易懂的雷达信号处理

面向学生与工程师

第六章：目标检测

作者：唐承乾

版本：社区版 V1

官方仓库：https://github.com/apple-art/easy-radar-tutorial

目录

前两章我们已经会从回波里读出距离和速度了。到这里，很多读者会自然产生一个误会：图上只要有个峰，那不就说明有目标吗？第 6 章就是专门来打破这个误会的。

因为真实雷达数据里，峰值从来不只可能来自目标。噪声会抬出小尖峰，杂波会形成大片背景起伏，旁瓣也会制造看起来“像目标”的结构。所以这一章要解决的，不是“再算一个新物理量”，而是一个非常现实的判断问题：眼前这个峰，到底该不该信？

按这个目标来看，本章的结构其实很清楚：

核心内容（必须掌握）：

- 6.1：为什么“有峰”不等于“有目标”，检测问题的本质到底是什么
- 6.2：最简单的检测为什么就是设一个阈值，阈值高低又会带来什么后果
- 6.3：虚警、漏检、检测概率这些量各自在说什么，它们之间为什么总要权衡
- 6.4：CFAR 为什么会出现，它怎样把固定阈值变成随背景变化的本地阈值

巩固内容：

- 6.5：小练习，用来把“阈值比较”和“虚警/漏检权衡”真正算一遍、想一遍

6.1 目标检测的基本问题

前面两章我们已经会做两件事：第 4 章把回波里的**时间延迟**变成了距离，第 5 章把回波里的**频率偏移**变成了速度。看起来似乎已经足够：只要在距离像或者速度谱里找到一个尖峰，不就知道那里有目标了吗？

但真正做起来，很快就会遇到一个问题：图上永远不会只有干干净净的“目标峰”，还会有噪声、杂波、旁瓣，以及各种偶然起伏。如果看到一个凸起就认定有目标，误报会多得无法使用。

所以，目标检测这一章要解决的问题是：眼前这个峰，到底值不值得相信？

6.1.1 检测判决的直观例子

假设匹配滤波之后，我们得到一条距离像。某 8 个距离单元的幅度如下：

$$0.8, 1.1, 0.9, 1.0, 4.7, 1.2, 1.0, 0.7$$

这时候大多数人都会立刻指向第 5 个单元：4.7 这么高，显然像目标。这个判断通常是对的，因为它比周围背景高出很多。但要注意，这里其实已经隐含了检测判断。你并不是只在“看见数字”，而是在比较：背景大概在 1 左右，第 5 个单元明显更大，所以它更像真实目标，而不是噪声波动。

现在把数据稍微换一下：

$$0.8, 1.1, 0.9, 1.0, 1.6, 1.2, 1.0, 0.7$$

第 5 个单元现在只有 1.6。它还算不算目标？这时就没那么笃定了。因为 1.6 虽然比周围大

一点，但也可能只是噪声恰好抬高了一次。目标检测真正麻烦的地方，就出现在这种“不够明显但又不能轻易忽略”的情形里。

这也说明，仅仅“看峰值”还不够。你面对的不是一个抽象的数学问题，而是在一堆起伏里分辨：哪个凸起值得相信，哪个只是背景偶然冒高了一下。

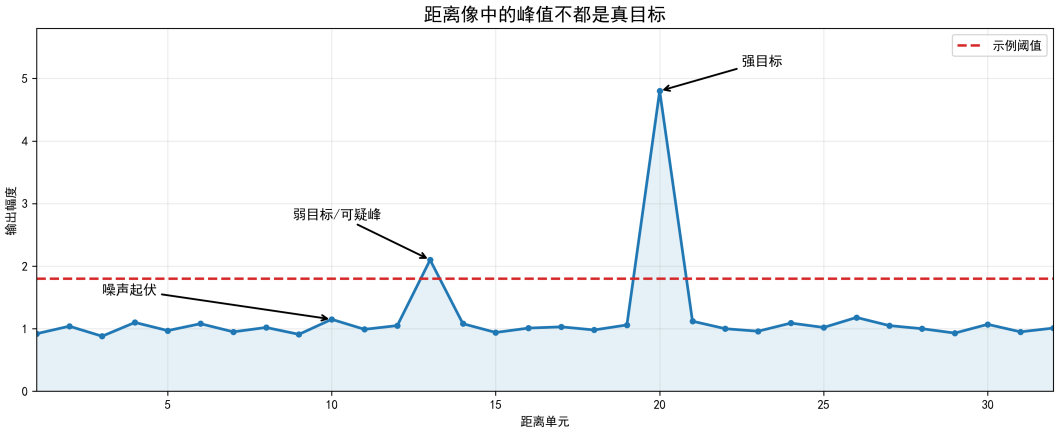

图 1: 距离像中的强目标、弱目标和噪声起伏

6.1.2 检测的两种基本假设

在检测问题里，最理想的情况当然是：有目标时，回波很强；没目标时，输出几乎是零。但实际系统很难做到这么干净。接收机本身有热噪声，环境里有杂波，前面的脉冲压缩和频谱分析还可能带来旁瓣和泄漏。于是即使“没有目标”，距离单元或频率单元里也常常会出现非零输出，而且这些输出还会上下起伏。

