# PN碼擴展資料之相位偏移校正與原始訊號還原技術分析

## 1. 緒論

在現代數位通訊系統中，直接序列展頻（Direct-Sequence Spread Spectrum, DSSS）技術因其抗干擾、低截獲率及多址接取（Code Division Multiple Access, CDMA）等優勢而被廣泛應用。DSSS系統的核心是使用偽隨機噪聲碼（Pseudorandom Noise code, PN code）將原始資料訊號擴展到較寬的頻帶上。PN碼具有良好的自相關特性，即在其本身未對準時自相關值很低，而在完全對準時則出現明顯的峰值，此特性對於訊號的同步與解擴至關重要。

本報告旨在解決一個特定的訊號還原問題：從一萬筆經過PN碼擴展的複數資料中恢復原始訊號。在此過程中，原始複數資料的實部（x軸）先經過一已知PN碼進行擴展，而後整個複數訊號在傳輸過程中經歷了未知的、隨時間變化的相位偏移（0至360度）。這一相位偏移實質上等效於載波相位誤差，是實際通訊系統中由於收發兩端本地振盪器不完全同步或通道變化等因素引起的常見現象。

核心挑戰在於，必須準確估計並補償此時變相位偏移，才能成功地對接收訊號進行解擴以還原原始資料。若相位未得到妥善校正，PN碼的相關性將大幅下降，導致解擴失敗。一個關鍵的理解是，相位旋轉是作用於整個複數訊號的，因此，相位校正必須在提取用於PN碼相關的實部之前，對複數訊號本身進行操作。若原始資料的實部 d[n] 被PN碼 p[n] 擴展為 d[n]p[n]，形成複數訊號 s\_{complex}[n] = d[n]p[n] + j \cdot 0（假設虛部初始為零或與實部PN擴展無關）。經過相位偏移 \theta[n] 後，接收訊號為 r[n] = (d[n]p[n]) e^{j\theta[n]}。為了提取用於PN碼相關的原始擴展實部，必須先將整個接收到的複數訊號 r[n] 乘以估計相位偏移的共軛複數 e^{-j\hat{\theta}[n]}，然後再取其時部進行PN碼相關運算。

針對此問題，本報告提出一套解決方案，包含：對接收資料進行分段處理；採用兩階段（粗調與細調）搜索策略估計相位偏移；利用已知PN碼與相位校正後訊號的實部進行相關運算，並以相關峰值作為相位估計準確性的評分；以及設計一種自我調整機制，當準確性評分低於預設門檻時，重新觸發相位估計，以應對相位偏移的時變特性。此時變特性使得單一的相位估計值不足以處理全部資料，必須採用分段或滑動窗口的方式進行自我調整追蹤。

## 2. 系統模型與問題定義

為清晰闡述解決方案，首先定義系統模型及相關參數。

* **原始資料 (d[n]):** 假設為一實數序列。根據問題描述，僅其x軸分量（實部）被PN碼擴展。
* **PN碼 (p[n]):** 一段已知的二位元序列（例如，由+1和-1組成）。PN碼通常由線性回饋移位暫存器（LFSR）產生，其特性由生成多項式和初始狀態決定。雖然使用者已提供PN碼，了解其來源有助於理解其特性。
* **擴展過程:** 原始實數資料 d[n] 與PN碼 p[n] 相乘，得到擴展後的實部訊號 s\_{real}[n] = d[n] \cdot p[n]。此 s\_{real}[n] 作為複數基頻訊號的實部，因此在相位偏移前的複數擴展訊號可表示為 s\_{complex}[n] = s\_{real}[n] + j \cdot 0（假設虛部為零）。
* **相位偏移 (\theta[n]):** 發射的複數訊號 s\_{complex}[n] 在傳輸過程中經歷了一個相位旋轉 e^{j\theta[n]}。此相位偏移 \theta[n] 是未知的，並且隨時間 n 變化。相位偏移的來源可能包括收發設備的電子元件特性、本地振盪器的相位噪聲以及通道的都卜勒效應等。
* **接收訊號 (r[n]):** 經過相位偏移後的接收訊號為 r[n] = s\_{complex}[n] \cdot e^{j\theta[n]} = (d[n]p[n]) \cdot e^{j\theta[n]}。這是一萬筆複數資料的起點。
* **目標:**
  1. 針對接收資料的每一分段，估計其相位偏移 \hat{\theta}[n]。
  2. 進行相位校正：r\_{corrected}[n] = r[n] \cdot e^{-j\hat{\theta}[n]}。
  3. 提取校正後的實部：x\_{corrected}[n] = \text{real}(r\_{corrected}[n])。
  4. 利用已知的PN碼 p[n] 對 x\_{corrected}[n] 進行解擴，以恢復原始資料的估計值 \hat{d}[n]。若原始資料 d[n] 代表符元，則解擴通常涉及在一個符元週期內對 x\_{corrected}[n] \cdot p[n] 的乘積進行累加或平均。

在C++實現中，將廣泛使用 std::complex<double> 來表示和操作複數資料，例如相位旋轉可透過 std::exp(std::complex<double>(0, angle\_rad)) 實現，複數乘法則直接使用重載運算子。

