้น

```
model(freq, rdat, idat, ndat, n, m, p)
/*
    routine to generate w-matrix for IV algorithm.
int n, m, ndat;
float freq[], rdat[], idat[], p[];
{
    int i, j, k, sign;
    float nr[NMAX], ni[NMAX], mr[NMAX], mi[NMAX], pfreq, cdat;

    for (k = 1; k <= ndat; k++) {
        nr[k] = 0;
        pfreq = 1;
        sign = -1;
        for (i = 0; i <= n; i += 2) {
            sign *= -1;
            nr[k] += p[i] * pfreq * sign;
            pfreq *= freq[k] * freq[k];
        }
        ni[k] = 0;
        pfreq = freq[k];
        sign = -1;
        for (i = 1; i <= n; i += 2) {
            sign *= -1;
            ni[k] += p[i] * pfreq * sign;
            pfreq *= freq[k] * freq[k];
        }
        p[n + m + 1] = 1.0;
        mr[k] = p[n + 1];
        pfreq = freq[k] * freq[k];
        sign = -1;
        for (i = n + 3; i <= n + m + 1; i += 2) {
            mr[k] += p[i] * pfreq * sign;
            pfreq *= freq[k] * freq[k];
            sign *= -1;
        }
        mi[k] = 0;
        pfreq = freq[k];
        sign = -1;
        for (i = n + 2; i <= n + m + 1; i += 2) {
            sign *= -1;
            mi[k] += p[i] * pfreq * sign;
            pfreq *= freq[k] * freq[k];
        }
        cdat = (mr[k] * mr[k] + mi[k] * mi[k]);
        rdat[k] = (nr[k] * mr[k] + ni[k] * mi[k]) / cdat;
        idat[k] = (ni[k] * mr[k] - nr[k] * mi[k]) / cdat;
    }
}
```

โปรแกรม ข-9 โปรแกรมย่อยสำหรับสร้างข้อมูลผลตอบความถี่จากค่าพารามิเตอร์ของทรานสเฟอร์ฟังก์ชัน

```
iv(freq, rdat, idat, ndat, n, m, e, iterm, p)
int ndat, n, m, iterm;
float freq[], rdat[], idat[], e, p[];
{
    int i, j, er, ert;
    float a[NMAX2][PMAX], b[NMAX2], w[NMAX2][PMAX], wt[PMAX][NMAX2],
     wta[PMAX][PMAX], wtb[PMAX], pp[PMAX], ee[PMAX], dp;

    iter = 0;
    aform(freq, rdat, idat, ndat, n, m, a);
    bform(freq, rdat, idat, ndat, m, b);
    for (i = 1; i <= ndat2; i++) {
        for (j = 0; j <= n + m; j++) {
            w[i][j] = a[i][j];
        }
    }
    transp(w, ndat, n, m, wt);
    amulti(wt, a, ndat, n, m, wta);
    bmulti(wt, b, ndat, n, m, wtb);
    solve(wta, p, wtb, n, m);
    printf("\ninitial value for IV-algorithm\nby the least square method\n");
    for (i = 0; i <= n; i++) {
        pi[i][iter] = p[i];
        printf("    b[%d] = %e\n", i, p[i]);
    }
    for (i = 1; i <= m; i++) {
        pi[n + i][iter] = p[n + i];
        printf("    a[%d] = %e\n", i - 1, p[n + i]);
    }
    er = 1;
    while ((ert = er) > 0) {
        iter += 1;
        model(freq, rdat, idat, ndat, n, m, p);
        aform(freq, rdat, idat, ndat, n, m, w);
        transp(w, ndat, n, m, wt);
        amulti(wt, a, ndat, n, m, wta);
        bmulti(wt, b, ndat, n, m, wtb);
        solve(wta, pp, wtb, n, m);
        for (i = 0; i <= n; i++) {
            pi[i][iter] = pp[i];
        }
        for (i = 1; i <= m; i++) {
            pi[n + i][iter] = pp[n + i];
        }
        for (i = 0; i <= n + m; i++)
            ee[i] = p[i] * e / 100;
        er = 0;
```

