测距雷达解距离模糊的两种快速算法 [1]

许邦建　　李　纲　　皇甫堪

(国防科技大学电子科学与工程学院　长沙　410073)

摘　要　针对测距雷达中存在的实数域内的测距解模糊问题，该文认真分析了全局搜索算法的计算量，指出其计算量过大，难于适应实时测距的要求．在此基础上该文提出了两种适用于一般情况的且能在规定的评价函数下取得最优解的快速算法．理论分析及计算机仿真表明了这些快速算法的有效性．

关键词　测距雷达，信号处理，距离模糊，快速算法

中图号　TN951

1 引　言

无论是脉冲测距雷达还是连续波比相测距雷达，皆有测距模糊的问题 [1,2] 。对于脉冲测距雷达，在单一的脉冲高重复频率之下其测距不模糊距离是很有限的．为扩展其最大不模糊测距距离，我们可以采用多重脉冲重复周期的办法，即顺序发射多种脉冲重复周期 (依次为基本脉冲重复周期的 m_i 倍) 的脉冲信号。这样在 m_i 两两互素的情况下，最大不模糊测距距离可以扩展到基本脉冲重复频率下最大不模糊测距距离 R_0 的 $\prod_i m_i$ 倍 [1] 。

对于连续波比相测距雷达，其测距的最大不模糊距离与测距精度之间是一对矛盾 [1,2] 。为了能够既达到很高的测距精度，又有很大的测距不模糊距离，我们也要采用类似于脉冲雷达的多重脉冲重复周期的办法，即同时或顺序发射多个频差，其频差周期依次为基本频差周期的 m_i 倍．在 m_i 两两互素的情况下，同样地，最大不模糊测距距离可以扩展到基本频差下最大不模糊测距距离 R_0 的 $\prod_i m_i$ 倍 [2] 。

这样，我们就面临着一个如何根据各次测得的模糊距离解出一个在最大不模糊距离范围之内快速解出一个真实距离值的问题。

2 问题的数学模型

设所测目标的实际距离为 $R\cdot R_0$(R 为一系数)，在第 i 次测距时所得的距离为 $r_iR_0(0\le r_i<m_i)$ 。现在的问题可以写成如下的方程组

$$\left.\begin{aligned} R_1 &= k_1m_1+(r_1+\sigma_1)\\ &\vdots\\ R_N &= k_Nm_N+(r_N+\sigma_N)\end{aligned}\right\}\tag{1}$$

其中 $\sigma_i\cdot R_0$ 为第 i 次测距时的测距误差。此处为考虑问题简单起见，假定 m_i 两两互素。则此处 R 的最大不模糊值为 $\prod_{i=1}^{N} m_i$ 。

在不存在测距误差的情况下，我们可以利用微分方程求解法、孙子定理、递推算法等 [3-6] 来快速求解 (1) 式。但在一般情况尤其是在连续波比相测距的情况下，σ_i 的存在及其影响是必然的 (关于 σ_i 对测距解模糊的影响，将另文论述)，而且 $(r_i+\sigma_i)$ 一般为实数，因此此时已无法再利用这些方法。

[1] 2000-04-01 收到，2000-10-26 定稿

3 N 维全局搜索算法的计算量分析

为解 (1) 式，我们可以采用全局搜索 (即 N 维搜索) 的办法。即先确定一个评价函数，再令各 k_i 在其取值范围 $0 \sim k_{i\max}$ 内变化，以寻找使得该评价函数达到最小的各 k_i 的值。典型地，我们取该评价函数为 $f=\sum_{i=1}^{N}(\hat{R}-R_i)^2$，其中 $\hat{R}=\frac{1}{N}\sum_{i=1}^{N}R_i$， $R_i=k_im_i+r_i$，则本算法经优化后可以形式地写为以下的算法 1 。