由於相位偏移 \theta[n] 是時變的，這直接意味著不能對全部一萬筆資料使用單一的相位估計值。若 \theta[n] 在資料傳輸期間發生顯著變化，則從一部分資料中獲得的相位估計 \hat{\theta} 對於另一部分資料可能是不準確的。這將導致相位校正效果不佳，解擴後的相關性得分降低，最終使得恢復的資料 \hat{d}[n] 產生較大誤差。因此，必須將資料分段（block-wise processing），並對每一段資料獨立估計其相位偏移，或採用追蹤機制。

DSSS系統的PN解擴依賴於處理增益（processing gain）。原始資料位元或符元的傳輸速率遠低於PN碼的碼片速率（chip rate）。在解擴過程中，將相位校正後的接收訊號與本地PN碼複本相乘並累加（或積分）。若資料位元/符元被長度為 L 的PN碼擴展，則解擴時會累加 L 個乘積項。此累加過程能有效地平均掉噪聲（本問題中忽略噪聲）和未對準的PN碼干擾，同時同調地累加期望的訊號分量，從而實現處理增益。

解擴的成功與否，以及作為準確性評分的相關峰值大小，對相位估計 \hat{\theta}[n] 的準確性高度敏感。即使是 \hat{\theta}[n] 的微小誤差，也可能導致解擴後訊號功率的顯著損失，甚至造成「災難性的信噪比惡化」（catastrophic SNR degradation）。若 \hat{\theta}[n] 不準確，則 r\_{corrected}[n] 仍將帶有殘餘相位誤差。此時，\text{real}(r\_{corrected}[n]) 與 p[n] 進行相關運算時，殘餘相位誤差會降低相關峰值的幅度。這種敏感性使得相關峰值成為一個衡量相位鎖定程度的良好指標。

## 3. 相位偏移估計與資料還原方法論

本節詳細闡述相位偏移估計和資料還原的具體步驟和策略。

### 3.1. 接收資料預處理：複數資料表示

接收到的一萬筆資料是複數形式的。在C++中，這些資料應存儲於 std::vector<std::complex<double>> 類型的容器中。std::complex 類別範本提供了處理複數所需的各種算術運算（如加法、乘法）以及標準數學函數（如 std::exp、std::real、std::imag、std::abs、std::arg），這對於後續的相位旋轉和訊號處理至關重要。例如，一個複數 Z = a + bi 可以通過 std::complex<double> z(a, b); 來創建。雖然浮點數運算本身存在精度問題，對於本專案的資料長度和運算複雜度，使用 double 型別的複數通常能夠提供足夠的精度。

### 3.2. 相位偏移估計策略

考慮到相位偏移的時變性，資料將被分成若干區塊（block）或段（segment）進行處理。對於每個區塊的接收複數資料 r\_{block}[k]，採用兩階段的搜索策略來估計相位偏移：

#### 3.2.1. 粗調階段 (Coarse Adjustment Stage)

1. **定義試驗角度集：** 設定一組粗略的試驗角度。例如，將 0^\circ 到 350^\circ 的範圍以 10^\circ 為間隔劃分成36個角度（即 0, \pi/18, 2\pi/18, \dots, 35\pi/18 弧度）。
2. **旋轉與相關：** 對於每一個粗調試驗角度 \theta\_{coarse}：
   * 將當前資料區塊 r\_{block}[k] 進行反向旋轉：r\_{rotated\\_block}[k] = r\_{block}[k] \cdot \exp(-j\theta\_{coarse\\_rad})。這裡 \exp(-j\theta\_{coarse\\_rad}) 可以使用 std::exp(std::complex<double>(0, -theta\_coarse\_rad)) 計算。
   * 提取旋轉後資料的實部：x\_{rotated\\_block}[k] = \text{real}(r\_{rotated\\_block}[k])。
   * 計算 x\_{rotated\\_block}[k] 與已知PN碼 p[n] 在該區塊長度內的相關值。此相關值（或其絕對值/功率）將作為該試驗角度 \theta\_{coarse} 的「準確性評分」。PN碼的同步擷取（acquisition）過程即依賴於此相關性。相關運算可以簡化為乘積和：\text{Score}(\theta\_{coarse}) = \left| \sum\_{k} (x\_{rotated\\_block}[k] \cdot p[k \pmod{L\_{pn}}]) \right|，其中 L\_{pn} 是PN碼的長度。
3. **選擇最佳粗調角度：** 找出產生最高準確性評分的粗調角度 \theta\_{best\\_coarse}。這是一種最大選擇決策模式（maximum-selection decision mode）。

#### 3.2.2. 細調階段 (Fine Adjustment Stage)

1. **定義搜索範圍與步長：** 在 \theta\_{best\\_coarse} 周圍設定一個較窄的搜索範圍，例如 \theta\_{best\\_coarse} \pm 10^\circ。同時，使用更小的角度步長，例如 0.1^\circ 或 1^\circ。
2. **重複旋轉與相關：** 在此更精細的範圍內，對每一個細調試驗角度 \theta\_{fine} 重複粗調階段的旋轉、實部提取和相關運算過程。
3. **確定最終估計相位：** 產生最高準確性評分的細調角度 \theta\_{fine} 即為當前資料區塊的最終相位估計值 \hat{\theta}。

此種兩階段搜索策略是在計算效率和估計精度之間的一種權衡。粗調階段快速定位可能的相位區間，而細調階段則在該區間內進行更精確的搜索。決策導向的載波恢復技術（decision-directed carrier recovery）也有類似思想，即根據一個初步判決（例如最接近的星座點）來調整相位 。在本情境中，「判決」即為試驗角度，而其「誤差」則反映在相關性評分上。當接收訊號的相位與本地PN碼複本的相位匹配時，相關性達到最大，這是使用相關峰值進行相位估計的根本原理。