所以在检测里，通常把一个单元里的输出看成两种成分的叠加：

$$\text{输出} = \text{目标回波} + \text{噪声/杂波}$$

如果这个单元里没有目标，那就退化成：

$$\text{输出} = \text{噪声/杂波}$$

更直接地说，检测问题最基本的判断只有两个：要么这里只有噪声，要么这里是“目标加噪声”。很多教材会立刻把它写成假设检验：

$$H_0 : x = n$$

$$H_1 : x = s + n$$

这里的 x 是观测值，n 是噪声或背景杂波，s 是目标信号；H_0 表示“没有目标”，H_1 表示“有目标”。这个写法看上去较正式，但表达的意思很直接：眼前这个数值，到底只是背景在波动，还是背景上真的叠加了一个目标？

顺着这个想法再往前走一步，也就能看出为什么“只找局部最大值”不够。噪声也会制造局部最大值。比如：

$$0.7, 1.0, 1.4, 1.1, 0.9$$

第 3 个单元是局部最大值，但它未必是真目标。它也可能只是随机起伏里恰好最高的一点。换句话说，“比左右两边高”只能说明它是个峰，不能说明它是个可信的目标峰。真正需要回答的是另一个问题：它高到什么程度，才足以让人相信它不是偶然起伏？

6.1.3 阈值检测的基本规则

为了把问题讲得最简单，先只考虑一个单元的幅度值 x。我们人为设定一个门槛，记作 T：如果 $x > T$，判定“有目标”；如果 $x \le T$，判定“没有目标”。写成公式就是：

$$\begin{cases} \text{判有目标}, & x > T \\ \text{判无目标}, & x \le T \end{cases}$$

这个规则非常朴素，却是绝大多数检测器的共同骨架。后面无论是固定阈值、CFAR，还是更复杂的统计检测方法，本质上都绕不开这一步：把一个连续变化的数值，变成“报”还是“不报”的离散判决。

这里最值得注意的，不是公式本身，而是它背后的代价。阈值设低了，容易把噪声当成目标；阈值设高了，容易把弱目标漏掉。也就是说，检测从来不是绝对正确的识别，而是带风险的判决。这正是本章的主线。

如果回头看前面的内容，就会发现本章其实是在为第 4 章和第 5 章的结果加上一道判别环节。第 4 章告诉我们某个峰出现在什么距离，第 5 章告诉我们某个峰对应什么速度，第 6 章则要回答：这个峰到底要不要信？如果没有这一步，前面的距离和速度估计就缺少可信性判断：虽然看到了许多峰值，却无法判断哪些对应真实目标，哪些只是背景起伏。

6.2 检测阈值与固定阈值检测

上一节已经把检测问题压缩成一句很简单的话：某个单元的输出值，超过多大才算目标？这个“多大”，就是阈值。

阈值看上去只是一个数，但它其实决定了检测器的取向：是宁可多报也不放过，还是宁可保守也不轻易报警。

6.2.1 固定阈值检测

假设某个雷达系统在一段时间内背景比较稳定。我们观察到，没有目标时，距离单元的幅度大多在 0.6 到 1.4 之间波动。于是可以把阈值设成 2.0，只要超过 2.0，就认定有目标。这就是最简单的**固定阈值检测**。

它的规则可以写成：

$$\begin{cases} \text{判有目标}, & x > T \\ \text{判无目标}, & x \le T \end{cases}$$

其中 x 是当前单元的观测值，T 是事先设定好的固定阈值。如果第一次接触检测问题，这个方法其实很自然。因为它直接回答了一个朴素问题：背景通常在 1 左右，那么 1.1、1.2 这些起伏大概率只是噪声；但如果某个单元突然冲到 4、5 甚至更高，就更像目标。固定阈值的价值，就在于它先给了我们一个最基础的判断标尺。

6.2.2 固定阈值下的判决示例

先不看更复杂的统计公式，只用最简单的数字体会阈值的作用。假设阈值取 $T = 2.0$。

某单元输出 $x = 4.7$，因为

$$4.7 > 2.0$$

所以判为有目标。这个结论通常较可靠，因为它比背景高出太多了。

某单元输出 $x = 1.1$，因为

$$1.1 < 2.0$$

所以判为无目标。这也没有什么争议，因为它落在背景正常波动范围内。

麻烦往往出在边缘区域。比如某单元输出 $x = 1.9$，因为

$$1.9 < 2.0$$

按规则仍然判为无目标。但这时就会犹豫：1.9 已经不算小了，会不会是一个较弱的真实目标？目标检测最难的不是 4.7 这种特别明显的峰，而是 1.8、1.9、2.1 这种“差一点”的情况。

现在再把阈值从 2.0 降到 1.3，看一组没有目标时的数据：

$$0.9, 1.1, 1.4, 1.0, 1.2, 1.5, 0.8$$

如果阈值是 2.0，这组数据一个都不会报警；但如果阈值降到 1.3，那么 1.4 和 1.5 都会报警。问题是，这里明明假设“没有目标”。也就是说，报警完全是由噪声起伏造成的。阈值一旦设得过低，系统就会变得过于敏感。

反过来，如果把阈值从 2.0 提高到 3.5，假设有一个较弱目标，它所在单元的输出是 2.8。若阈值为 2.0，则

$$2.8 > 2.0$$

目标能够被检出；但若阈值改成 3.5，则

$$2.8 < 3.5$$

系统会判成“没有目标”，真实目标就被漏掉了。

所以阈值高低带来的后果很清楚：阈值低，容易检出弱目标，但也容易误报；阈值高，误报减少，但弱目标更容易漏掉。这个矛盾不是后面某种高级算法才有，而是从最基础的阈值比较开始就已经存在了。