算法 1 最小误差等于一个很大的值；

for $k_1=0$ to $k_{1\max}$

{ $R_1=k_1\times m_1+r_1$; $R_{\text{sum}}=0$; $R_{\text{sum}}=R_{\text{sum}}+R_1$;

for $k_2=0$ to $k_{2\max}$

{ $R_2=k_2\times m_2+r_2$; $R_{\text{sum}}=R_{\text{sum}}+R_2$;

for $k_3=0$ to $k_{3\max}$

⋮

for $k_N=0$ to $k_{N\max}$

{ $R_N=k_N\times m_N+r_N$; $R_{\text{sum}}=R_{\text{sum}}+R_N$;

$\hat{R}=\dfrac{R_{\text{sum}}}{N}$; $f=\sum\limits_{i=1}^{N}(\hat{R}-R_i)^2$;

if $f<$ 最小误差

{ 最小误差 $=f$，各k_i的解 $=k_i$; }

}

⋮

}

}

上述算法中 $k_{i\max}=\left(\prod_{\substack{j=1\\j\neq i}}^{N}m_j-1\right)$ 。算法 1 中，在最内层为计算 $\hat{R}$ ，需要做 1 次乘法；为计算 f ，需要做 N 次乘法、 $(2N-1)$ 次加法。另外对每一个 k_i 的循环，都要计算一次 R_i 及 R_{sum} ，从而需要一次乘法、两次加法。因此算法 1 中所需的总的乘法次数为

$$(N+1)\left(\prod_{i=1}^{N}(1+k_{i\max})\right)+\sum_{j=1}^{N}\left(\prod_{i=1}^{j}(1+k_{i\max})\right)$$

所需的总的加法次数为

$$(2N-1)\left(\prod_{i=1}^{N}(1+k_{i\max})\right)+2\sum_{j=1}^{N}\left(\prod_{i=1}^{j}(1+k_{i\max})\right)$$

以上并未考虑诸如数值大小的比较等所花的时间。

假若要求一个数字信号处理器 (DSP) 在 10ms 之内完成这些计算量并且认为 DSP 计算一次加法或一次乘法所需时间均为一个指令周期，则在 m_i 依次为 17,15,13,11 的情况下，容易知道 DSP 的指令周期应该为 1.4×10^{-17}s 。这一要求就连目前速度最快的 DSP 亦不能达到。可见，我们必须寻找解 (1) 式的快速算法。

4 基于 $(N-1)$ 维搜索的快速解模糊算法

注意到对应某一组确定的 $k_i(i=1,\cdots,N-1)$ ，使得 $\sum_{i=1}^{N}(\hat{R}-R_i)^2$ 达到最小的 k_N 是一定的，因此我们可以将原来的 N 维搜索问题转化为 $(N-1)$ 维搜索问题。这样也就减少了很多计算量。

具体来说是这样的：令 $\sum_{i=1}^{N}(\hat{R}-R_i)^2$ 对 k_N 的导数为 0 ，可知在实数范围内 k_N 的解为 $k'_N=(\sum_{i=1}^{N-1}R_i-(N-1)r_N)/((N-1)m_N)$; 另外显然 $\sum_{i=1}^{N}(\hat{R}-R_i)^2$ 是 k_N 的一元二次函数，再注意到 k_N 为整数，因此对应某一组确定的 $k_i(i=1,\cdots,N-1)$ ，由一元二次函数的性质，使得 $\sum_{i=1}^{N}(\hat{R}-R_i)^2$ 达到最小的 k_N 的整数解显然应该为离 k'_N 最近的整数。

本算法经优化后可以形式地写为以下的算法 2 .

算法 2　最小误差等于一个很大的值;

for $k_1=0$ to $k_{1\max}$

{　$R_1=k_1\times m_1+r_1$;　$R_{\text{sum}}=0$;　$R_{\text{sum}}=R_{\text{sum}}+R_1$;

　for $k_2=0$ to $k_{2\max}$

　{　$R_2=k_2\times m_2+r_2$;　$R_{\text{sum}}=R_{\text{sum}}+R_2$;

　　for $k_3=0$ to $k_{3\max}$

　　　⋮

　　　for $k_{N-1}=0$ to $k_{(N-1)\max}$

　　　{　$R_{N-1}=k_{N-1}\times m_{N-1}+r_{N-1}$;　$R_{\text{sum}}=R_{\text{sum}}+R_{N-1}$;

$$k_N=\left\|\frac{R_{\text{sum}}-(N-1)r_N}{(N-1)m_N}\right\|;\quad (\|*\|\text{表示离}*\text{最近的整数})$$