粗調和細調的步數與步長是重要的設計參數。更多的步數和更小的步長可以提高估計精度，但同時也會增加計算負擔。每個試驗角度都涉及到對整個資料區塊進行複數乘法、取實部以及相關運算。例如，若粗調有36步，細調有20步（例如在 \pm 10^\circ 範圍內以 1^\circ 為步長），則每個資料區塊需要進行56次完整的相關運算。

雖然單純依賴最大相關峰值進行選擇在無噪聲環境下是可行的，但在實際應用中（或考慮到數值精度影響時），若相關性對角度的曲線存在多個局部極大值或峰頂平緩，則簡單取最大值可能不夠穩健。PN碼的自相關函數的「尖銳度」會影響細調階段的精度。一個具有非常尖銳自相關峰的PN碼有助於更精確地定位最佳角度。

以下表格總結了相位估計中可能使用的參數：

**表格 1: 相位估計參數**

| 參數名稱 | 符號 | 建議初始值/範圍 | 理據/註釋 |
| --- | --- | --- | --- |
| 粗調搜索角度步長 | \Delta\theta\_{coarse} | 10^\circ ($ \pi/18 $ rad) | 平衡計算量與初步定位精度；對應使用者提及的「36份」 |
| 粗調搜索範圍 | R\_{coarse} | 0^\circ 至 359^\circ | 覆蓋所有可能的相位偏移 |
| 細調搜索角度步長 | \Delta\theta\_{fine} | 1^\circ ($ \pi/180 $ rad) 或更小 | 在粗調結果附近進行更精確的搜索 |
| 細調搜索範圍 (相對粗調峰值) | R\_{fine\\_offset} | \pm 10^\circ (即粗調步長的範圍) | 集中搜索資源於最有可能的區域 |
| 資料區塊大小 | N\_{block} | 例如 256, 512, 或 1024 個樣本 | 需依據相位變化速率決定；太小則相關不穩定，太大則區塊內相位可能已變化 (詳見4.1節) |

### 3.3. 準確性評分機制

「準確性評分」是指導相位估計過程的核心指標，其本質是衡量在某個試驗相位校正角度下，訊號與PN碼的對齊程度。具體而言，將接收資料區塊用試驗角度 \theta\_{trial} 旋轉後，提取其實部 x\_{rotated}[k]，然後計算其與PN碼 p[k \pmod{L\_{pn}}] 的相關總和： S(\theta\_{trial}) = \sum\_{k=0}^{N\_{block}-1} (x\_{rotated}[k] \cdot p[k \pmod{L\_{pn}}]) 為了使評分更加穩健並反映訊號能量，通常使用相關總和的絕對值 |S(\theta\_{trial})| 或其功率 (S(\theta\_{trial}))^2 作為最終的準確性評分。使用功率值可以更強調強相關性，類似於能量檢測的概念。此評分是在粗調和細調的每個試驗角度下計算的，使得評分最大化的角度被認為是該階段的最佳估計。

此評分實質上是在特定試驗相位校正下解擴訊號強度的度量。當相位得到完美補償時，校正後訊號的實部將最大程度地與PN碼對齊，從而使相關總和達到最大值。若原始實部資料 d[k] 在一個PN碼週期 L\_{pn} 內近似恆定，且PN碼為雙極性（+1/-1），則完美對齊時的相關總和（解擴後）近似為 \sum (d[k] p[k] p[k]) = \sum d[k] = L\_{pn} \cdot d\_{symbol} （因 p[k]^2 = 1），這代表了資料符元能量乘以PN碼長度。

雖然對於單次搜索中尋找最大值而言，評分是否標準化影響不大，但若此評分後續將用於與一個絕對門檻值（見4.2節）比較，則可能需要進行標準化（例如，除以區塊長度或預期訊號功率），以使得門檻值的設定對於訊號電平或區塊大小的變化更具魯棒性。

儘管誤碼率（Bit Error Rate, BER）是衡量通訊質量的最終指標，但在相位估計階段，原始資料 d[n] 尚屬未知，無法直接計算BER。因此，相關峰值的大小是作為良好解調條件的一個代理指標。高相關性意味著良好的相位對準，這通常預示著後續的解擴和資料判決將具有較低的BER。

### 3.4. PN碼解擴與資料還原

一旦確定了某資料區塊的最佳相位估計值 \hat{\theta}，即可進行相位校正和資料還原：

1. **相位校正：** 將最佳相位估計 \hat{\theta} 應用於當前資料區塊：r\_{corrected\\_block}[k] = r\_{block}[k] \cdot \exp(-j\hat{\theta}\_{rad})。
2. **提取實部：** x\_{corrected\\_block}[k] = \text{real}(r\_{corrected\\_block}[k])。此 x\_{corrected\\_block}[k] 即為原始擴展實部訊號 d[k]p[k \pmod{L\_{pn}}] 的估計。
3. **PN碼解擴：** 將校正後的實部訊號 x\_{corrected\\_block}[k] 與PN碼 p[k \pmod{L\_{pn}}] 相乘，得到逐點的解擴訊號：despread\\_signal\\_points[k] = x\_{corrected\\_block}[k] \cdot p[k \pmod{L\_{pn}}]。
4. **資料符元估計：** 若一個原始資料位元/符元 d\_i 跨越 L\_{pn} 個PN碼碼片（即PN碼的一個週期長度），則 d\_i 的估計值可通過對相應的 L\_{pn} 個 despread\\_signal\\_points[k] 值進行累加或平均得到： \hat{d}\_i = \frac{1}{L\_{pn}} \sum\_{j=0}^{L\_{pn}-1} despread\\_signal\\_points[i \cdot L\_{pn} + j] 隨後，可以對 \hat{d}\_i 進行判決（例如，若 \hat{d}\_i > 0 則判為1，否則判為0）以恢復二進制資料。

