第 11 卷 第 4 期　　电 子 科 学 学 刊　　Vol. 11. No. 4
1989 年 7 月　　JOURNAL OF ELECTRONICS　　July, 1989

AR 谱估计的一种自适应算法*

余 辉 里
(航空工业部 634 所,北京)

摘要　本文给出 Householder 变换的一种实时递推算法及其收敛性证明，并用此算法进行 AR 自适应谱估计．仿真计算表明，此算法具有计算结果准确、可以实时判阶和快速跟踪性好等特点．

关键词　Householder 变换;实时递推算法;收敛性;自适应谱估计;实时判阶

1. 问题的提出

AR 过程可以表为

$$x(n) = -\sum_{i=1}^{P} a_i x(n-i) + e(n) \tag{1}$$

式中 $x(n)$ 是 AR 序列，P 是模型阶次，$e(n)$ 是方差为 1 的零均白噪序列．自适应 AR 谱估计就是由所获得的 AR 序列实时递推估计 AR 参数并求出功率谱．另外人们在研究中还经常计算双正弦加噪序列

$$x(n) = b_1 \sin(2\pi f_1 n\Delta t + \theta_1) + b_2 \sin(2\pi f_2 n\Delta t + \theta_2) + \varepsilon(n)$$

或单正弦加噪序列 ($b_2 = 0$)，式中 b_1，b_2 为正弦分量的幅值，f_1，f_2 为正弦频率，θ_1，θ_2 为初相位，Δt 为采样间隔，$\varepsilon(n)$ 为零均白噪序列．用自适应算法可以实时估计正弦加噪序列的功率谱，然后由谱峰位置确定序列中所含的频率分量．

目前主要有三类自适应谱估计算法：(1) 基于系统识别技术的最小二乘法 (LS)[1]、广义最小二乘法 (GLS)、辅助变量法 (IV) 和最大似然法 (ML)[2-5]．这类算法的缺点是计算量大，每更新一个数据需要 $O(P^3)$ 计算量．近年来快速 LS 法的研究取得了重大进展，计算量已降到 $O(P)$[6,7]．(2) 梯度法[1,8]．这类算法的计算量虽然是 $O(P)$，但是收敛速度慢，而且很难一次选定收敛因子．(3) 各种网格法[9,10]．这类算法的计算量为 $O(P^2)$ 或 $O(P)$，它常常是前两种算法与网格法的组合．近年来虽然对自适应谱估计算法的研究取得了很大的进展，但是实时判阶、计算准确性以及对时变谱的快速跟踪性等问题仍然是自适应谱估计研究中引人注目的问题．本文提出的自适应算法是基于 Householder 变换(简记为H变换)的 LS 算法，此算法的计算量虽然为 $O(P^2)$，但是可以得到较高质量的谱估计性能．

2. H 变换实时递推算法及其收敛性

设 $\boldsymbol{X} \in \boldsymbol{R}^{N\times L}$，$\boldsymbol{A} \in \boldsymbol{R}^{L}$，$\boldsymbol{B} \in \boldsymbol{R}^{L}$，$\boldsymbol{E} \in \boldsymbol{R}^{L}$，$N \geqslant L$，矩阵方程 $\boldsymbol{XA} - \boldsymbol{B} = \boldsymbol{E}$ 的 LS

* 1987 年 9 月 2 日收到，1988 年 3 月 26 日修改定稿．

解可以表为

$$\boldsymbol{A}=(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{B}$$

在实际应用中常称 $\boldsymbol{A}$ 为参数向量，$\boldsymbol{E}^T\boldsymbol{E}$ 为模型的 L 阶残差平方和．用H变换也可以得到 LS 解

$$\boldsymbol{R}_L\boldsymbol{A}=\boldsymbol{b}_L$$

式中 $\boldsymbol{R}_L$ 是 L 维上三角阵，$\boldsymbol{b}_L$ 是 L 维向量．

在(1)式中用 L 代替 P，取 $n=L+1,\cdots,N$，以 $\boldsymbol{A}\triangleq[a_L,\cdots,a_1,1]^T$ 为顺序将(1)式表为矩阵方程