　　　　$R_N=k_N\times m_N+r_N$;　$R_{\text{sum}}=R_{\text{sum}}+R_N$;

$$\hat{R}=\frac{R_{\text{sum}}}{N};\quad f=\sum_{i=1}^{N}(\hat{R}-R_i)^2;$$

　　　　if $f<$ 最小误差

　　　　{　最小误差 $=f$;　各k_i的解 $=k_i$;}

　　　}

　　⋮

　}

}

类似于算法 1 的分析，我们容易知道，算法 2 中所需的总的乘法次数为

$$(N+3)\left(\prod_{i=1}^{N-1}(1+k_{i\max})\right)+\sum_{j=1}^{N-1}\left(\prod_{i=1}^{j}(1+k_{i\max})\right)$$

所需的总的加法次数为

$$(2N+2)\left(\prod_{i=1}^{N-1}(1+k_{i\max})\right)+2\sum_{j=1}^{N-1}\left(\prod_{i=1}^{j}(1+k_{i\max})\right)$$

因为本算法只是减少了一个维度的搜索，因此其计算量还是很大的。但是显然可以看到，在诸如 $N=2$ 的情况下，算法 2 可以将计算量减少近一半。

5 基于 1 维主搜索、 $(N-1)$ 维从搜索的快速解模糊算法

为介绍本算法，我们首先引入下面的引理。

引理 使得 $f=\sum_{i=1}^{N}(\hat{R}-R_i)^2$ 达到最小值的 (1) 式在最大不模糊距离之内的整数解若为 $(k_1^*,k_2^*,\cdots,k_N^*)$，则必有 $|R_i^*-R_j^*|<\sqrt{N}$(任意 i,j, 且 $i\neq j$) 成立。其中 $R_i^*=k_i^*\cdot m_i^*+r_i$。

证明 显然，如果我们只考虑由 r_i 的整数部分 $\lfloor r_i\rfloor$ (本文中 $\lfloor\bullet\rfloor$ 表示对 $\bullet$ 向 $-\infty$ 方向取整， $\lceil\bullet\rceil$ 表示对 $\bullet$ 向 $+\infty$ 方向取整) 构成的整数同余方程组：

$$\left.\begin{array}{c} Q_1=k_1\cdot m_1+\lfloor r_1\rfloor \\ \vdots \\ Q_N=k_N\cdot m_N+\lfloor r_N\rfloor \end{array}\right\} \tag{2}$$

则我们可以由孙子定理在最大不模糊距离范围之内确定一组整数系数解 $(k_{10},k_{20},\cdots,k_{N0})$(其中 $0\leq k_{i0}<k_{i\max}$)。把这些系数代进 $f=\sum_{i=1}^{N}(\hat{R}-R_i)^2$, 则此时 $\lfloor\hat{R}\rfloor=\lfloor R_i\rfloor$(其中 $R_i=k_{i0}m_i+r_i$)，因此 $f=\sum_{i=1}^{N}(<\hat{R}>-<R_i>)^2$(其中 $<x>$ 为 x 的小数部分)。而 $|<\hat{R}>-<R_i>|<1$，因此在点 $(k_{10},k_{20},\cdots,k_{N0})$ 处必然有 $|R_i-R_j|<\sqrt{N}$ 成立。既然如此，而且点 $(k_1^*,k_2^*,\cdots,k_N^*)$ 是 (1) 式的最优解，因此在 $(k_1^*,k_2^*,\cdots,k_N^*)$ 处必有 $|R_i^*-R_j^*|<\sqrt{N}$ 成立。 证毕