此過程即為展頻通訊中的解擴（de-spreading）操作，旨在從展頻訊號中恢復窄頻的原始資訊。

此處假設資料符元的邊界已知或與PN碼的重複週期對齊。若符元邊界未知，則需要額外的符元同步步驟，這超出了當前問題的明確範圍。若校正不完美，存在殘餘相位誤差 \Delta\theta，則校正後複數訊號的實部會變為 (d[k]p[k])\cos(\Delta\theta) - (\text{虛部})\sin(\Delta\theta)。這將導致期望訊號分量被因子 \cos(\Delta\theta) 衰減，並可能引入由虛部（理想情況下校正後應為零）帶來的干擾，從而影響最終資料還原的準確性。

## 4. 時變相位偏移的適應性追蹤

由於相位偏移 \theta[n] 隨時間變化，採用固定相位校正值處理全部資料是不可行的。必須引入自我調整追蹤機制。

### 4.1. 資料分段處理概念

解決時變相位偏移的關鍵策略是將全部一萬筆接收樣本資料分割成較短的、連續的資料區塊（data blocks）或訊框（frames）進行處理。例如，每 N\_{block} 個樣本構成一個區塊。相位估計過程（粗調與細調搜索）將針對每個區塊獨立進行，或者，前一個區塊的相位估計結果可以用作當前區塊搜索的初始猜測值，這有助於縮小搜索範圍或直接進入細調階段，從而提高效率。

區塊大小 N\_{block} 的選擇是一個重要的權衡：

* **N\_{block} 過小：** 每個區塊包含的樣本數不足，可能導致相關運算結果不夠穩定或「有噪」（即使在忽略通道噪聲的情況下，短時相關也可能因訊號本身的隨機性而波動），使得相位估計不夠準確。
* **N\_{block} 過大：** 如果相位 \theta[n] 在一個區塊的持續時間內發生了顯著變化，那麼對整個區塊使用單一的相位估計值 \hat{\theta} 將會對區塊內的某些部分不適用，導致校正效果不佳。文獻中提到，載波頻率偏移（CFO）的變化可能在毫秒級，而相位偏移本身可能變化更快，甚至在符元級別。

理想情況下，區塊大小的選擇應基於相位偏移的相干時間（coherence time），即相位保持相對穩定的時間長度。若相位變化的速率已知或可以估計，則 N\_{block} 應小於該相干時間對應的樣本數。然而，在實際情況中，相干時間通常是未知的。因此，可以從一個經驗性的、相對保守的（較小的）N\_{block} 值開始（例如，幾個PN碼週期的長度），並根據後續的準確性監控結果（4.2節）來判斷是否需要調整。若重新估計頻繁觸發，可能意味著 N\_{block} 相對於相位變化速率而言過大。

雖然更高級的方法可能包括自我調整區塊大小 或使用重疊的滑動窗口（sliding window）以獲得更平滑的相位估計，但對於本專案，採用固定大小的連續（非重疊）區塊是一個合理的起點，實現相對簡單。前一個區塊的相位估計值 \hat{\theta}\_{i-1} 可以作為當前區塊 \hat{\theta}\_i 細調搜索的中心點，這是一種簡單的追蹤形式，前提是相位變化相對緩慢且連續。

### 4.2. 準確性監控與重新估計觸發機制

在處理完一個資料區塊並獲得其最佳相位估計 \hat{\theta} 及對應的最大準確性評分 S\_{max\\_current\\_block} 後，需要對此評分進行監控，以判斷相位估計的有效性，並在必要時觸發重新估計。

1. **設定門檻值 (S\_{threshold}):** 需要定義一個「失效」或「不及格」的準確性評分門檻值 S\_{threshold}。當 S\_{max\\_current\\_block} 低於此門檻時，表明當前的相位估計（即使經過細調）質量不佳。此門檻值的選擇至關重要。它可以是：
   * 一個固定值，基於對系統在良好鎖定狀態下預期能達到的最小相關性評分的了解。
   * 一個相對值，例如，近期觀察到的高質量評分的一個百分比（例如，最大可能評分的50%-70%）。在無噪聲情況下，理論上的最大相關值（完美對準時）是可以估算的（與訊號幅度和PN碼長度有關）。
   * 自我調整門檻值：考慮到訊號幅度可能存在的緩慢變化（即使問題假設原始資料幅度固定，實際系統中可能存在AGC不完美等因素），門檻值可以基於近期「鎖定」狀態下 S\_{max} 的滑動平均值來動態調整。
2. **重新估計觸發條件：** 若 S\_{max\\_current\\_block} < S\_{threshold}，則觸發重新估計程序。這可能由以下原因造成：
   * 相位從前一個區塊到當前區塊發生了快速跳變，導致基於前一區塊估計的細調搜索範圍未能覆蓋真實相位。
   * 當前區塊的初始粗調搜索未能找到一個足夠好的候選區域。
3. **重新估計行動：** 一旦觸發，系統應對當前資料區塊重新執行相位估計。這可能意味著：
   * 重新進行一次完整的粗調搜索（例如，從0到350度，步長10度）。
   * 或者，採用更為積極的細調策略，例如擴大細調範圍或減小細調步長。

