

中正國際機場都卜勒氣象雷達資料錯誤剖析與回復方法

楊健生

魏運寶

飛航服務總台氣象中心

中正氣象台

摘 要

中正國際機場都卜勒氣象雷達所蒐集之雷達氣象資料主要分為極座標與直角座標資料兩部分，而這些資料檔均經壓縮處理後再錄存於磁帶上。然因於民國78年6月5日前所使用之錄存軟體在壓縮過程中之演繹邏輯發生錯誤，以致部分資料在重取時會因少數資料的遺漏連帶影響整體資料產生移位之情形。本文即是針對此一問題加以剖析，並透過文中程式六、程式七之虛擬碼詳細說明解決極座標與直角座標檔因資料壓縮不當所遺留之問題。

一、前言

中正國際機場都卜勒氣象雷達於民國76年5月1日起運作至今近四年，各類型天氣資料之蒐集十分完整，頗受氣象學術界及實做單位之重視。此雷達系統資料之錄存係依原廠商（瑞典 ERICSSON 公司）提供之軟體進行，除具特定格式外，尚經壓縮以減少龐大之資料量。

由於資料壓縮過程演繹邏輯錯誤，以致解碼展開時原始資料會有部分遺失，經本氣象台通知原廠商後於民國78年6月5日更正錯誤之模組，自此雷達資料之錄存方得無誤。然原廠商並無提供錯誤資料回復之軟體與方法，為使兩年多寶貴的雷達資料得以正常使用及滿足各氣象界的需求，我們自行發展了一套軟體以解決這個存在已久的問題。

二、資料結構

由於中正國際機場都卜勒氣象雷達軟體係以 VAX/PASCAL 語言所撰寫，因此資料型態 (DATA TYPE) 與處理方式皆架構於該語言之上 (參 VAX/PASCAL USER'S MANUAL)，而資料檔案主要分為極座標資料檔與直角座標資料檔兩大類，其檔案資料儲存方法概述於下：

(一) 極座標資料

其檔案儲存格式第一個 BLOCK (1 BLOCK=512 BYTE) 為該空域掃描 (VOLUME SCAN) 之檔案頭 (FILE HEADER)，而檔案頭是由多筆記錄 (RECORD) 所組成之巢狀記錄 (NEST RECORD) 其中每一筆記錄包含一個或以上欄位 (FIELD) 不等，其詳細內容請參附表一。

由第二個 BLOCK 開始則依最低仰角至最高仰角存放各 PPI 之資料，每一仰角資料皆自成一筆，以 BLOCK 為儲存單位，如圖一所示。每一仰角之方位角資料則由正北順時鐘方向連續存放，其中每一波束 (BEAM, BEAM-WIDTH=0.86°) 前四個 BYTE 存放與此仰角開始掃描之時間差，而後則為該方位角 RANGE GATE 之資料，因此每一方位角之資料量為 124 BYTES (IN DOPPLER MODE 1KM/RANGE GATE; IN NON-DOPPLER MODE 2KM/RANGE GATE)，如圖二所示。

(二) 直角座標資料

直角座標資料是將極座標空域掃描 (VOLUME SCAN) 之資料以一事先建妥的轉換表 (CONVERSION TABLE) 填入直角座標中，其內插方式是把最靠近該網格點之資料當做該網格點之資料值，因此定高面雷達回波圖邊界附近之回波值皆為方位角波束最後幾個資料值 (若回波圖邊界出現不正常之環狀圈則視極座標資料被壓縮時遺失資料個數而定，此點在下節將予詳述)。檔案存放格式亦以一檔案頭 (參表一) 為起始，每層資料自成一筆以 BLOCK 為儲存單位，依最低高度至最高層存放各不同高度 CAPPI 資料，且每一高度 CAPPI 資料亦含一檔案頭 (佔 1 BLOCK)，續接之 BLOCK 始存放該層之資料，資料之存放順序為由西向東而後南向北遞增。直角座標資料之儲存方式如圖三所示。