$$\boldsymbol{X}\boldsymbol{A}=\boldsymbol{E} \tag{2}$$

对 $\boldsymbol{X}$ 阵做H变换使其上三角化，则上三角阵中对角线元素的平方 (d_{m+1}^2) 就是模型的各阶残差平方和 $J(m)$[11]，即

$$J(m)=d_{m+1}^2,\ m=0,\cdots,L \tag{3}$$

文献[11]中给出了H变换的非实时递推算法，可以递推判定阶次并逐列递推求出上三角阵．如果设 $x(t,k)_{i,j}$ 表示对矩阵 $\boldsymbol{X}$ 做H变换后的第 i 行第 j 列元素，(t,k) 表示H变换中的第 t 步实时递推中的第 R 次H变换．并设实时递推的 L 维初值方阵为

$$\boldsymbol{X}(0,0)=\alpha\boldsymbol{I} \tag{4}$$

α 与矩阵计算中所需保留的有效位数有关，一般选 $10^{-8}\leqslant\alpha\leqslant10^{-6}$（详见有关收敛性的讨论）．如下构造第 t 步实时递推矩阵

$$\boldsymbol{X}(t,0)=\begin{bmatrix}\boldsymbol{R}(t-1)\\ \hline \boldsymbol{\phi}(t)\end{bmatrix},\ t=1,2,\cdots \tag{5}$$

式中 $\boldsymbol{R}(t-1)$ 为第 $t-1$ 步递推后的上三角阵，$\boldsymbol{\phi}(t)$ 为第 t 步递推的新数据向量．则由文献[11]中的非实时递推公式可得如下实时递推公式

$$\left.\begin{aligned}
&x(t,0)_{L+1,j}=\varphi(t)_j,\ \boldsymbol{\phi}(t)\text{ 的第 }j\text{ 个元素}\\
&d(t,k)=[x^2(t,k-1)_{k,k}+x^2(t,k-1)_{L+1,k}]^{\frac{1}{2}}\\
&f(t,k)=x(t,k-1)_{k,k}/d(t,k)\\
&g(t,k)=x(t,k-1)_{L+1,k}/d(t,k)\\
&x(t,d)_{k,j}=g(t,k)x(t,k-1)_{L+1,j}+f(t,k)x(t,k-1)_{k,j}\\
&x(t,k)_{L+1,j}=g(t,k)x(t,k-1)_{k,j}-f(t,k)x(t,k-1)_{L+1,j}\\
&k=1,\cdots,L-1;\ j=k+1,\cdots,L;\ t=1,2,\cdots
\end{aligned}\right\} \tag{6}$$

如果忽略 α 的影响，H变换的实时递推算法与非实时递推算法的计算结果是相同的用 $\boldsymbol{H}(t)$ 表示第 t 步实时递推中的 H变换，$\boldsymbol{H}_t$ 表示与 $\boldsymbol{H}(t)$ 相对应的非实时递推中的H变换．当 $t=1$ 时，显然有

$$\boldsymbol{H}(1)\begin{bmatrix}\alpha\boldsymbol{I}\\ \hline \boldsymbol{\phi}(1)\end{bmatrix}=\boldsymbol{H}_1\begin{bmatrix}\alpha\boldsymbol{I}\\ \hline \boldsymbol{\phi}(1)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(1)\\ \hline \boldsymbol{0}\end{bmatrix}$$

当 $t=2$ 时有

$$\boldsymbol{H}(2)\begin{bmatrix}\boldsymbol{R}(1)\\ \hline \boldsymbol{\phi}(2)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(2)\\ \hline \boldsymbol{0}\end{bmatrix}$$

$$H_2\begin{bmatrix}\boldsymbol{R}(1)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{\phi}(2)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}^*(2)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{0}\end{bmatrix}$$