由本引理，在某一个 k_i(此处假定为 k_1) 已知的情况下，根据 $|R_j^*-R_1|<\sqrt{N}$，我们可以知道： $\lfloor(R_1-\sqrt{N}-r_j)/m_j\rfloor<k_j^*<\lceil(R_1+\sqrt{N}-r_j)/m_j\rceil$。从而可将 k_j 的范围大大缩小。这样，我们就可以将原来的 N 维搜索的问题转化为一个主要是 k_1 的一维搜索的问题。这就是本文提出的基于一维主搜索、 $(N-1)$ 维从搜索的快速解模糊算法。

本算法经优化后可以形式地写为以下的算法 3 。

算法 3 最小误差等于 N；

$$
\begin{array}{l}
\text{for } k_1 = 0 \text{ to } k_{1\max} \\
\{ \quad R_1 = k_1 \times m_1 + r_1;\; R_{\text{sum}} = 0;\; R_{\text{sum}} = R_{\text{sum}} + R_1; \\
\quad k_{2\text{low}} = \left\lceil \dfrac{R_1 - r_2 - \sqrt{N}}{m_2} \right\rceil;\quad k_{2\text{up}} = \left\lfloor \dfrac{R_1 - r_2 + \sqrt{N}}{m_2} \right\rfloor; \\
\quad \vdots \\
\quad k_{N\text{low}} = \left\lceil \dfrac{R_1 - r_N - \sqrt{N}}{m_N} \right\rceil; k_{N\text{up}} = \left\lfloor \dfrac{R_1 - r_N + \sqrt{N}}{m_N} \right\rfloor; \\
\quad \text{for } k_2 = \max\{0, k_{2\text{low}}\} \text{ to } \min\{k_{2\text{up}}, k_{2\max}\} \\
\quad \{ \quad R_2 = k_2 \times m_2 + r_2;\; R_{\text{sum}} = R_{\text{sum}} + R_2; \\
\qquad \text{for } k_3 = \max\{0, k_{3\text{low}}\} \text{ to } \min\{k_{3\text{up}}, k_{3\max}\} \\
\qquad \{ \quad R_3 = k_3 \times m_3 + r_3;\; R_{\text{sum}} = R_{\text{sum}} + R_3; \\
\qquad\quad \vdots \\
\qquad\quad \text{for } k_N \max\{0, k_{N\text{low}}\} \text{ to } \min\{k_{N\text{up}}, k_{M\max}\} \\
\qquad\quad \{ \quad R_N = k_N \times m_N + r_N; R_{\text{sum}} = R_{\text{sum}} + R_N; \\
\qquad\qquad \hat{R} = \dfrac{R_{\text{sum}}}{N};\quad f = \sum\limits_{i=1}^{N} (\hat{R} - R_i)^2; \\
\qquad\qquad \text{if } f < \text{最小误差} \\
\qquad\qquad \{ \quad \text{最小误差} = f;\quad \text{各}k_i\text{的解} = k_i; \} \\
\qquad\qquad\quad \} \\
\qquad\quad \vdots \\
\qquad \} \\
\quad \} \\
\}
\end{array}
$$

由 $[(R_1 - r_j - \sqrt{N})/m_j] < k_j^* < [(R_1 - r_i + \sqrt{N})/m_j]$ 可知，在算法 3 中 k_j 的取值个数一般为 $\left[1 + \lfloor \frac{2\sqrt{N}}{m_j} \rfloor\right]$．再经过与上节类似的分析，我们得出算法 3 中的乘法次数为

$$
(1 + k_{1\max}) \sum_{j=2}^{N} \prod_{i=2}^{j} \left(1 + \left\lfloor \frac{2 \cdot \sqrt{N}}{m_i} \right\rfloor \right) + (1 + k_{1\max})(2N - 1) + (1 + k_{1\max})(N + 1) \prod_{i=2}^{N} \left(1 + \left\lfloor \frac{2\sqrt{N}}{m_i} \right\rfloor \right)
$$

所需的总的加法次数为

$$2(1+k_{1\max})\sum_{j=2}^{N}\prod_{i=2}^{j}\left(1+\left\lfloor\frac{2\cdot\sqrt{N}}{m_i}\right\rfloor\right)+(1+k_{1\max})(4N-2)$$
$$+(1+k_{1\max})(2N-1)\prod_{i=2}^{N}\left(1+\left\lfloor\frac{2\cdot\sqrt{N}}{m_i}\right\rfloor\right)$$