使用者提出的問題「準確率通常差多少會判斷說是不及格，要再重新進行一次角度猜測」直接關係到 S\_{threshold} 的設定。此門檻的設定是一個權衡：設定過低，可能導致系統對真實的失鎖情況不敏感；設定過高，則可能因為微小的評分波動而頻繁觸發不必要的重新估計，增加計算負擔。

為避免在評分值圍繞門檻波動時系統在「追蹤」和「重新擷取」狀態間頻繁切換（即「乒乓效應」），可以引入遲滯（hysteresis）機制。例如，進入重新估計狀態的門檻（S\_{threshold\\_low}）可以比退出重新估計狀態並恢復正常追蹤的門檻（S\_{threshold\\_high}，其中 S\_{threshold\\_high} > S\_{threshold\\_low}）要求更高。

以下表格概述了自我調整追蹤中涉及的關鍵參數：

**表格 2: 自我調整追蹤參數**

| 參數名稱 | 符號 | 設定準則/方法 | 設定過低/過高的影響 |
| --- | --- | --- | --- |
| 準確性評分門檻值 | S\_{threshold} | 1. 固定值 (基於預期最小性能)。 <br> 2. 相對值 (例如，近期最大評分的70%)。 <br> 3. 自我調整門檻 (基於近期評分的滑動平均)。 | 過低：可能錯過失鎖情況。 <br> 過高：可能頻繁觸發不必要的重新估計。 |
| 重新估計搜索策略 | - | 1. 重置為完整粗調搜索。 <br> 2. 擴大細調搜索範圍。 <br> 3. 減小細調步長。 | 策略保守則重新擷取慢；策略積極則計算量大。 |
| 最大重新估計嘗試次數 (單一區塊) | N\_{retries} | 例如 2-3 次 | 過低：可能過早放棄困難區塊。 <br> 過高：可能在無法恢復的區塊上浪費過多計算資源。 |

## 5. 詳細解決方案流程圖

下面將以流程圖形式展示整個解決方案的執行步驟。

graph TD  
 A[開始] --> B{初始化參數};  
 B --> C{載入已知PN碼};  
 C --> D{設定區塊大小 $N\_{block}$};  
 D --> E{設定準確性門檻 $S\_{threshold}$};  
 E --> F{讀取10,000筆接收複數資料};  
  
 F --> G{For each block of $N\_{block}$ samples (當前區塊)};  
 G -- No more blocks --> Z[結束];  
 G -- Next block --> H{相位估計子流程 (當前區塊)};  
  
 H --> I{粗調搜索};  
 I --> J{For $\theta\_{coarse}$ from $0^\circ$ to $350^\circ$ (step $10^\circ$)};  
 J -- Next $\theta\_{coarse}$ --> K{旋轉當前區塊: $r' = r \cdot e^{-j\theta\_{coarse}}$};  
 K --> L{提取實部: $x' = \text{real}(r')$};  
 L --> M{計算相關評分: $S\_{coarse}(\theta\_{coarse}) = |\sum x' \cdot p|$};  
 M --> J;  
 J -- All $\theta\_{coarse}$ done --> N{找到最佳粗調角 $\theta\_{best\\_coarse}$ (對應最大 $S\_{coarse}$)};  
  
 N --> O{細調搜索};  
 O --> P{For $\theta\_{fine}$ in range $[\theta\_{best\\_coarse} \pm 10^\circ]$ (step $1^\circ$)};  
 P -- Next $\theta\_{fine}$ --> Q{旋轉當前區塊: $r'' = r \cdot e^{-j\theta\_{fine}}$};  
 Q --> R{提取實部: $x'' = \text{real}(r'')$};  
 R --> S{計算相關評分: $S\_{fine}(\theta\_{fine}) = |\sum x'' \cdot p|$};  
 S --> P;  
 P -- All $\theta\_{fine}$ done --> T{找到最佳估計角 $\hat{\theta}$ (對應最大 $S\_{fine}$)};  
 T --> U{當前區塊最大評分 $S\_{max\\_curr} = \max(S\_{fine})$};  
  
 U --> V{判斷: $S\_{max\\_curr} < S\_{threshold}$?};  
 V -- Yes (準確性低) --> W{重新估計? (例如: 嘗試次數 < $N\_{retries}$ )};  
 W -- Yes --> H; %% 返回相位估計子流程，對同一區塊重試  
 W -- No (達到最大嘗試次數 或 不重試) --> X\_Fail[標記區塊為失敗/跳過];  
 X\_Fail --> G; %% 處理下一個區塊  
  
 V -- No (準確性OK) --> X{相位校正與解擴};  
 X --> Y1{校正相位: $r\_{corrected} = r\_{block} \cdot e^{-j\hat{\theta}}$};  
 Y1 --> Y2{提取實部: $x\_{corrected} = \text{real}(r\_{corrected})$};  
 Y2 --> Y3{PN碼解擴: $\hat{d}\_{block} = \text{Despread}(x\_{corrected}, p)$};  
 Y3 --> Y4{儲存解擴資料 $\hat{d}\_{block}$};  
 Y4 --> G; %% 處理下一個區塊  
  
