式中 λ_k 为 Lagrange 乘子。求 $J[P(\omega)]$ 相对于 $P(\omega)$ 的偏导，并令其等于零，则有

$$P(\omega)=\frac{1}{\sum_{k=-p}^{p}\lambda_k e^{j\omega k}} \tag{3.6.16}$$

作变量代换 $\mu_k=\lambda_{-k}$，则式 (3.6.16) 可以写作

$$P(\omega)=\frac{1}{\sum_{k=-p}^{p}\mu_k e^{-j\omega k}} \tag{3.6.17}$$

令

$$W(z)=\sum_{k=-p}^{p}\mu_k z^{-k}$$

则 $P(\omega)=1/W(e^{j\omega})$。由于功率谱密度是非负的，故有

$$W(e^{j\omega})>0$$

定理 3.6.1 (Fejer-Riesz定理) [143,p.231] 若

$$W(z)=\sum_{k=-p}^{p}\mu_k z^{-k} \quad 和 \quad W(e^{j\omega})>0$$

则可以找到一函数

$$A(z)=\sum_{i=0}^{p}a(i)\,z^{-i} \tag{3.6.18}$$

使得

$$W(e^{j\omega})=|A(e^{j\omega})|^2 \tag{3.6.19}$$

并且若限制 $A(z)=0$ 的根全部在单位圆内，则函数 $A(z)$ 是惟一确定的。

根据 Fejer-Riesz 定理知，若假定 $a(0)=1$，则式 (3.6.17) 可以表示成

$$P(\omega)=\frac{\sigma^2}{|A(e^{j\omega})|^2} \tag{3.6.20}$$

恰好就是前面的 AR 功率谱密度。这表明，Burg 最大熵功率谱与 AR 功率谱等价。

3.6.2 Levinson 递推

为了实现最大熵谱估计，需要确定阶数 p 和系数 a_i。这就带来一个问题：阶数多大才合适？Burg 提出使用线性预测方法递推计算不同阶数的预测器系数，然后比

较各预测器的预测误差功率。实现这一递推计算的基础是著名的 Levinson 递推 (也称 Levinson-Durbin 递推)。

最大熵方法同时使用前向预测和后向预测。所谓前向预测就是利用给定的 m 个数据 $x(n-m),\cdots,x(n-1)$ 预测 $x(n)$ 的值，并称之为 m 阶前向线性预测。

$x(n)$ 的 m 阶前向线性预测值记为 $\hat{x}(n)$，定义为

$$\hat{x}(n) \stackrel{\text{def}}{=} -\sum_{i=1}^{m} a_m(i)x(n-i) \tag{3.6.21}$$

式中 $a_m(i)$ 表示 m 阶预测器的第 i 个系数。

类似地，利用给定的 m 个数据 $x(n-m+1),\cdots,x(n)$ 预测 $x(n-m)$ 的值称为 m 阶后向线性预测，定义为

$$\hat{x}(n-m) \stackrel{\text{def}}{=} -\sum_{i=1}^{m} a_m^*(i)x(n-m+i) \tag{3.6.22}$$

式中 $a_m^*(i)$ 是 $a_m(i)$ 的复数共轭。

下面讨论如何根据最小均方误差 (MMSE) 准则设计前向和后向预测滤波器。

前、后向预测误差分别定义为

$$f(n) \stackrel{\text{def}}{=} x(n) - \hat{x}(n) = \sum_{i=0}^{m} a_m(i)x(n-i) \tag{3.6.23a}$$

$$g(n-m) \stackrel{\text{def}}{=} x(n-m) - \hat{x}(n-m) = \sum_{i=0}^{m} a_m^*(i)x(n-m+i) \tag{3.6.23b}$$

式中 $a_m(0)=1$。根据正交性原理知，为了使预测值 $\hat{x}(n)$ 是 $x(n)$ 的线性均方估计，则前向预测误差 $f(n)$ 必须与已知的数据 $x(n-m),\cdots,x(n-1)$ 正交，即有

$$\mathrm{E}\{f(n)x^*(n-k)\}=0, \quad 1 \leqslant k \leqslant m \tag{3.6.24}$$

将式 (3.6.23a) 代入式 (3.6.24)，立即得到一组法方程

$$\left.\begin{aligned}
R_x(0)a_m(1)+R_x(-1)a_m(2)+\cdots+R_x(-m+1)a_m(m)&=-R_x(1)\\
R_x(1)a_m(1)+R_x(0)a_m(2)+\cdots+R_x(-m+2)a_m(m)&=-R_x(2)\\
&\vdots\\
R_x(m-1)a_m(1)+R_x(m-2)a_m(2)+\cdots+R_x(0)a_m(m)&=-R_x(m)
\end{aligned}\right\} \tag{3.6.25}$$