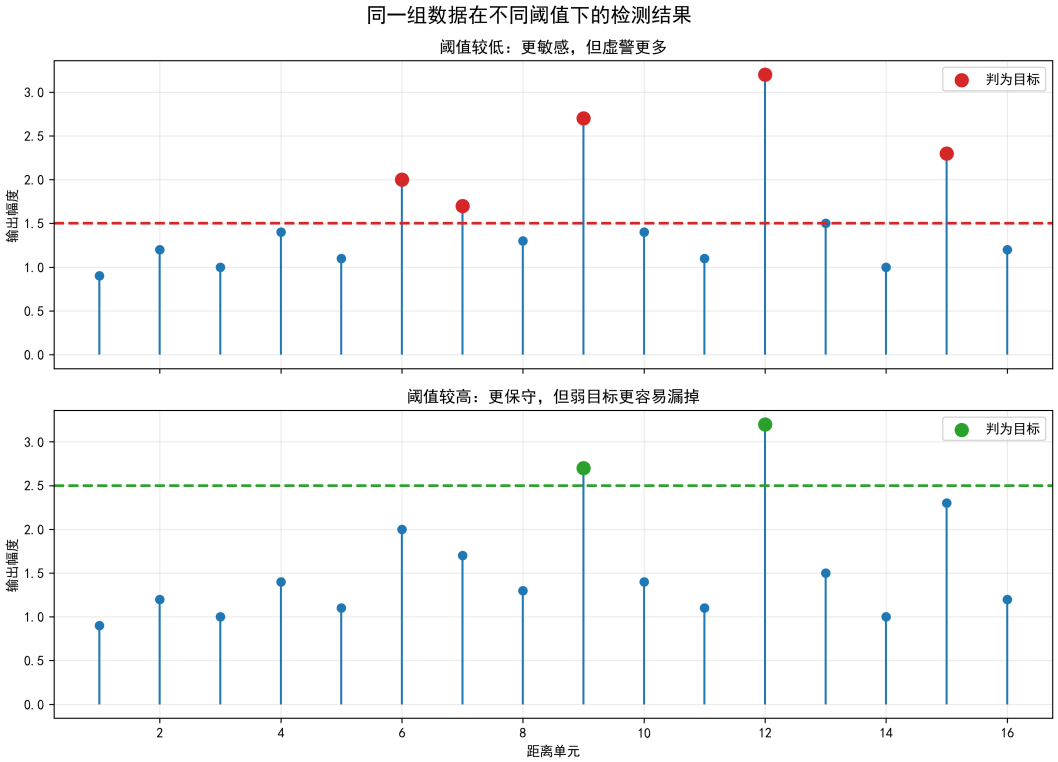

图 2: 同一组数据在不同阈值下的检测结果

6.2.3 阈值判决的数学形式

前面一直在用一个数 x 和一个门槛 T 做比较。现在可以稍作形式化，把这件事写成一个“判决器”：

$$\delta(x) = \begin{cases} 1, & x > T \\ 0, & x \le T \end{cases}$$

这里的 $\delta(x)$ 只是一个记号，表示检测器最后的输出：$\delta(x) = 1$ 表示“报目标”，$\delta(x) = 0$ 表示“不报目标”。这个式子本身并不复杂，它只是把前面做过的算术比较压缩成了一个统一写法。

这样写有两个好处。第一，后面讨论虚警率、检测概率时，更方便统一表述；第二，当阈值不再是固定常数，而是像 CFAR 那样随背景变化时，这个形式仍然能直接用。也就是说，复杂公式往往只是把前面反复做过的简单动作，写成一个更适合推广的形式。

6.2.4 背景对阈值设计的影响

到这里你可能会继续追问：既然阈值这么重要，那实际工程里它到底依据什么来定？最粗略的想法，是参考背景噪声的典型大小。例如，背景平均在 1 左右，就把阈值放在 2 或 3 附近；背景平均在 5 左右，就不能还用 2 这样的阈值。

这已经说明一个关键问题：阈值不能脱离背景单独讨论。若背景一直稳定，固定阈值还有可能好用；可一旦背景随距离、方位或场景发生变化，固定阈值就会开始失灵。

想象你在看一幅距离-速度图。某个区域本来很干净，背景亮度大多只有 1 左右；另一个区域因为地杂波或海杂波，背景亮度可能一直在 4 到 6 之间起伏。如果你用同一个固定阈值 $T = 3$，那么在干净区域，3 已经很高，虚警很少；可在杂波区域，3 反而比背景还低，会触发大量误报。

这就是固定阈值最大的弱点：它默认整张图的背景都差不多，而真实世界往往不是这样。不过，在背景比较均匀、教学入门、或先建立基础直觉时，固定阈值依然是最好的起点。因为所有更复杂的方法，都是围绕这个基础判断规则改出来的。

6.3 虚警、漏检与检测性能

上一节已经看到，阈值不可能同时取低又取高。阈值降低时，弱目标更容易被检出，但噪声也更容易越过阈值，误报随之增加。工程上不能只凭“感觉差不多”来设计检测器，还需要用明确指标描述检测性能：误报出现得多频繁，漏掉目标的概率有多大。