 Z[結束] --> Z\_Out{輸出所有已恢復的 $\hat{d}$};

**流程圖說明：**

1. **初始化：** 設定系統運行所需的各項參數，包括PN碼、區塊大小、搜索步長、準確性門檻值等。
2. **分塊循環：** 逐塊處理接收到的複數資料。
3. **相位估計子流程：**
   * **粗調：** 在 0^\circ 至 350^\circ 範圍內以較大步長（如 10^\circ）進行搜索，對每個試驗角度計算相關評分，找到評分最高的粗調角度 \theta\_{best\\_coarse}。
   * **細調：** 在 \theta\_{best\\_coarse} 附近的一個小範圍內（如 \pm 10^\circ）以更小步長（如 1^\circ）進行精確搜索，找到評分最高的細調角度 \hat{\theta}，此即為當前區塊的最終相位估計。
4. **準確性檢查：** 將得到的最大相關評分 S\_{max\\_current\\_block} 與預設門檻 S\_{threshold} 比較。
   * 若低於門檻（準確性不足）：可選擇對當前區塊重新執行相位估計（可能擴大搜索範圍或採用不同策略），或在達到一定重試次數後標記此區塊處理失敗並繼續處理下一區塊。流程圖中示意為可重試。
   * 若高於或等於門檻（準確性足夠）：則認為相位估計有效。
5. **相位校正與解擴：** 使用估計出的 \hat{\theta} 對當前區塊的複數資料進行相位校正，提取校正後訊號的實部，再利用PN碼進行解擴，得到原始資料的估計值 \hat{d}\_{block}。
6. **儲存與輸出：** 儲存每個區塊解擴後的資料，待所有區塊處理完畢後統一輸出。

此流程隱含了狀態管理的概念：系統可能處於「追蹤」（準確性OK）或「重新擷取」（準確性低）的狀態。對於第一個資料區塊，由於沒有前一個區塊的相位資訊，必須從完整的粗調搜索開始。對於後續區塊，若前一區塊追蹤良好，其相位估計 \hat{\theta}\_{i-1} 可作為當前區塊 \hat{\theta}\_i 細調搜索的中心，以加速處理。若準確性檢查失敗，則可能需要退回到完整的粗調搜索。同時，應考慮為單一區塊的重新估計設定最大嘗試次數，以避免在無法恢復的資料段上陷入無限循環。

## 6. C++ 程式架構設計

為實現上述流程，建議採用模組化的C++程式架構。

### 6.1. 核心資料結構

* std::vector<std::complex<double>> received\_samples;：存儲所有輸入的複數樣本。
* std::vector<double> pn\_code;：存儲已知的PN碼序列（例如，+1.0, -1.0）。使用者提及「x的pn code」，暗示其為實數序列。
* std::vector<double> recovered\_data;：存儲最終解擴得到的原始資料估計值。
* （可選）struct SignalBlock { std::vector<std::complex<double>> samples; double estimated\_phase\_rad; double max\_correlation\_score; bool processing\_successful; };：用於封裝每個資料區塊及其處理結果，便於管理。

### 6.2. 主要模組/類別設計建議

建議將功能劃分為以下幾個主要類別或模組：

1. **DataManager 類別 (或一組函式)**
   * **職責：** 負責讀取輸入資料（例如，從檔案讀取，或為測試目的生成模擬資料），並按需提供資料區塊。
   * **主要方法：**
     + bool loadSamples(const std::string& filePath);
     + std::vector<std::complex<double>> getBlock(size\_t block\_index, size\_t block\_size);
     + size\_t getTotalSamples() const;
     + bool loadPNCode(const std::string& filePath); (或直接在建構函式中初始化)
     + const std::vector<double>& getPNCode() const;
2. **PhaseProcessing 類別 (或一組函式)**
   * **職責：** 執行複數訊號的相位旋轉及相關運算。
   * **主要方法：**
     + std::vector<std::complex<double>> rotateSamples(const std::vector<std::complex<double>>& input\_samples, double angle\_rad);
       - 內部實現：對每個樣本 s，計算 s \* std::exp(std::complex<double>(0.0, -angle\_rad))。
     + double calculateCorrelationScore(const std::vector<std::complex<double>>& complex\_samples, const std::vector<double>& pn\_sequence);
       - 內部實現：遍歷 complex\_samples，取其實部 std::real(sample)，與對應的 pn\_sequence 元素（注意PN碼的循環使用，如 pn\_sequence[k % pn\_sequence.size()]）相乘，然後累加這些乘積。返回累加和的絕對值或平方值。雖然存在如GNU Radio、Armadillo等DSP函式庫，但對於此特定任務，直接實現相關運算更為簡潔且符合學習目的。
3. **AngleEstimator 類別**
   * **職責：** 實現兩階段（粗調與細調）的相位估計邏輯。依賴 PhaseProcessing 模組進行旋轉和相關計算。
   * **成員變數：** 粗調/細調的角度步長、搜索範圍等參數。
   * **主要方法：**
     + double estimateCoarsePhase(const std::vector<std::complex<double>>& block\_samples, const std::vector<double>& pn\_sequence, double& out\_correlation\_score);
     + double estimateFinePhase(const std::vector<std::complex<double>>& block\_samples, const std::vector<double>& pn\_sequence, double coarse\_phase\_guess\_rad, double& out\_correlation\_score);
     + double estimateBestPhase(const std::vector<std::complex<double>>& block\_samples, const std::vector<double>& pn\_sequence, double initial\_guess\_rad, bool perform\_coarse\_search, double& out\_max\_score); (整合性方法)
4. **DataRecovery 類別 (或一組函式)**
   * **職責：** 對經過相位校正的資料區塊（實部）和PN碼進行解擴，輸出恢復的資料符元。
   * **主要方法：**
     + std::vector<double> despreadData(const std::vector<double>& real\_corrected\_samples, const std::vector<double>& pn\_sequence, size\_t symbol\_length\_in\_chips);
       - 內部實現：將 real\_corrected\_samples 與 pn\_sequence 逐點相乘，然後按 symbol\_length\_in\_chips 的長度分段累加（或平均）以得到每個資料符元的估計值。
5. **MainController 類別 (或 main() 函式中的主要邏輯)**
   * **職責：** 協調整個處理流程，管理各模組間的資料流動，實現分塊處理循環以及重新估計的觸發邏輯。
   * **成員變數：** DataManager, PhaseProcessing, AngleEstimator, DataRecovery 的實例（或作為靜態工具類使用其函式），以及處理參數如 block\_size, S\_threshold。
   * **主要方法：** void processAllData();