โปรแกรม ข-10 โปรแกรมย่อยสำหรับหาค่าพารามิเตอร์โดยวิธีตัวแปรอินสทรูเมนทอล

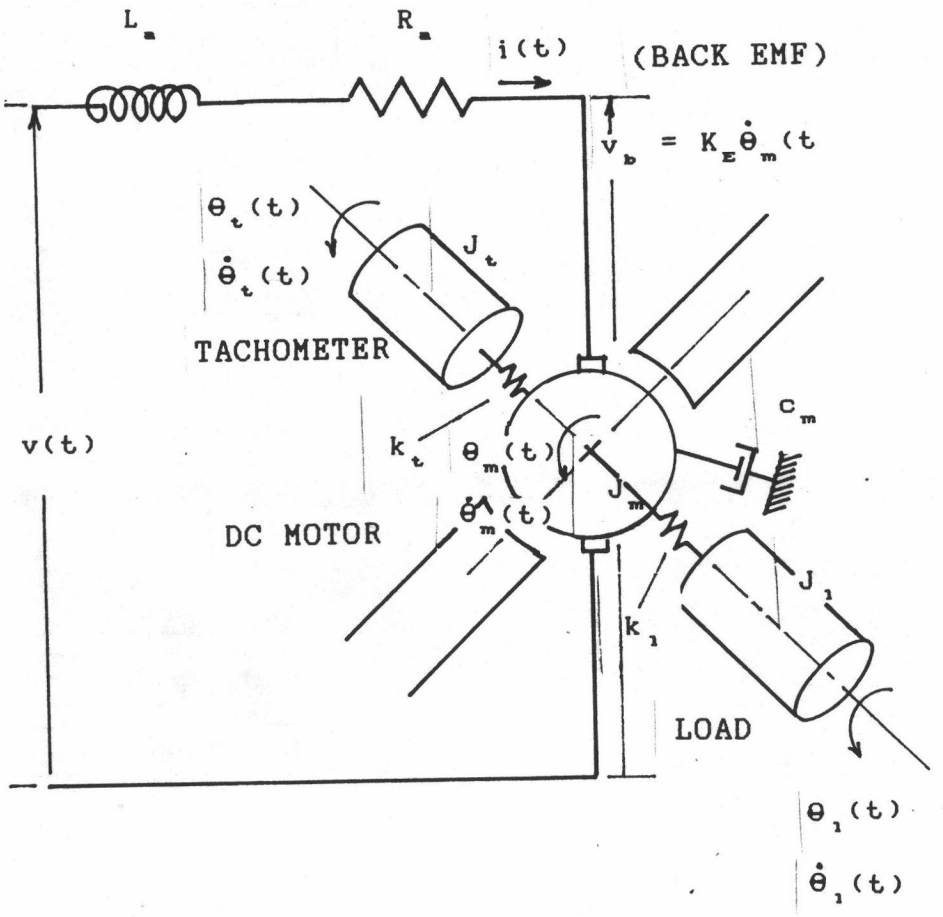
```
            for (i = 0; i <= n + m; i++) {
                dp = pp[i] - p[i];
                if (dp < 0)
                    dp *= -1;
                if (dp > ee[i])
                    er += 1;
            }
            if (iter >= iterm)
                er = 0;
            for (i = 0; i <= n + m; i++)
                p[i] = pp[i];
        }
    }
```

โปรแกรม ข-10 (ต่อ)