6 三种算法计算量的理论数值及计算机仿真结果比较

首先，我们可以在一个假设条件下看一看各种算法在不同情况下对 DSP 指令周期的要求。类似于算法 I，同样假设要求一个 DSP 在 10ms 之内完成这些计算量并且认为 DSP 计算一次加法或一次乘法所需时间均为一个指令周期，则在表 1 中列出了根据各算法计算量的理论结果给出的对 DSP 指令周期的要求。

从表 1 可以看出：算法 3 的效率一般而言都是最高的且能满足实时的要求 (DSP 中较低档的 TMS320C31 指令周期为 60ns)，算法 2 在 N=2 的情况下效率也是很高的。

为了进一步验证以上结论，我们在普通的 PC 机上进行了算法的验证。该机主频为 400MHz。表 2 列出了在各种情况下这 3 种算法各自解一次模糊所需的时间。

表 1　各种算法在不同情况下对 DSP 指令周期的要求 (单位：s)

对应算法	$m_1=17$, $m_2=15$, $m_3=13$, $m_4=11$	$m_1=17$, $m_2=15$, $m_3=13$	$m_1=17$, $m_2=15$
算法 1	1.4×10^{-17}	7.6×10^{-11}	4.3×10^{-6}
算法 2	3.4×10^{-14}	5.3×10^{-11}	4.8×10^{-5}
算法 3	7.8×10^{-8}	1.3×10^{-6}	3.2×10^{-5}

表 2　各种算法在不同情况下对 DSP 指令周期的要求 (单位：s)

对应算法	$m_1=17$, $m_2=15$, $m_3=13$, $m_4=11$	$m_1=17$, $m_2=15$, $m_3=13$	$m_1=17$, $m_2=15$
算法 1	4×10^{6}	3	1×10^{-6}
算法 2	200	0.05	$<1\times10^{-6}$
算法 3	$<1\times10^{-6}$	$<1\times10^{-6}$	$<1\times10^{-6}$

参 考 文 献

[1] (美)M.I.Skolnik, 林茂庸等 (译), 雷达系统导论, 北京：国防工业出版社, 1992 年 2 月第 1 版, 45–46, 58–84.

[2] 沈福民, 贾永康, 相位测距中的解模糊技术, 西安电子科技大学学报, 1997, 24(1)

[3] 孙洪, 姚天任, 同余方程组的微分方程求解法, 应用数学, 1996 增刊, 158–161.

[4] 朱宏勋, 适合用计算机解雷达模糊的几种算法, 现代雷达, 1986(4), 93–97.

[5] S. A. Hovanessian, An algorithm for calculation of range in a multiple PRF radar, IEEE Trans. on AES 1976, 12(3), 287–290.

[6] 黄振兴，史燕，用孙子定理解雷达测距模糊性，成都电讯工程学院学报， 1986, 15(3), 47–51.

TWO FAST ALGORITHMS OF DE-AMBIGUOUS TECHNIQUE IN RANGING RADAR

Xu Bangjian　　Li Gang　　Huang Fukan

(*School of Electronic Science and Engineering, NUDT, Changsha* 410073, China)

Abstract In ranging radar, range ambiguity usually exists. Computing amount of overall search algorithm to solve the problem is analyzed, and it is pointed out that this algorithm is unsuitable for ranging in real time. Then two fast algorithms that can attain the true solution of original equation set under certain evaluation function are put forward. These algorithms' effectiveness is proved by theory analyses and computer simulation.

Key words Ranging radar, Signal processing, Range ambiguity, Fast algorithm

许邦建：　男，1974 年生，博士生，从事雷达系统和信号处理、神经网络等方面的研究.
李　纲：　男，1966 年生，讲师，主要从事雷达和通信信号处理及其实现技术研究.
皇甫堪：　男，1939 年生，教授，博士生导师，主要从事现代信号处理技术方面的研究.