式中 $R_x(k)=\mathrm{E}\{x(n)x^*(n-k)\}$ 是 $\{x(n)\}$ 的自相关函数。

定义前向预测均方误差 (即前向预测误差的输出功率) 为

$$\begin{aligned}P_m &\overset{\text{def}}{=} \mathrm{E}\{|f(n)|^2\} = \mathrm{E}\{f(n)[x(n)-\hat{x}(n)]^*\}\\ &= \mathrm{E}\{f(n)x^*(n)\} - \sum_{i=1}^{m} a_m^*(i)\mathrm{E}\{f(n)x^*(n-i)\\ &= \mathrm{E}\{f(n)x^*(n)\}\end{aligned} \tag{3.6.26}$$

式中的求和项 $\sum$ 为零，这是直接代入式 (3.6.24) 的结果。展开式 (3.6.26) 的右边，直接得到

$$P_m = \sum_{i=0}^{m} a_m(i)R_x(-i) \tag{3.6.27}$$

合并式 (3.6.25) 和式 (3.6.27)，即有

$$\begin{bmatrix} R_x(0) & R_x(-1) & \cdots & R_x(-m)\\ R_x(1) & R_x(0) & \cdots & R_x(-m+1)\\ \vdots & \vdots & \vdots & \vdots\\ R_x(m) & R_x(m-1) & \cdots & R_x(0)\end{bmatrix}\begin{bmatrix}1\\ a_m(1)\\ \vdots\\ a_m(m)\end{bmatrix} = \begin{bmatrix}P_m\\ 0\\ \vdots\\ 0\end{bmatrix} \tag{3.6.28}$$

求解式 (3.6.28)，即可直接得到 m 阶前向预测滤波器的系数 $a_m(1),\cdots,a_m(m)$。问题是相应的预测误差功率不一定小。这就需要求出各种可能阶数的前向预测滤波器的系数及相应的预测误差功率。对不同 m 值，独立求解滤波器方程 (3.6.28) 显然过于浪费时间。假定 $m-1$ 阶前向预测滤波器的系数 $a_{m-1}(1),\cdots,a_{m-1}(m-1)$ 业已求出，那么，如何由它们递推计算 m 阶前向预测滤波器的系数 $a_m(1),\cdots,a_m(m)$ 呢？

由式 (3.6.28)，很容易列出 $m-1$ 阶前向预测所满足的滤波器方程

$$\begin{bmatrix} R_x(0) & R_x(-1) & \cdots & R_x(-m+1)\\ R_x(1) & R_x(0) & \cdots & R_x(-m)\\ \vdots & \vdots & \vdots & \vdots\\ R_x(m-1) & R_x(m-2) & \cdots & R_x(0)\end{bmatrix}\begin{bmatrix}1\\ a_{m-1}(1)\\ \vdots\\ a_{m-1}(m-1)\end{bmatrix} = \begin{bmatrix}P_{m-1}\\ 0\\ \vdots\\ 0\end{bmatrix} \tag{3.6.29}$$

再考虑 $m-1$ 阶后向预测误差，由式 (3.6.23b) 有

$$\begin{aligned}g(n-m+1) &\overset{\text{def}}{=} x(n-m+1) - \hat{x}(n-m+1)\\ &= \sum_{i=0}^{m-1} a_{m-1}^*(i)x(n-m+1+i)\end{aligned} \tag{3.6.30}$$