此種基於類別的劃分能提升程式碼的模組化程度，使得開發、測試和除錯更加容易。例如，PhaseProcessing 中的相關運算可以獨立於其他邏輯進行單元測試。此外，關鍵參數（如區塊大小、搜索步長、門檻值）應設計為可配置的，例如通過建構函式參數傳遞或從設定檔讀取，而非硬編碼，以方便實驗和調優。

**表格 3: C++ 核心模組與職責**

| 模組/類別名稱 | 核心職責 | 主要方法範例 (部分) | 主要處理的資料結構 |
| --- | --- | --- | --- |
| DataManager | 讀取輸入樣本和PN碼，按需提供資料區塊。 | loadSamples(), getBlock(), loadPNCode(), getPNCode() | std::vector<std::complex<double>>, std::vector<double> |
| PhaseProcessing | 執行複數訊號的相位旋轉，計算旋轉後實部與PN碼的相關評分。 | rotateSamples(), calculateCorrelationScore() | std::vector<std::complex<double>>, std::vector<double> |
| AngleEstimator | 實現粗調和細調相位搜索邏輯，確定最佳相位估計。 | estimateCoarsePhase(), estimateFinePhase(), estimateBestPhase() | std::vector<std::complex<double>>, std::vector<double> |
| DataRecovery | 對相位校正後的實部訊號進行PN碼解擴，恢復原始資料符元。 | despreadData() | std::vector<double> |
| MainController | 協調整體流程，管理資料區塊處理、相位估計、準確性監控、重新估計觸發及最終資料恢復與儲存。 | processAllData() (包含主循環邏輯) | (協調其他模組，處理 SignalBlock 等中介結構) |

### 6.3. 模組間互動示意

1. MainController (或 main() 函式) 創建各模組實例。
2. MainController 指示 DataManager 載入接收樣本和PN碼。
3. MainController 進入主循環，按 block\_size 從 DataManager 獲取當前資料區塊。
4. MainController 調用 AngleEstimator 的方法（例如 estimateBestPhase）來估計當前區塊的相位。
   * AngleEstimator 內部會多次調用 PhaseProcessing 的 rotateSamples 和 calculateCorrelationScore 方法來執行搜索。
5. AngleEstimator 返回最佳相位估計 \hat{\theta} 和對應的最大相關評分 S\_{max} 給 MainController。
6. MainController 比較 S\_{max} 與 S\_{threshold}：
   * 若 S\_{max} < S\_{threshold}：MainController 決定是否對當前區塊重新估計（例如，調用 AngleEstimator 時傳遞指示強制進行完整粗調的參數），或記錄失敗並處理下一區塊。
   * 若 S\_{max} \ge S\_{threshold}：
     1. MainController 使用 PhaseProcessing::rotateSamples 方法，以 \hat{\theta} 對當前區塊進行最終的相位校正，得到 r\_{corrected\\_block}。
     2. 提取 x\_{corrected\\_block} = \text{real}(r\_{corrected\\_block})。
     3. MainController 調用 DataRecovery::despreadData 方法，傳入 x\_{corrected\\_block} 和PN碼，得到解擴後的資料符元 \hat{d}\_{block}。
     4. MainController 將 \hat{d}\_{block} 存入 recovered\_data。
7. 循環直至所有資料區塊處理完畢。
8. MainController 輸出 recovered\_data。