由于上三角阵的递推计算与零行元素无关，所以必有 $\boldsymbol{R}(2)=\boldsymbol{R}^*(2)$. 当 $t=N-1$ 时，设有

$$\boldsymbol{H}(N-1)\begin{bmatrix}\boldsymbol{R}(N-2)\\ \hline \boldsymbol{\phi}(N-1)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(N-1)\\ \hline \boldsymbol{0}\end{bmatrix}$$

$$\boldsymbol{H}_{N-1}\begin{bmatrix}\boldsymbol{R}(N-2)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{\phi}(N-1)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(N-1)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{0}\end{bmatrix}$$

则当 $t=N$ 时，

$$\boldsymbol{H}(N)\begin{bmatrix}\boldsymbol{R}(N-1)\\ \hline \boldsymbol{\phi}(N)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(N)\\ \hline \boldsymbol{0}\end{bmatrix}$$

$$\boldsymbol{H}_N\begin{bmatrix}\boldsymbol{R}(N-1)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{\phi}(N)\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}^*(N)\\ \hline \boldsymbol{0}\\ \hline \boldsymbol{0}\end{bmatrix}$$

同样可以得到 $\boldsymbol{R}(N)=\boldsymbol{R}^*(N)$. 如果 α 很小，则 $\alpha\boldsymbol{I}$ 对H变换的影响也很小，有

$$\boldsymbol{H}\begin{bmatrix}\alpha\boldsymbol{I}\\ \hline \boldsymbol{\phi}(1)\\ \vdots\\ \boldsymbol{\phi}(N)\end{bmatrix}\approx\boldsymbol{H}\begin{bmatrix}\boldsymbol{\phi}(1)\\ \vdots\\ \boldsymbol{\phi}(N)\end{bmatrix}\triangleq\boldsymbol{H}\boldsymbol{\phi}$$

因此如果取 $10^{-8}\leqslant\alpha\leqslant10^{-6}$ 而忽略其影响，则对 $\boldsymbol{\phi}(t)$ 实时递推计算 N 个数据等于对 N 个数据构造的 $\boldsymbol{\phi}$ 阵做非实时递推计算. 由于 LS 估计是强一致收敛的，因此上述结果证明了H变换实时递推算法的收敛性.

3. 用 H 变换做 AR 自适应谱估计

用(5)式和(6)式对(2)式中的 $\boldsymbol{X}$ 阵做实时递推H变换便可以得到 AR 参数的实时估计，进而求出 AR 自适应谱估计. 设 AR 模型阶次 $P<L$，按(4)式选初始矩阵，取(5)式为

$$\boldsymbol{\phi}(t)=[x(t),\cdots,x(t+L)] \tag{7}$$

便可以用(6)式进行实时递推. 为了有效地判定阶次，在 L 步递推后再利用对角线元素判阶. 如果判阶为 P，则在以后的每一步递推后计算

$$J_P(t)=|d(t,P)|/|d(L,P)|,\quad t=L+1,L+2,\cdots \tag{8}$$

如果 $J_P(t)\leqslant3$，继续递推计算，否则重新判阶. 设在第 t_i 步重新判阶为 P_i，如果 $P_i=P$，继续递推计算，否则设定阶次 P 为 P_i，继续递推计算. (8)式的一般形式为

$$J_P(t) = |d(t, P_i)| / |d(t_i, P_i)|, \quad t = t_i + 1, \ t_i + 2, \cdots, i = 1, 2 \cdots \tag{9}$$

每步递推后判定 (9) 式是否大于 3，以决定是否需要重新判阶，这样就形成了实时判阶方法.

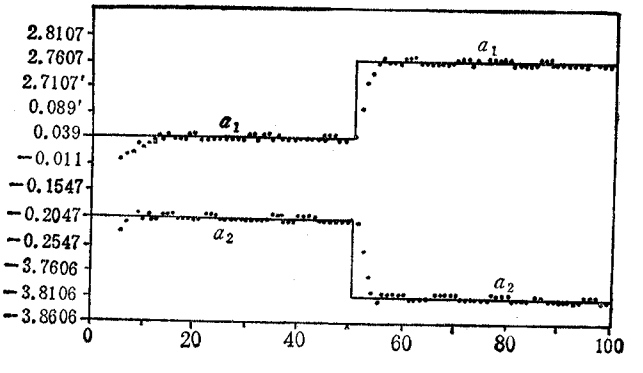
为了减少计算量，判阶后 $\boldsymbol{X}(t, 0)$ 阵的后 $(L - P + 1)$ 列不参与计算. 每步递推后可得到上三角阵 $\boldsymbol{R}(t)$，用回代法便可以求出参数估计 $\{\hat{a}_i(t), i = 1, \cdots, P_i\}$，进而求出 AR 谱估计