由正交性原理知，为了使 $\hat{x}(n-m+1)$ 是 $x(n-m+1)$ 的线性均方估计，后向预测

误差 $g(n-m+1)$ 应该与已知数据 $x(n-m+2),\cdots,x(n)$ 正交，即

$$\mathrm{E}\{g(n-m+1)x^*(n-m+1+k)\}=0,\quad k=1,\cdots,m-1 \tag{3.6.31}$$

由式 (3.6.30) 和式 (3.6.31)，可以得到

$$\sum_{i=0}^{m-1} a_{m-1}^*(i)R_x(i-k)=0,\quad k=1,\cdots,m-1 \tag{3.6.32}$$

和

$$P_{m-1}=\mathrm{E}\{|g(n-m+1)|^2\}=\sum_{i=0}^{m-1} a_{m-1}^*(i)R_x(i) \tag{3.6.33}$$

式 (3.6.32) 和式 (3.6.33) 合并在一起，可写成矩阵形式

$$\begin{bmatrix} R_x(0) & R_x(-1) & \cdots & R_x(-m+1) \\ R_x(1) & R_x(0) & \cdots & R_x(-m) \\ \vdots & \vdots & \vdots & \vdots \\ R_x(m-1) & R_x(m-2) & \cdots & R_x(0) \end{bmatrix} \begin{bmatrix} a_{m-1}^*(m-1) \\ \vdots \\ a_{m-1}^*(1) \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ P_{m-1} \end{bmatrix} \tag{3.6.34}$$

利用 $a_{m-1}(i)$ 递推计算 $a_m(i)$ 的基本思想是直接使用 $a_{m-1}(i)$ 及一校正项之和作为 $a_m(i)$ 的值。递推公式的这种构成思想广泛应用于信号处理 (例如自适应滤波)、神经网络及神经计算 (例如学习算法) 中。在 Levinson 递推里，校正项由 $m-1$ 阶后向预测滤波器的系数反射而成，即

$$a_m(i)=a_{m-1}(i)+K_m a_{m-1}^*(m-i),\quad i=0,1,\cdots,m \tag{3.6.35}$$

式中 K_m 称为反射系数。

下面是式 (3.6.35) 的两种边缘情况。

- 当 $i=0$ 时，由于 $a_m(0)=a_{m-1}(0)+K_m a_{m-1}^*(m)$ 及 $a_{m-1}(m)=0$，故有 $a_m(0)=a_{m-1}(0)$。这表明，若令 $a_1(0)=1$，则恒有 $a_m(0)=1$，$m\geqslant 2$。
- 当 $i=m$ 时，由于 $a_m(m)=a_{m-1}(m)+K_m a_{m-1}^*(0)$，$a_{m-1}(m)=0$ 以及 $a_{m-1}(0)=1$，所以有 $a_m(m)=K_m$。

除了预测滤波器系数的递推外，还需要预测误差功率 P_m 的递推。注意式 (3.6.35)

可以改写作

$$\begin{bmatrix} 1 \\ a_m(1) \\ \vdots \\ a_m(m-1) \\ a_m(m) \end{bmatrix} = \begin{bmatrix} 1 \\ a_{m-1}(1) \\ \vdots \\ a_{m-1}(m-1) \\ 0 \end{bmatrix} + K_m \begin{bmatrix} 0 \\ a_{m-1}^*(m-1) \\ \vdots \\ a_{m-1}^*(1) \\ 1 \end{bmatrix} \tag{3.6.36}$$

将式 (3.6.36) 代入式 (3.6.28) 后，即有

$$\begin{bmatrix} R_x(0) & R_x(-1) & \cdots & R_x(-m) \\ R_x(1) & R_x(0) & \cdots & R_x(-m+1) \\ \vdots & \vdots & \vdots & \vdots \\ R_x(m) & R_x(m-1) & \cdots & R_x(0) \end{bmatrix} \times$$

$$\left\{ \begin{bmatrix} 1 \\ a_{m-1}(1) \\ \vdots \\ a_{m-1}(m-1) \\ 0 \end{bmatrix} + K_m \begin{bmatrix} 0 \\ a_{m-1}^*(m-1) \\ \vdots \\ a_{m-1}^*(1) \\ 1 \end{bmatrix} \right\} = \begin{bmatrix} P_m \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.6.37}$$

将式 (3.6.29) 和式 (3.6.34) 代入上式，得到

$$\begin{bmatrix} P_{m-1} \\ 0 \\ \vdots \\ 0 \\ \text{---} \\ X \end{bmatrix} + K_m \begin{bmatrix} Y \\ \text{---} \\ 0 \\ \vdots \\ 0 \\ P_{m-1} \end{bmatrix} = \begin{bmatrix} P_m \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