这就引出三个核心量：虚警率、漏检率和检测概率。

6.3.1 虚警率、漏检率与检测概率

在实际上没有目标时，检测器却报出了“有目标”，这叫**虚警**。在大量无目标场景中，虚警出现的比例称为虚警率，通常记作 P_{FA}。

在实际上有目标时，检测器却没有报出来，这叫**漏检**，表示真实目标被漏掉的概率，通常记作 P_M。与之对应，在实际上有目标时，系统成功报出“有目标”，这叫正确检测，它的概率记作 P_D。漏检和检测是同一件事的两面，因此有：

$$P_D = 1 - P_M$$

这个公式并不复杂，它只是说明：有目标时，要么检出，要么漏检，两种情况加起来就是全部。

先看一个简单的统计例子。假设我们做两组测试：第一组没有目标，让雷达在“无目标”条件下工作 100 次，结果有 8 次报警，那么

$$P_{FA} = \frac{8}{100} = 0.08$$

即虚警率为 8%。第二组确实有目标，再让雷达在“有目标”条件下工作 100 次，结果有 83 次成功报出目标，17 次没有报出，那么

$$P_D = \frac{83}{100} = 0.83$$

$$P_M = \frac{17}{100} = 0.17$$

这组结果说明，检测性能不能只说“效果不错”，而要明确给出这些统计量。

6.3.2 虚警与漏检的权衡

假设背景噪声的幅度大多分布在 0 到 2 之间，弱目标叠加噪声后的输出大多分布在 1 到 3 之间。可以看到，这两个范围是重叠的：一些较大的噪声样本会到 1.8、1.9、2.1，一些较弱的目标样本也可能只有 1.5、1.7、2.0。

既然两类样本混在一起，就不存在一条完美分界线，能把它们完全分开。如果阈值设在 1.2，多数目标都能检出，但很多纯噪声样本也会越线；如果阈值设在 2.6，虚警会减少，但很多弱目标也会被挡在阈值之下。

也就是说，只要噪声分布和目标分布有重叠，检测错误就不可避免。工程上能做的，不是消灭所有错误，而是在两类错误之间作取舍。

这时阈值就像一个旋钮。阈值升高，P_{FA} 会下降，P_D 往往也会下降；阈值降低，P_D 会上升，P_{FA} 往往也会升高。换句话说，提高可靠性，通常要牺牲灵敏度；提高灵敏度，通常要牺牲可靠性。

看一个更直观的表：

阈值	检测概率 P_D	虚警率 P_{FA}
1.2	0.97	0.22
1.8	0.88	0.08
2.5	0.65	0.01

从表中可以看到，阈值取 1.2 时，系统对目标更敏感，但误报也较多；阈值取 2.5 时，误报减少，但不少弱目标会漏掉；阈值取 1.8 时，则处于一个折中位置。这类表格在工程中很常见，因为它能直接展示这种取舍关系。

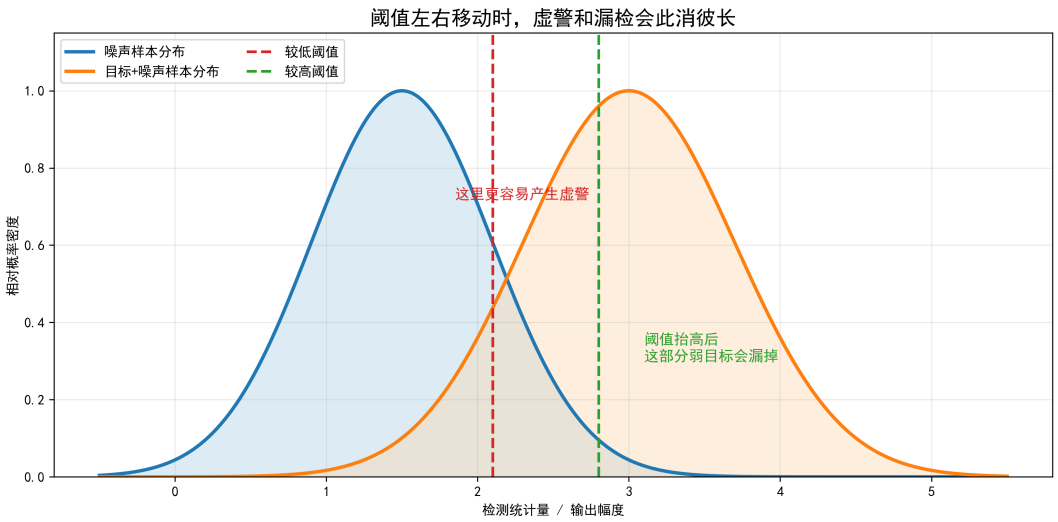

图 3: 噪声样本和目标样本重叠时的阈值权衡

如果希望更整体地看这种取舍，还可以把每个阈值对应的一组 (P_{FA}, P_D) 点连成一条曲线。图中的趋势很清楚：想把检测概率继续提高，通常就要接受更高的虚警率；想把虚警率压得更低，检测概率往往也会下降。