ภาคผนวก ค

สมมติฐานต่างๆ ในการหาทรานสเฟอร์ฟังก์ชันของชุดขับเคลื่อนมอเตอร์กระแสตรง

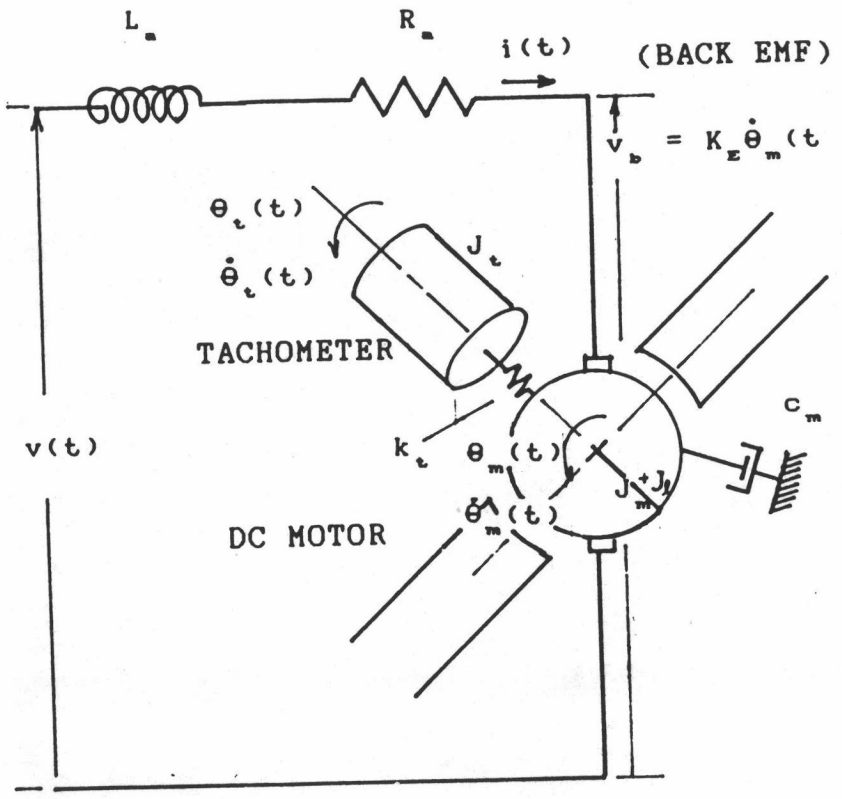
1. เมื่อคำนึงถึงสติฟเนสระหว่างมอเตอร์กับโหลด และระหว่างมอเตอร์กับแทคโคมิเตอร์ (รูป ค-1) เมื่อหาทรานสเฟอร์ฟังก์ชันระหว่างอินพุทคือ โวลเตจที่ป้อนให้กับมอเตอร์กับความเร็วเชิงมุมของแทคโคมิเตอร์ ตามวิธีการในบทที่ 2 จะได้ออเดอร์ของเศษเป็น 2 และของส่วนเป็น 6