或

$$P_{m-1} + K_m Y = P_m \tag{3.6.38a}$$

$$X + K_m P_{m-1} = 0 \tag{3.6.38b}$$

式中

$$X = \sum_{i=0}^{m-1} a_{m-1}(i) R_x(m-i) \tag{3.6.39a}$$

$$Y = \sum_{i=0}^{m-1} a_{m-1}^*(i) R_x(i-m) \tag{3.6.39b}$$

注意到 $R_x(i-m)=R_x^*(m-i)$，所以式 (3.6.39b) 和式 (3.4.39a) 给出

$$Y=X^* \tag{3.6.40}$$

然而，由式 (3.6.38b) 知

$$X=-K_mP_{m-1}$$

故 $Y=-K_m^*P_{m-1}$，注意预测误差功率 P_{m-1} 为实数。将 $Y=-K_m^*P_{m-1}$ 代入式 (3.6.38a) 后，便得到预测误差功率的递推公式

$$P_m=(1-|K_m|^2)P_{m-1} \tag{3.6.41}$$

总结以上讨论，即有以下递推算法。

算法 3.6.1 (Levinson 递推算法 (上推))

$$a_m(i)=a_{m-1}(i)+K_ma_{m-1}^*(m-i),\quad i=1,\cdots,m-1 \tag{3.6.42a}$$

$$a_m(m)=K_m \tag{3.6.42b}$$

$$P_m=(1-|K_m|^2)P_{m-1} \tag{3.6.42c}$$

当 $m=0$ 时，式 (3.6.27) 给出预测误差功率的初始值

$$P_0=R_x(0)=\frac{1}{N}\sum_{n=1}^{N}|x(n)|^2 \tag{3.6.42d}$$

式 (3.6.42) 称为 (向) 上 (递) 推的 Levinson 递推公式，即由 1 阶预测滤波器的系数递推 2 阶预测滤波器的系数，再由 2 阶预测滤波器递推 3 阶预测滤波器。有的时候，我们对由 m 预测滤波器递推 $(m-1)$ 预测滤波器，再由 $(m-1)$ 阶预测滤波器递推 $(m-2)$ 阶预测滤波器感兴趣。这样一种由高阶滤波器系数求低阶滤波器系数的递推称为 (向) 下 (递) 推。下面是下推的 Levinson 递推算法 [151]。

算法 3.6.2 (Levinson 递推算法 (下推))

$$a_m(i)=\frac{1}{1-|K_{m+1}|^2}[a_{m+1}(i)-K_{m+1}a_{m+1}(m-i+1)] \tag{3.6.43a}$$

$$K_m=a_m(m) \tag{3.6.43b}$$

$$P_m=\frac{1}{1-|K_{m+1}|^2}P_{m+1} \tag{3.6.43c}$$

式中 $i=1,\cdots,m$。

在算法的递推过程中，当 $|K_0|=1$ 时，递推即停止。

3.6.3 Burg 算法

观察 Levinson 递推知，剩余的问题是如何求出反射系数 K_m 的递推公式。这一问题一直困扰着 Levinson 递推的实际应用，直到 Burg[29] 在研究最大熵方法时才得以解决。Burg 算法的基本思想是使前、后向预测误差的平均功率为最小。

Burg 定义 m 阶前、后向预测误差为

$$f_m(n) = \sum_{i=0}^{m} a_m(i)x(n-i) \tag{3.6.44a}$$

$$g_m(n) = \sum_{i=0}^{m} a_m^*(m-i)x(n-i) \tag{3.6.44b}$$

将式 (3.6.42a) 分别代入式 (3.6.44a) 和式 (3.6.44b)，经整理后，即得到前、后向预测误差的阶数递推公式

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1) \tag{3.6.45a}$$

$$g_m(n) = K_m^* f_{m-1}(n) + g_{m-1}(n-1) \tag{3.6.45b}$$

定义 m 阶前、后向预测误差的平均功率

$$P_m = \frac{1}{2}\sum_{n=m}^{N}\left[|f_m(n)|^2 + |g_m(n)|^2\right] \tag{3.6.46}$$