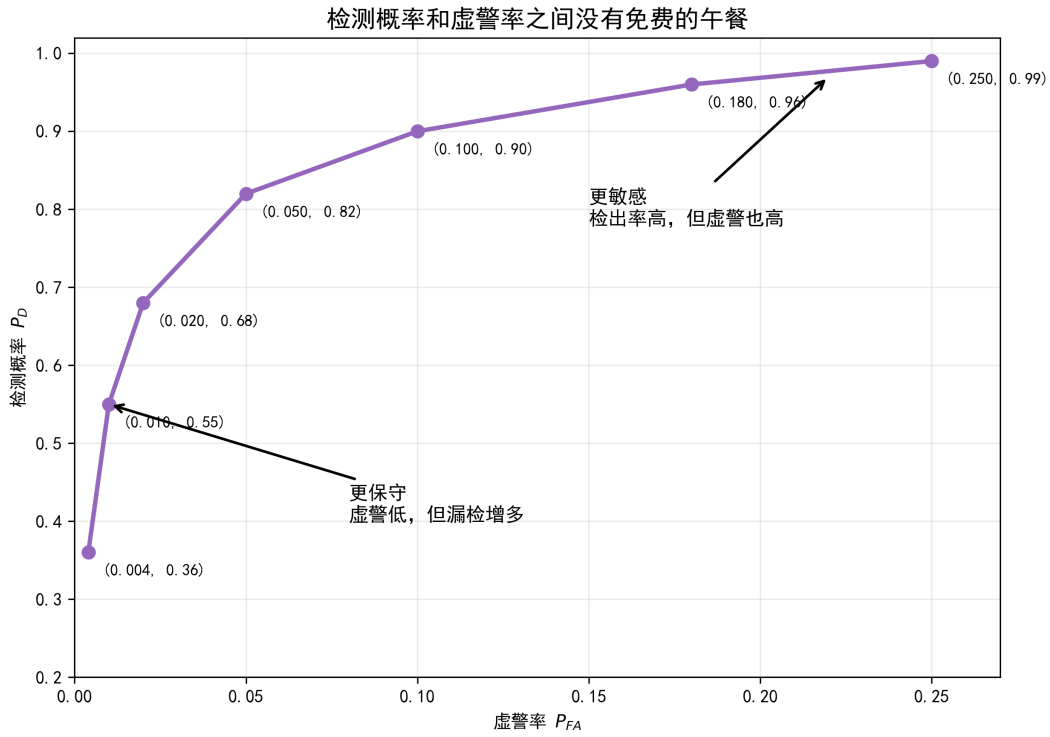

图 4: 检测概率和虚警率之间的权衡曲线

6.3.3 虚警率约束与阈值设计

对雷达系统来说，一个常见的问题不是偶尔漏掉很弱的小目标，而是系统持续误报，导致后端跟踪、显示甚至人工判读都被噪声淹没。所以在很多工程场景中，往往先规定一个可接受的虚

警水平。例如，每 1000 个单元平均只允许 1 个虚警，或者每一帧图像中的虚警数量不能多到让后端失控。有了这个要求，再去倒推阈值应当设在什么位置。

这也是“恒虚警率”思想的来源：不是先凭经验设一个阈值，而是先规定系统能够承受多大的虚警，再让阈值去满足这个要求。第四节的 CFAR，就是沿着这条思路发展出来的。

这里还可以顺便引入一个量：**信噪比**（Signal-to-Noise Ratio, SNR）。如果目标回波很强、背景噪声很弱，那么目标样本和噪声样本更容易区分，检测也更容易；反过来，如果目标和噪声差不多大，判决就会困难得多。最简单的功率型写法是：

$$\mathrm{SNR} = \frac{P_s}{P_n}$$

其中 P_s 是目标信号功率，P_n 是噪声功率。这个式子本身并不直接做检测，但它能帮助理解检测难度：SNR 高时，目标分布和噪声分布更容易分开，P_D 往往更高；SNR 低时，两者更容易重叠，需要更细致的阈值控制。也就是说，阈值负责“判”，而 SNR 决定这个判决有多难。

到这里，应当形成这样一个认识：检测一定会犯错；错误主要分成虚警和漏检两类；阈值是调节这两类错误比例的旋钮；背景越复杂、目标越弱，这个旋钮就越难用一个固定值调好。这最后一点尤其重要，因为前面的许多例子都默认背景比较稳定。可在真实雷达图上，不同区域的噪声底和杂波水平经常变化，这时同一个固定阈值在这里合适，到了另一块区域就可能完全不合适。这正是第四节要解决的问题。

6.4 恒虚警率检测（CFAR）

前面两节讨论了阈值，但都默认了一个隐藏前提：背景大致稳定。只有在这个前提下，固定阈值才像一把通用尺子，放到哪里都差不多能用。可真实雷达环境往往不是这样：近距离可能杂波更强，远距离噪声底可能更低，某些方位上有地物、海浪、雨区，背景会明显抬高，距离-速度图不同区域的起伏水平也可能完全不同。

这时，固定阈值容易出现的问题是：在安静区域偏高，在嘈杂区域又偏低。CFAR 的出发点，就是把阈值从写死的常数变成根据周围背景自动调整的量。

6.4.1 固定阈值的局限

假设你在一条距离像上考察两个位置。位置 A 的背景很干净，附近单元大致是：

$$0.8, 1.0, 0.9, 1.1, 2.6, 1.0, 0.8$$

这里如果用固定阈值 $T = 2.0$，2.6 很容易被检出来。

但另一个位置 B 的背景明显抬高了：

$$4.2, 4.8, 5.1, 4.5, 5.8, 4.9, 4.4$$

如果仍然用 $T = 2.0$，几乎整片区域都会报警。问题不在于位置 B 里真的全是目标，而在于那里的背景本来就高。同一个阈值脱离本地背景去使用，就会出错。反过来，如果为了适应位置 B，把阈值抬到 6.0，那么位置 A 里很多真实但较弱的目标又会被漏掉。

这正说明，阈值不能只跟系统总体情况比较，还要跟当前位置附近的背景比较。因此，一个自然的想法是：与其拿当前单元去和一个固定常数比较，不如先看它周围的背景大概有多高，再按那个背景水平去设阈值。这就是 CFAR 的核心思想。