K_E = Motor Voltage Constant

รูป ค-1 แสดงระบบมอเตอร์กระแสตรงตามสมมติฐานในข้อ 1

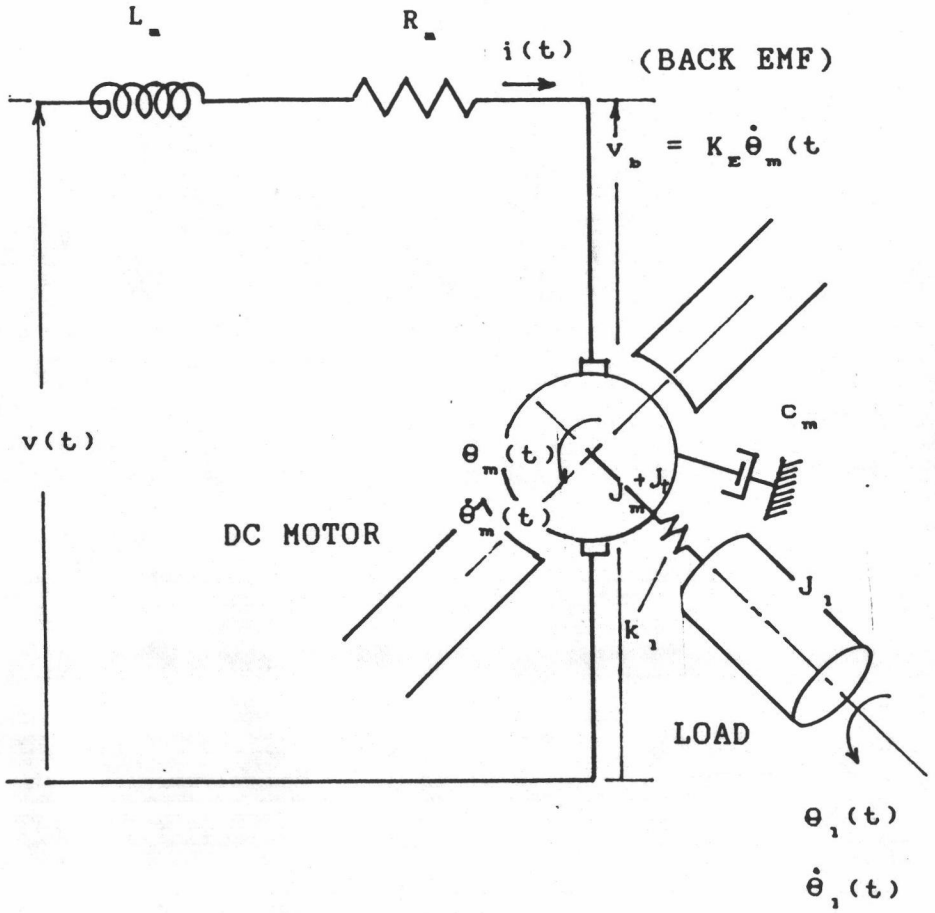
2. เมื่อไม่คิดสติฟเนสระหว่างมอเตอร์กับโหลด แต่คำนึงถึงสติฟเนสระหว่างมอเตอร์กับแทคโคมิเตอร์ (รูป ค-2) เมื่อหาทรานสเฟอร์ฟังก์ชันระหว่างอินพุทคือ โวลเตจที่ป้อนให้กับมอเตอร์กับความเร็วเชิงมุมของแทคโคมิเตอร์ ตามวิธีการในบทที่ 2 จะได้ออเดอร์ของเศษเป็น 0 และของส่วนเป็น 4



K_E = Motor Voltage Constant

รูป ค-2 แสดงระบบมอเตอร์กระแสตรงตามสมมติฐานในข้อ 2

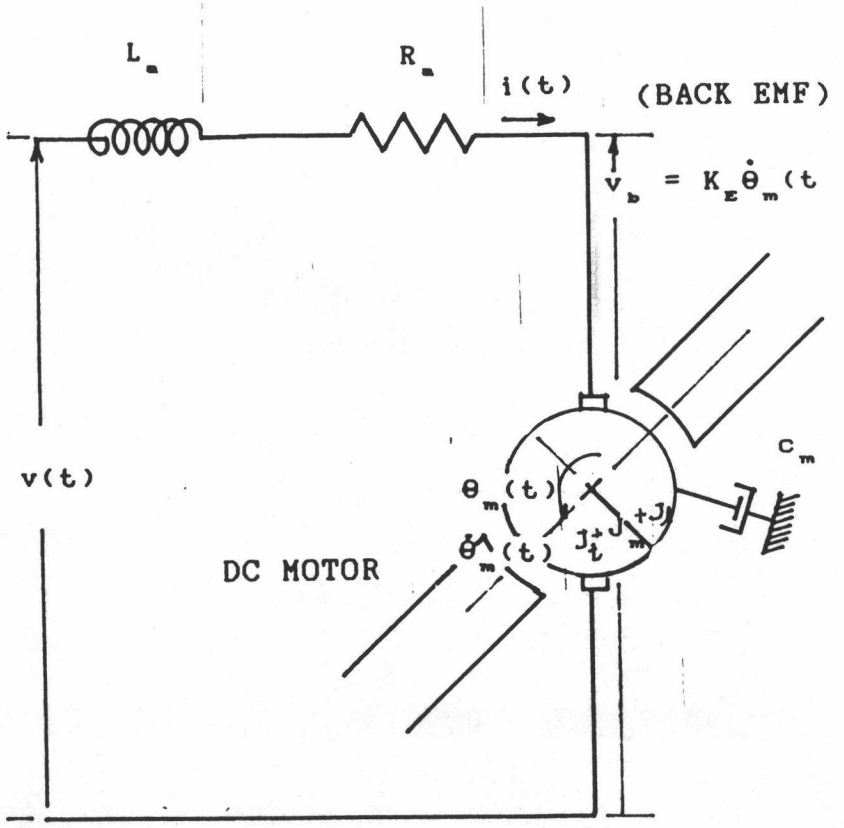
3. เมื่อคำนึงถึงสติฟเนสระหว่างมอเตอร์กับโหลด แต่ไม่คิดสติฟเนสระหว่างมอเตอร์กับแทคโคมิเตอร์ (รูป ค-3) ซึ่งเมื่อหาทรานสเฟอร์ฟังก์ชันระหว่างอินพุทคือ โวลเตจที่ป้อนให้กับมอเตอร์กับความเร็วเชิงมุมของแทคโคมิเตอร์ ตามวิธีการในบทที่ 2 จะได้ออเดอร์ของเศษเป็น 2 และของส่วนเป็น 4