将阶数递推公式 (3.6.45) 代入式 (3.6.46)，并令 $\dfrac{\partial P_m}{\partial K_m} = 0$，则由习题 3.10 知

$$K_m = \frac{-\sum\limits_{n=m+1}^{N} f_{m-1}(n)g_{m-1}^*(n-1)}{\frac{1}{2}\sum\limits_{n=m+1}^{N}\left[|f_{m-1}(n)|^2 + |g_{m-1}(n-1)|^2\right]} \tag{3.6.47}$$

式中 $m = 1, 2, \cdots$。

归纳起来，计算前向预测滤波器系数的 Burg 算法如下。

算法 3.6.3 (Burg 算法)

步骤 1 计算预测误差功率的初始值

$$P_0 = \frac{1}{N}\sum_{n=1}^{N}|x(n)|^2$$

和前、后向预测误差的初始值

$$f_0(n) = g_0(n) = x(n)$$

并令 $m = 1$。

步骤 2 求反射系数

$$K_m = \frac{-\sum\limits_{n=m+1}^{N} f_{m-1}(n) g_{m-1}^*(n-1)}{\frac{1}{2}\sum\limits_{n=m+1}^{N}\left[|f_{m-1}|^2 + |g_{m-1}(n-1)|^2\right]}$$

步骤 3 计算前向预测滤波器系数

$$a_m(i) = a_{m-1}(i) + K_m a_{m-1}^*(m-i),\quad i = 1,\cdots,m-1$$
$$a_m(m) = K_m$$

步骤 4 计算预测误差功率

$$P_m = (1-|K_m|^2)P_{m-1}$$

步骤 5 计算滤波器输出

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1)$$
$$g_m(n) = K_m^* f_{m-1}(n) + g_{m-1}(n-1)$$

步骤 6 令 $m \leftarrow m+1$，并重复步骤 2 至步骤 5，直到预测误差功率 P_m 不再明显减小。

3.6.4 Burg 最大熵谱分析与 ARMA 谱估计

若增加一个约束条件，则 Burg 最大熵功率谱密度与 ARMA 功率谱密度等价，这一结论是 Lagunas [108] 和 Ihara [91] 采用不同方法独立证明的。Lagunas 等人 [109] 从工程应用的角度出发，使用一种很容易理解的方法演绎这一等价性，并提出了具体的最大熵 ARMA 谱估计算法。

考虑对数功率谱密度 $\ln P(\omega)$，其 Fourier 反变换称为倒谱 (cepstrum) 系数，即有

$$c_x(k) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \ln P(\omega) e^{j\omega k} d\omega \qquad (3.6.48)$$

令 $2M+1$ 个样本自相关函数 $\hat{R}_x(m), m=0,\pm1,\cdots,\pm M$ 和 $2N$ 个样本倒谱 $\hat{c}_x(l), l=\pm1,\cdots,\pm N$ 为已知，现在考虑在自相关函数匹配

$$\hat{R}_x(m)=\frac{1}{2\pi}\int_{-\pi}^{\pi}P(\omega)\mathrm{e}^{\mathrm{j}\omega m}\mathrm{d}\omega,\quad m=0,\pm1,\cdots,\pm M \tag{3.6.49a}$$

与倒谱匹配

$$\hat{c}_x(l)=\frac{1}{2\pi}\int_{-\pi}^{\pi}\ln P(\omega)\mathrm{e}^{\mathrm{j}\omega l}\mathrm{d}\omega,\quad l=\pm1,\cdots,\pm N \tag{3.6.49b}$$

这两个条件的约束下，求功率谱密度 $P(\omega)$，使其谱熵 $H[P(\omega)]$ 最大。注意，当 $l=0$ 时，式 (3.6.49b) 定义的倒谱 $\hat{c}_x(0)$ 恰好就是谱熵，故 $\hat{c}_x(0)$ 不应包含在倒谱匹配条件式 (3.6.49b) 中。

应用 Lagrange 乘子法，构造代价函数

$$\begin{aligned}J[P(\omega)]=&\frac{1}{2\pi}\int_{-\pi}^{\pi}\ln P(\omega)\mathrm{d}\omega+\sum_{m=-M}^{M}\lambda_m\left[\hat{R}_x(m)-\frac{1}{2\pi}\int_{-\pi}^{\pi}P(\omega)\mathrm{e}^{\mathrm{j}\omega m}\mathrm{d}\omega\right]+\\&\sum_{l\neq0,l=-N}^{N}\mu_l\left[\hat{c}_x(l)-\frac{1}{2\pi}\int_{-\pi}^{\pi}\ln P(\omega)\mathrm{e}^{\mathrm{j}\omega l}\mathrm{d}\omega\right]\end{aligned} \tag{3.6.50}$$