清晰的資料流和控制流劃分有助於架構的理解和實現。

### 6.4. 關鍵演算法的C++實現思路

* **相位旋轉 (單一樣本):**  
  std::complex<double> rotateSample(const std::complex<double>& input\_sample, double angle\_radians) {  
   // angle\_radians 為負表示反向旋轉以校正相位  
   return input\_sample \* std::exp(std::complex<double>(0.0, -angle\_radians));  
  }
* **相關評分計算 (簡化示意，假設 complex\_block 已被某試驗角度旋轉過):**  
  double calculateScore(const std::vector<std::complex<double>>& rotated\_complex\_block,  
   const std::vector<double>& pn\_sequence) {  
   double correlation\_sum = 0.0;  
   if (rotated\_complex\_block.empty() |

| pn\_sequence.empty()) { return 0.0; // 或拋出錯誤 } for (size\_t i = 0; i < rotated\_complex\_block.size(); ++i) { double real\_part = std::real(rotated\_complex\_block[i]); correlation\_sum += real\_part \* pn\_sequence[i % pn\_sequence.size()]; } // 可以返回絕對值或功率 return std::abs(correlation\_sum); // return correlation\_sum \* correlation\_sum; // 功率 } ``` pn\_sequence[i % pn\_sequence.size()] 確保了當區塊長度大於PN碼長度，或處理非PN碼整數倍長度的區塊時，PN碼能正確地循環使用。

* **PN碼解擴 (示意一個資料符元的恢復，假設 real\_corrected\_block 長度為 symbol\_len\_chips):**  
  double despreadSymbol(const std::vector<double>& real\_corrected\_symbol\_chips,  
   const std::vector<double>& pn\_sequence) {  
   double despread\_sum = 0.0;  
   size\_t symbol\_len\_chips = real\_corrected\_symbol\_chips.size();  
   if (symbol\_len\_chips == 0 |

| pn\_sequence.empty()) { return 0.0; // 或拋出錯誤 } // 假設 pn\_sequence 的長度至少為 symbol\_len\_chips，或能正確索引 for (size\_t i = 0; i < symbol\_len\_chips; ++i) { despread\_sum += real\_corrected\_symbol\_chips[i] \* pn\_sequence[i % pn\_sequence.size()]; } // 返回平均值，然後可進行判決 return despread\_sum / static\_cast<double>(symbol\_len\_chips); } ``` 相位估計時的相關運算可能在一個包含多個資料符元的區塊上進行，以獲得更穩定的相關峰值。而最終的資料解擴則應精確地在每個資料符元對應的碼片段上進行累加和判決。

## 7. 結論與建議

本報告詳細分析了從經過PN碼擴展且帶有時變相位偏移的複數資料中恢復原始訊號的技術挑戰，並提出了一套基於分塊處理、兩階段相位搜索、相關性評分以及自我調整重新估計的解決方案。該方案直接回應了使用者需求，旨在提供一個清晰且可操作的實施框架。

**核心方法總結：**

1. **資料分段：** 將長序列資料切分為較短區塊，以應對相位偏移的時變性。
2. **相位估計：** 對每個區塊，採用「粗調-細調」搜索策略。通過旋轉接收到的複數資料，提取實部與已知PN碼進行相關運算，以相關峰值（準確性評分）最大化為目標，找到最佳相位估計。
3. **相位校正與解擴：** 使用估計出的相位對該區塊資料進行校正，然後對校正後的實部訊號進行PN碼解擴，以恢復原始資料。
4. **自我調整追蹤：** 監控每個區塊處理後的最大相關評分。若評分低於預設門檻，則觸發對該區塊的相位重新估計機制。

**實施建議：**

1. **模組化開發與測試：** 建議從實現和獨立測試基礎模組（如 PhaseProcessing 中的相位旋轉和相關計算，DataRecovery 中的解擴邏輯）開始。使用已知輸入和預期輸出的測試案例來驗證每個模組的正確性。
2. **參數調優：** 系統性能高度依賴於多個關鍵參數，如資料區塊大小 (N\_{block})、粗調/細調的搜索步長和範圍、以及準確性評分門檻 (S\_{threshold})。由於問題明確指出不考慮噪聲，參數調優的重點將是平衡計算複雜度與對相位變化追蹤的靈敏度和準確性。應預留充分時間進行實驗以找到適用於特定資料特性的最佳參數組合。
3. **日誌記錄：** 在開發和測試階段，詳細記錄中間結果（例如，每個試驗角度的相關評分、每個區塊估計出的相位、觸發重新估計的事件等）對於除錯和理解系統行為至關重要。
4. **門檻值設定：** 對於準確性評分門檻 S\_{threshold}，可以從一個經驗值開始（例如，在系統「鎖定」狀態下觀察到的典型最大評分的50%-70%），然後根據實際處理效果進行調整。考慮到無噪聲環境，理論上完美對準時的相關值是可以預估的，這有助於設定一個合理的初始門檻。
5. **驗證策略：** 評估資料還原效果的最佳方式是與原始資料 d[n] 進行比較。若原始資料已知（例如在模擬環境中自行生成發射訊號），則可以計算誤碼率（BER）或均方誤差（MSE）等量化指標。若原始資料未知，則持續獲得較高的相關性評分、以及隨時間平滑變化的相位估計值，可以作為系統工作正常的間接證據。

**未來擴展性思考：**

儘管本問題設定忽略了通道噪聲，但所設計的框架為未來可能的擴展奠定了基礎。若需考慮噪聲，則相關運算的積分長度（即區塊大小）、門檻設定策略（可能需要更複雜的自我調整門檻，如基於噪聲功率估計的CFAR類門檻）以及相位估計的穩健性將成為新的挑戰。

總體而言，通過精心設計和細緻實現上述流程與架構，應能有效解決所述的相位偏移校正與資料還原問題。

#### 引用的著作

1. Carrier recovery - Wikipedia, https://en.wikipedia.org/wiki/Carrier\_recovery