K_E = Motor Voltage Constant

รูป ค-2 แสดงระบบมอเตอร์กระแสตรงตามสมมติฐานในข้อ 3

4. เมื่อไม่คิดทั้งสติฟเนสระหว่างมอเตอร์กับโหลด และสติฟเนสระหว่างมอเตอร์กับแทคโคมิเตอร์ (รูป ค-4) เมื่อหาทรานสเฟอร์ฟังก์ชันระหว่างอินพุทคือ โวลเตจที่ป้อนให้กับมอเตอร์กับความเร็วเชิงมุมของแทคโคมิเตอร์ ตามวิธีการในบทที่ 2 จะได้ออเดอร์ของเศษเป็น 0 และของส่วนเป็น 2



K_E = Motor Voltage Constant

รูป ค-4 แสดงระบบมอเตอร์กระแสตรงตามสมมติฐานในข้อ 4

ภาคผนวก ง

รายการอุปกรณ์การทดลองที่ใช้ในการวิจัย

อุปกรณ์ต่างๆ ที่ใช้ในการวิจัยมีดังนี้

1. ไมโครคอมพิวเตอร์ ขนาด 16 บิท คอมแพททิเบิล (Compatible) กับ IBM XT ใช้ CPU เบอร์ 8088 พร้อมด้วย Math Co-processor 8087 หน่วยความจำ 640 K และมอนิเตอร์แบบกราฟฟิก (ใช้บอร์ด CGA)

2. บอร์ด A/D (เสียบพ่วงกับสลอตในเครื่องไมโครคอมพิวเตอร์ที่ใช้) รุ่น 2801-A บริษัท Data Translation ประเทศสหรัฐอเมริกา

3. เครื่องวิเคราะห์เสปคตรัม (Spectrum Analyzer) รุ่น 5820A แบบ Cross-Channel ช่วงกว้างความถี่ 0-50 kHz บริษัท Wavetek Rockland เมืองซานดิเอโก ประเทศสหรัฐอเมริกา

4. ชุดทดลองมอเตอร์กระแสตรง รุ่น Modular Servo System MS-150 บริษัท Feedback Instruments ประเทศอังกฤษ

5. ตัววัดความเร่ง (Accelerometer) แบบ Piezo รุ่น PV-95 ประเทศญี่ปุ่น

6. เครื่องขยายแบบประจุ (Charge Amplifier) รุ่น 1440-A บริษัท DAWE ประเทศอังกฤษ

ประวัติผู้เขียน

นาย กิตติ ผดุงชีวิต เกิดเมื่อวันที่ 14 มิถุนายน 2506 ณ โรงพยาบาลศิริราช กรุงเทพมหานคร ได้รับปริญญาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมเครื่องกล จากจุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2528 เคยเสนอบทความเรื่องโปรแกรมวิเคราะห์เสปคตรัมสำหรับการทดสอบโมดิล ร่วมกับ อ.ดร. วิบูลย์ แสงวีระพันธุ์ศิริ ในการประชุมทางวิชาการสาขาวิศวกรรมเครื่องกลครั้งที่ 1 ณ สถาบันเทคโนโลยีพระจอมเกล้า วิทยาเขตเจ้าคุณทหาร เมื่อวันที่ 18 มิถุนายน 2530 ปัจจุบันทำงานในตำแหน่งวิศวกรประจำบริษัทไทยซีอาร์ที จำกัด