$$\hat{P}_i(\omega) = \frac{d^2(t, P_i)}{N(t)\left|1 + \sum_{i=1}^{P_i} \hat{a}_i(t) e^{-j\omega_i}\right|^2}, \quad i = 1, 2, \cdots \tag{10}$$

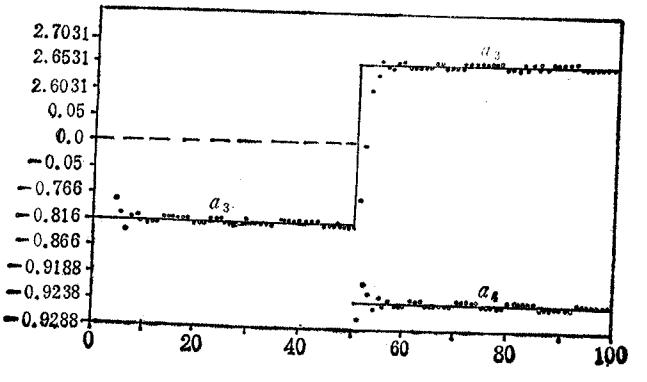
式中 $N(t)$ 是到第 t 步递推为止所使用的数据量，$d(t, P_i)$ 是阶次 P_i 所对应的对角线元素. 这样就构成了基于H变换的 AR 自适应谱估计算法.

用H变换做自适应谱估计，每增加一个新数据需要 $O(P^2)$ 计算量. 由于H变换是一种算法稳定性很好的 LS 估计算法，而H变换实时递推计算等价于对(2)式中的 $\boldsymbol{X}$ 阵直接做H变换，因此本文提出的自适应谱估计算法具有较高的估计质量.

4. 仿真计算



(a)



(b)

图 1　本文算法对 AR 参数的估计

例 1 以三阶和四阶 AR 模型为例

$$x_1(n) = 0.039x_1(n-1) - 0.2047x_1(n-2) - 0.816x_1(n-3) + e(n)$$

$$x_2(n) = 2.7607x_2(n-1) - 3.8106x_2(n-2) + 2.6535x_2(n-3) - 0.9238x_2(n-4) + e(n)$$

用计算机生成 $x_1(n)$ 序列，式中 $e(n)$ 是方差为 1 的零均白噪序列. L 步递推后判定阶次 $P=3$，然后继续递推，并用 (9) 式判断是否需要改变阶次. 在 44 步递推后(相当于生成了 50 个 $x_1(n)$ 数据)，生成 $x_2(n)$ 序列作为新数据进行递推计算. 在 48 步递推后用 (9) 式判断需要重新改变阶次，通过判阶得到 $P_1=4$ 后继续递推. 在第 53 步和 60 步递推后都重新判定了阶次，结果 $P_2=P_3=P_1=4$. 在第 94 步递推后计算结束，整个计算过程共用了 100 个数据. 计算结果用 AR 参数表示，如图 1 所示. 为表示清晰起见，图中使用了不等间隔的坐标. 从图 1 可以看出，本文提出的算法不但具有较准确的估计结果，而且可以快速跟踪时变参数.