名字里的 Constant False Alarm Rate，直译是“恒虚警率”。它让系统在不同背景下都尽量维持接近的虚警水平——阈值跟着局部背景变化，每个位置根据周围情况自动调整。

6.4.2 CFAR 的基本思想

假设我们要判断中间这个单元是否为目标。把它称为**被检测单元**，也常写成 CUT（Cell Under Test）。考虑下面一串数据：

$$1.0, 1.1, 0.9, \boxed{3.2}, 1.2, 1.0, 1.1$$

中间的 3.2 是 CUT。直观上它像目标，因为周围都在 1 左右。最朴素的做法是先估计周围背景平均值：

$$\frac{1.0 + 1.1 + 0.9 + 1.2 + 1.0 + 1.1}{6} = 1.05$$

然后把阈值设成背景平均值的若干倍。比如取 3 倍：

$$T = 3 \times 1.05 = 3.15$$

此时 CUT 为 3.2，因为

$$3.2 > 3.15$$

于是判为目标。这个过程已经很像 CFAR 了：先看周围背景有多高，再根据背景计算阈值，最后看当前单元是否超过这个阈值。虽然公式还很简单，但思路已经变了：阈值不再是预先写死，而是现场算出来的。

不过这里会遇到一个细节：不能把紧邻 CUT 的单元直接拿来算背景。原因是，真实目标通常不只影响一个单元，它附近可能还有主瓣边缘、旁瓣泄漏。如果把离 CUT 很近的单元也直接拿来平均，背景估计就会被目标“污染”。

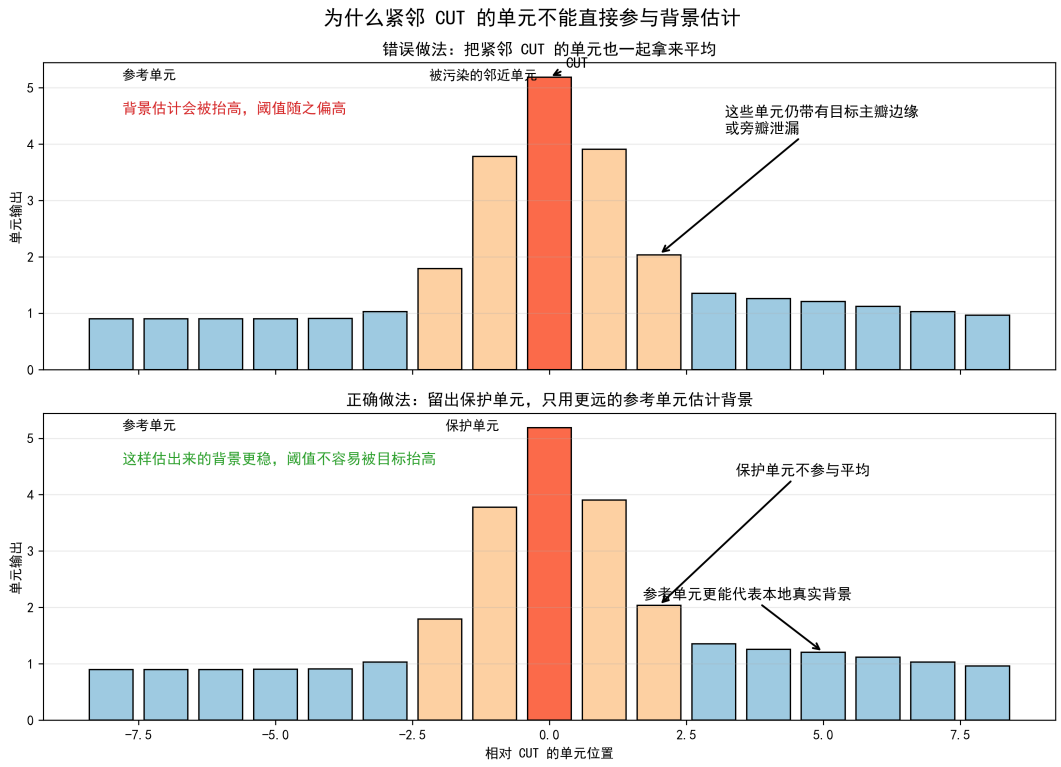

图 5: 为什么紧邻 CUT 的单元不能直接参与背景估计

所以 CFAR 通常会在 CUT 两边各留出一小圈**保护单元**（Guard Cells），不把它们用于背景估计；更远一点的单元才作为**参考单元**（Reference Cells）来估计背景。你可以把结构想成这样：

$$\text{参考 参考 保护 } \boxed{\text{CUT}} \text{ 保护 参考 参考}$$

这样做的目的很明确：保护单元是为了防止目标能量泄漏到背景估计里，参考单元则尽量代表“这一带本来有多吵”。这是 CFAR 能否工作好的关键细节之一。

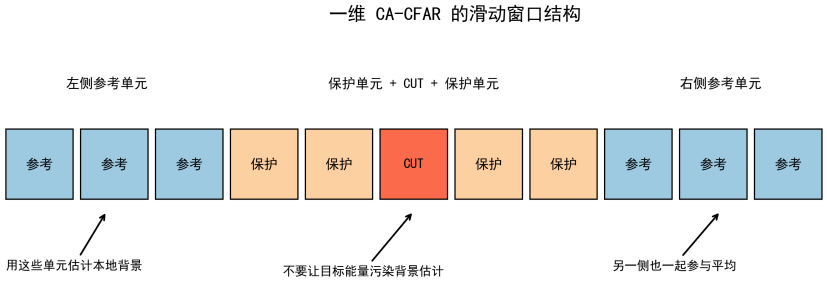

图 6: 一维 CA-CFAR 的参考单元、保护单元和 CUT 结构

6.4.3 CA-CFAR 的基本流程

最基础、最容易理解的一类 CFAR 叫 CA-CFAR，其中 CA 是 Cell Averaging 的缩写，意思是“单元平均”。它的流程可以概括成四步：选定一个 CUT；在 CUT 两侧留出保护单元；用更