式中 λ_m 和 μ_l 为两个待定的 Lagrange 乘子。令偏导 $\partial J[P(\omega)]/\partial P(\omega)=0$，并作代换 $\beta_l=-\mu_l$，即得到功率谱密度的表达式

$$P(\omega)=\frac{\displaystyle\sum_{l=-N}^{N}\beta_l\mathrm{e}^{\mathrm{j}\omega l}}{\displaystyle\sum_{m=-M}^{M}\lambda_m\mathrm{e}^{\mathrm{j}\omega m}} \tag{3.6.51}$$

式中 $\beta_0=1$。

由定理 3.6.1 即 Fejer-Riesz 定理知，式 (3.6.51) 可以写成

$$P(\omega)=\frac{|B(z)|^2}{|A(z)|^2}=\left.\frac{\left|\displaystyle\sum_{i=0}^{N}b_iz^{-i}\right|^2}{\left|1+\displaystyle\sum_{i=1}^{M}a_iz^{-i}\right|^2}\right|_{z=\mathrm{e}^{\mathrm{j}\omega}} \tag{3.6.52}$$

显然，这是一个有理式 ARMA 功率谱。

不难看出，自相关函数的约束条件导致了自回归部分，作用为极点；而倒谱的约束条件则贡献为滑动平均部分，作用为零点。

Lagunas 等人 [109] 介绍了最大熵 ARMA 谱估计的具体算法，这里从略。

上述最大熵法同时利用自相关函数匹配和倒谱匹配来获得最大熵 ARMA 功率谱，而 Burg 最大熵法只利用自相关函数匹配，所以它与 AR 功率谱等价。

在 Burg 定义的谱熵中，我们看到它与连续随机变量的熵的定义式 (3.6.12) 在形式上明显不同。如果严格仿照式 (3.6.12) 定义功率谱的熵，应为

$$H_2[P(\omega)] = -\frac{1}{2\pi}\int_{-\pi}^{\pi} P(\omega)\ln P(\omega)\mathrm{d}\omega \tag{3.6.53}$$

称之为构型熵，是 Frieden [67] 提出使用的。

习惯上，采用谱熵最大化的功率谱估计称为第一类最大熵方法 (简写为 MEM1)，采用构型熵 (它是负熵) 最小化的功率谱估计称为第二类最大熵方法 (简称 MEM2)。

应用 Lagrange 乘子法，构造代价函数

$$\begin{aligned} J[P(\omega)] = &-\frac{1}{2\pi}\int_{-\pi}^{\pi} P(\omega)\ln P(\omega)\mathrm{d}\omega + \\ &\sum_{m=-M}^{M}\lambda_m\left[\frac{1}{2\pi}\int_{-\pi}^{\pi}P(\omega)\mathrm{e}^{\mathrm{j}\omega m}\mathrm{d}\omega - \hat{R}_x(m)\right] \end{aligned} \tag{3.6.54}$$

令 $\dfrac{\partial J[P(\omega)]}{\partial P(\omega)} = 0$，则得到

$$\ln P(\omega) = -1 + \sum_{m=-M}^{M}\lambda_m \mathrm{e}^{\mathrm{j}\omega m} \tag{3.6.55}$$

式中 λ_m 为待定的 Lagrange 乘子。作变量代换 $c_m = \lambda_{-m}, m = \pm 1,\cdots,\pm M$ 及 $c_0 = -1 + \lambda_0$ 后，则式 (3.6.55) 可改写作

$$\ln P(\omega) = \sum_{m=-M}^{M} c_m \mathrm{e}^{-\mathrm{j}\omega m}$$

故最大熵功率谱密度为

$$P(\omega) = \exp\left(\sum_{m=-M}^{M} c_m \mathrm{e}^{-\mathrm{j}\omega m}\right) \tag{3.6.56}$$

显然，求 MEM2 功率谱密度的关键是估计系数 c_m。通常，将 c_m 称为复倒谱。