例 2 以正弦加噪序列为例

$$x_1(n) = \sin(2\pi n\Delta t + \theta) + 0.5\sin(4\pi n\Delta t + \theta) + \varepsilon_1(n)$$

$$x_2(n) = \sin(14.5\pi n\Delta t + \theta) + \varepsilon_2(n)$$

式中 $\Delta t_1 = 0.05s$，$\Delta t_2 = 0.01s$，$\theta = \pi/4$，$\varepsilon_1(n)$ 的方差为 0.05，$\varepsilon_2(n)$ 的方差为 0.01. 用计算机生成 $x_1(n)$ 序列，取 $L=14$，按 AR 模型进行实时递推计算. 与例 1 中的叙述相同，在计算 $x_1(n)$ 序列时判阶为 $P=12$. 在生成 40 个 $x_1(n)$ 数据后生成 $x_2(n)$ 序列并改变信噪比和采样间隔. 对 $x_2(n)$ 的阶次判定为 $P_1=10$. 每步递推求出 AR 参数后，按 (10) 式计算 AR 功率谱并求出谱峰对应的频率. 整个计算共使用 100 个数据. 计算结果 (图 2) 用频率值表示. 从图中可以看出，本文提出的算法具有较好的分辨率与跟踪性.

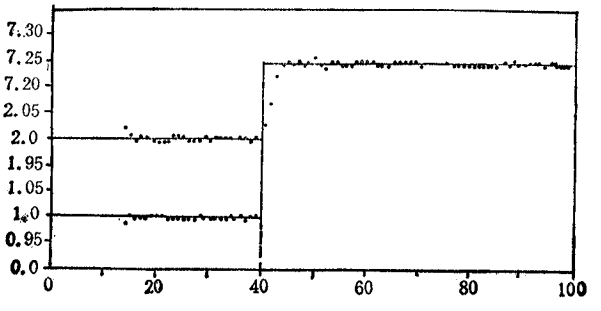


图 2 本文算法对正弦频率的估计

例 2 中正弦加噪过程是按容易产生谱分裂和谱位移的情况选取的[12]，其中 $x_1(n)$ 容易产生谱位移，$x_2(n)$ 容易产生谱分裂. 但是在本文的仿真计算中这两种现象均没有出现. 本文的算法对低信噪比的正弦加噪过程也具有较好的估计质量，由于篇幅所限，在此不再举例.

5. 结论

本文用H变换实时递推算法做 AR 自适应谱估计和实时判阶. 由于H变换具有很

好的算法稳定性，而且H变换实时递推算法的计算结果又与非实时递推算法的计算结果相同，所以本文提出的算法具有较好的估计质量。

参　考　文　献

[1] S. M. Kay, S. L. Marple. *Proc. IEEE*, **69**(1981), 1380—1414.
[2] Y. T. Chan, *IEEE Trans. on ASSP*, **ASSP-29**(1981), 214—219.
[3] M. Dragosevic, et al., *ICASSP*, **3**(1982), 2080—2083.
[4] B. Friedlander, *IEEE Trans. on ASSP*, **ASSP-31**(1983), 405—415.
[5] B. Friedlander, *IEEE Trans. on ASSP*, **ASSP-30**(1982), 240—245.
[6] N. Kalouptsidis, S. Theodoridis, *IEEE Trans. on ASSP*, **ASSP-35**(1987), 661—669.
[7] P. Fabre, C. Gueguen, *IEEE Trans. on ASSP*, **ASSP-34**(1986), 296—308.
[8] Y. Iiguni, et al., *IEEE Trans. on ASSP*, **ASSP-33**(1985), 1427—1434.
[9] B. Friedlander, *Proc. IEEE*, **70**(1982), 829—863.
[10] P. Strobach, *IEEE Trans. on ASSP*, **ASSP-34**(1986), 880—897.
[12] 余辉里，电子科学学刊，**6**(1986)，457—461.
[13] S. L. Marple, *IEEE Trans. on ASSP*, **ASSP-28**(1980), 441—454.

AN ADAPTIVE ALGORITHM OF AR SPECTRAL ESTIMATION

Yu Huili

(Research Institute No.634, Ministry of Aviation Industry, Beijing)

Abstract A real-time recursive algorithm of Householder transform and the proof of its convergency are given, and this algorithm is used for AR adaptive spectral estimation. The simulating computation shows that this algorithm has the features of correct computation results, real-time determination of order and excellent fast tracking.

Key words Householder transform; Real-time recursive algorithm; Convergency; Adaptive spectral estimation; Real-time determination of order