外侧的参考单元求平均，估计局部背景功率或幅度；再把这个平均值乘上一个系数，得到阈值，然后和 CUT 比较。

如果用功率形式来写，一个常见表达是：

$$T = \alpha \hat{P}_n$$

其中 $\hat{P}_n$ 是由参考单元估计出来的局部噪声功率，α 是阈值系数，用来控制虚警率。然后比较：

$$\begin{cases} \text{判有目标}, & P_{CUT} > T \\ \text{判无目标}, & P_{CUT} \le T \end{cases}$$

这里的 P_{CUT} 是被检测单元的功率。

这里最容易让初学者疑惑的是：为什么还要再乘一个 α？直接拿背景平均值当阈值不行吗？一般不行。因为如果阈值只等于背景平均值，那么背景中稍大一些的自然起伏就很容易越线，虚警会很多。所以要乘一个大于 1 的系数，也就是在本地背景水平之上再留出安全余量。

例如，若背景均值约为 2，取 $\alpha = 3$，阈值就是 6；若背景均值约为 5，取 $\alpha = 3$，阈值就是 15。同样是 3 倍规则，在两个区域的阈值会自动变成不同数值。这正是 CFAR 的灵活之处。至于 α 和目标虚警率之间如何精确对应，严格推导需要概率分布知识，本书在入门阶段不展开，只保留最重要的工程理解：α 越大，阈值越高，虚警越少，但弱目标越难检出。也就是说，CFAR 并没有消除前一节的权衡，它只是把这种权衡改成了局部自适应地进行。

6.4.4 CFAR 的优势与局限

CFAR 最大的优点，是它能适应背景变化。固定阈值面对不同区域时常常顾此失彼，而 CFAR 的阈值能跟着背景变化，系统整体更稳定。比如背景随距离、方位、时间缓慢变化时，阈值能自动调整；整幅图噪声底不均匀时，虚警率也更容易控制；和固定阈值相比，它明显更适合真实雷达场景。

但它也不是万能的。如果参考单元里混进了别的强目标，背景估计会被抬高，导致漏检；如果处在杂波边缘，一侧背景高、一侧背景低，简单平均可能不准确；如果目标特别密集，参考单元很难找到“纯背景”区域。这也是为什么工程上后来发展出了 GO-CFAR、SO-CFAR、OS-CFAR 等变种，用来应对边缘杂波、多目标遮挡等问题。

把固定阈值和 CFAR 放到同一条背景不均匀的距离像上对比，这种差别会更直观。固定阈值前面可能偏高，后面又偏低；CFAR 则会随着局部背景抬高或降低阈值。

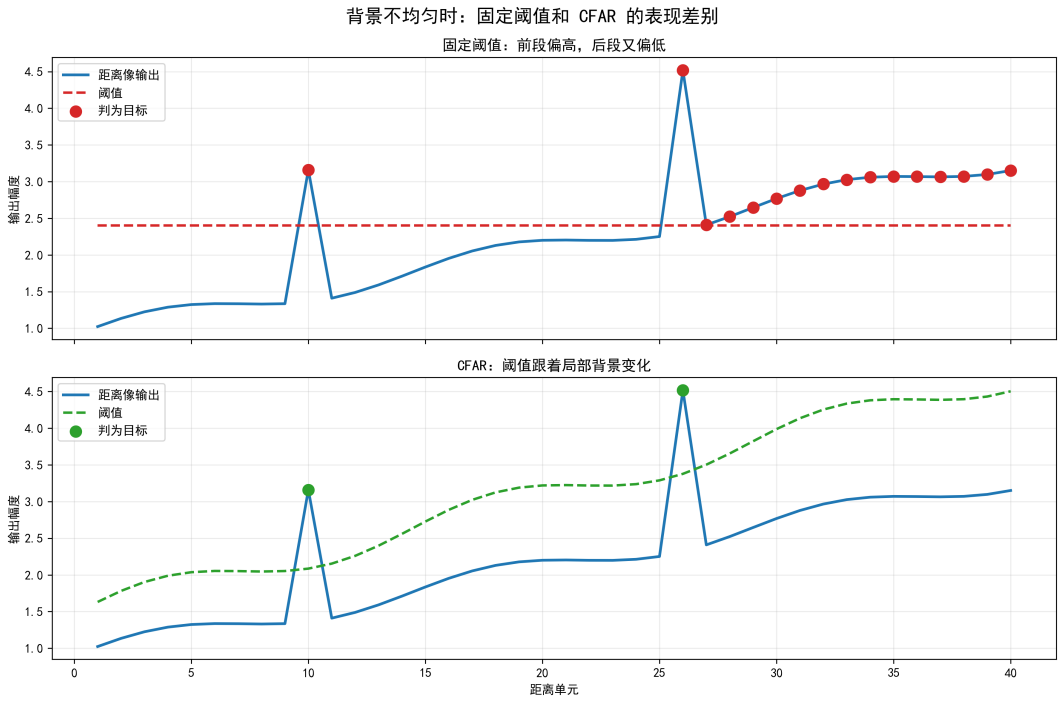

图 7: 固定阈值与 CFAR 在不均匀背景下的对比

6.5 小练习

这些练习围绕本章的主线展开：先理解固定阈值的基本判决，再理解虚警率、漏检率以及 CFAR 为什么要根据局部背景自适应调整阈值。

6.5.1 练习 1：固定阈值判决

问题：某距离单元输出幅度为 $x = 2.7$，固定阈值取 $T = 2.0$。该单元应判为有目标还是无目标?

解析：判决规则就是比较当前单元是否超过阈值。因为

$$2.7 > 2.0$$

所以该单元判为有目标。这道题虽然简单，但它说明了所有检测器最后都会落到“是否超阈值”这一步。

6.5.2 练习 2：阈值升高后的后果

问题：某弱目标所在单元输出为 $x = 2.4$。当阈值从 $T = 2.0$ 提高到 $T = 2.8$ 时，检测结果会怎样变化?

解析：当 $T = 2.0$ 时，

$$2.4 > 2.0$$

因此判为有目标。把阈值提高到 $T = 2.8$ 后，有

$$2.4 < 2.8$$

结果变成判为无目标。也就是说，阈值升高可以减少虚警，但也会使弱目标更容易漏掉。

6.5.3 练习 3：虚警率、检测概率和漏检率

问题：某检测器在 200 次无目标测试中误报了 6 次；在 150 次有目标测试中成功检测到 129 次。求：

1. 虚警率 P_{FA}
2. 检测概率 P_D
3. 漏检率 P_M

解析：虚警率为

$$P_{FA} = \frac{6}{200} = 0.03$$

即 3%。检测概率为

$$P_D = \frac{129}{150} = 0.86$$

即 86%。漏检率可以直接由成功检测次数反推，也可以由

$$P_M = 1 - P_D = 1 - 0.86 = 0.14$$

得到，因此漏检率为 14%。

6.5.4 练习 4：背景变化下，固定阈值为什么会出问题

问题：在同一幅距离像中，区域 A 的背景大致在 1 左右波动，区域 B 的背景大致在 5 左右波动。若系统统一使用固定阈值 $T = 3$，请分析它在两个区域的表现。

解析：在区域 A 中，背景只有 1 左右，阈值 3 已经比较高，因此虚警不会太多，只有明显高出的峰值才会触发。可是在区域 B 中，背景本来就在 5 左右，阈值 3 反而低于很多背景起伏，于是会出现大量误报。也就是说，固定阈值默认整个场景背景差不多，一旦背景明显不均匀，它就会在不同区域表现失衡。这正是 CFAR 要解决的问题。

6.5.5 练习 5：一个简化的 CA-CFAR 计算

问题：某一维距离像中，待检测单元 CUT 的功率为 12。两侧参考单元的功率分别为

$$3,\ 4,\ 5,\ 4,\ 3,\ 5$$

假设保护单元已经跳过，不参与计算。若阈值系数取 $\alpha = 2$，问该单元是否判为目标?

解析：先计算参考单元平均值：

$$\hat{P}_n = \frac{3+4+5+4+3+5}{6} = 4$$

然后构造阈值：

$$T = \alpha \hat{P}_n = 2 \times 4 = 8$$

最后比较 CUT 与阈值：

$$12 > 8$$

因此判为有目标。这个练习最重要的不是算数，而是顺序：先估计局部背景，再形成局部阈值，最后判断 CUT。

6.5.6 练习 6：为什么需要保护单元

问题：如果把紧挨着 CUT 的单元也直接放进参考单元平均里，可能出现什么问题?

解析：如果 CUT 附近真的有目标，那么目标主瓣边缘或旁瓣能量可能会泄漏到相邻单元。若这些单元也拿去估计背景，就会把背景均值抬高。背景一旦被高估，阈值也会随之升高，原本应该检出的弱目标就更容易被压到阈值以下，因此漏检概率会上升。保护单元的作用，就是避免目标能量污染背景估计。

6.5.7 练习 7：CFAR 也会失效吗

问题：设想两个强目标距离很近，或者某一侧参考单元落在很强的杂波区内。此时简单的 CA-CFAR 可能遇到什么困难?

解析： CA-CFAR 的前提是：参考单元能够代表本地纯背景。但如果两个强目标靠得太近，参考单元里可能混进另一个目标；如果参考单元本身就落在强杂波区，背景估计也会被抬高。这样一来，阈值会偏高，真正的目标反而更容易漏掉。这就是 CA-CFAR 的典型弱点，也是后来出现 GO-CFAR、SO-CFAR、OS-CFAR 等改进方法的原因。

6.5.8 练习 8：编程实践（可选）

问题：用 MATLAB 或 Python 做一个简化的一维检测实验：

1. 构造一串背景噪声数据
2. 人为在其中插入一个或两个目标峰值
3. 先用固定阈值检测
4. 再用简化的 CA-CFAR 检测
5. 比较两种方法在背景变化前后的结果差异

解析：这道题可以从一个非常小的例子开始，例如先手工构造 20 到 30 个数据点。重点不是把程序一开始就写复杂，而是观察两个现象：在低背景区域，固定阈值和 CFAR 是否都能正常工作；在高背景区域，固定阈值是否开始大量误报，而 CFAR 的阈值是否会随着局部背景自动抬高。只要这两个现象能在图上看出来，这个实验就已经达到了目的。