三、壓縮之流程演繹與除錯

由於雷達資料過於龐大，為節省磁帶錄存空間與作業時間，錄存資料時便須先透過軟體做壓縮處理，然後

		7	Height turbulence.
		8	Accumulated precipitation.
		9	Height in km.
		10	Vertical profile indicator.
35	weath	enumeration	Reception weather. Not used.
		0	Drizzle.
		1	Rain.
		2	Showers.
		3	Drysnow.
		4	Wetsnow.
36	comment1	10 characters	First part of comment, written on display.
46	comment2	30 characters	Second part of comment.
76	sign	2 characters	Signature of operator storing picture.
78	place	20 characters	Place of radar, 6 first chars written on display.
98	geo_coord	2 reals	Geographical coordinates, long/lat of center of picture.
106	radar_coord	2 reals	Radar coordinates relative center of, picture in km.
114	radar_typ	enumeration	Radar type, doppler, normal, s/x-band etc.
115	east_uppb	integer	Number of elements = max index in east direction of picture.
119	east_size	real	Size of pixel element in east direction in km.
123	north_uppb	integer	Number of elements = max index in north direction of picture.
127	north_size	real	Size of pixel element in north direction in km.
131	hei_uppb	integer	Max index in height direction, min index = -1.
135	hei_size	real	Mean size of pixel element in height direction.
139	pixel_cnt	integer	Number of pixels following header.
143	store_min	integer	Smallest possible stored value.
147	store_max	integer	Largest possible stored value.
151	store_slope	real	Conversion to physical value.
155	store_ord	real	Conversion to physical value.
159	store_offset	integer	Number to subtract to get 0.
163	store_bits	integer	Number of used bits per datum.
167	store_align	integer	Number of allocated bits per datum.
171	store_quant	8 characters	Physical quantity.
179	compressed	boolean	The data of the file is compressed.
179	fill	99 characters	Fill bytes, to enable extension without invalidating old files.
279	pic_fil_type	enumeration	Picture file type, the data that follows depends on the type.
		0	Polar volume
		1	KYZ volume.
		2	Horizontal picture.
		3	Vertical picture, perpendicular
		4	Colour display image.
		5	Vertical picture, oblique
280	elev_uppb	integer	Polar, number of elevations = max index used.
284	elev	20 reals	Actual elevations in degrees.
364	azim_uppb	integer	Index of azimuth, min index = 0.
368	range_uppb	integer	Number of ranges = max index.
372	range_lim	3 integers	Limits in pixels for different range gates.
384	range_siz	3 reals	Size of range gates.
396	scan_size	integer	Size of each range scan, data follows after a header with a time stamp.
400	elev_block_nr	20 integers	Polar volume, starting block in file for different elevations.
280	pic_block_nr	17 integers	Rectangular volume, starting block in file for different height levels. Each level stored as a complete file with header and data.
348	pic_level	17 reals	Height levels in km, -1 is pseudo cappi, usually not used.
280	hei_nr	integer	Horizontal picture, height level nr.
284	hei	real	Height in km of plane.

288	acc_start	real	Time when accumulation was started, YYMMDDHHMMSS.
292	acc_interval	real	Time interval in seconds for the accumulation.
296	acc_lost	real	Delta time in seconds when the radar has been turned off, during accumulation.
280	north_nr	integer	Vertical perpendicular picture, defines place of cross in pixels, where applicable lower left corner (south-west) has coordinates 1,1.
284	east_nr	integer	Height levels in km, -1 is pseudo cappi, usually not used.
288	hei_level	17 reals	
292	north_nr_1	integer	Vertical oblique picture, points where section crosses picture borders
296	east_nr_1	integer	
300	north_nr_2	integer	
304	east_nr_2	integer	

再登錄於磁帶上；因該壓縮模組 (COMPRESSED MODULE) 之演繹邏輯錯誤 (BUG)，因此導至重取資料時發生資料缺失與位移現象。為確保未更新壓縮模組前錄存資料之正確性及可用性，即在不破壞原始資料的原則下，我們發展出一套軟體作資料復原之工作，以彌補原始資料的缺失。

茲將雷達資料壓縮原理、錯誤的壓縮流程和修正後的壓縮流程以及回復 (更正由於壓縮錯誤而造成資料缺失) 的方法概述於後：

(一) 壓縮原理 (COMPRESS PRINCIPLE)

當雷達極座標及直角座標資料錄存時須作壓縮處理，此壓縮技術則為當比鄰 4 個 BYTE 及以上資料相同時即作壓縮處理，而每一壓縮資料組則包刮相同資料之數量 (BYTE COUNT) 以及資料值 (VALUE)；每一組壓縮資料又可區分為兩部分，即資料頭 (DATA HEADER) 與資料值 (DATA VALUE)，其中資料頭長度可能為 1 或 2 個位元組視其後之相同資料值或不相同資料值的資料量而定，其詳細格式與說明如表二所示。

表二、壓縮格式

The header has the following format, the most significant bit described first	
size	meaning
6 or 14 bits	length A length indication, giving number of bytes in the expanded (uncompressed) data. If type=0 it also gives the number of bytes following header. If the header is two bytes long, the second byte gives the least significant part of the number.
1 bit	header length =0 means one byte header =1 means two byte header
1 bit	type =0 expanded data are the same as compressed data, ie the data are all different bytes. The number of data bytes following the header is given by the data length indication below =1 expanded data are all the same. One data byte follows the header. The expanded data consists of the same byte, replicated the number of times given by the length field
The control byte 00000001 has the special meaning 'end-of-data'; examples of compressed data, and the cooresponding expanded data	

```

xxxxxx00. data,data...
                A string of 1..63 different data bytes
xxxxxx01. data
                A string of 4..63 data bytes, all the same
xxxxxx10. xxxxxxxx, data, data, ....
                A string of 64..16383 different data bytes
xxxxxx11. xxxxxxxx. data
                A string of 64..16383 data bytes, all the same.

```

(二)檔案壓縮演繹邏輯 (COMPRESS ALGORITHM)

中正國際機場都卜勒氣象雷達之資料皆以壓縮檔案方式錄存，於取用時再加以解壓縮 (DECOMPRESSED)，然於民國78年6月5日以前之壓縮資料由於其壓縮模組 (MODULE) 邏輯上之錯誤導致部份錄存資料有誤致使解壓縮後之資料與原始資料不盡相同，因此本節將針對修正前、後之壓縮演繹邏輯加以分析及闡述；於下述討論中我們將使用 PASCAL之虛擬碼 (PSEUDO CODE) 此即利用 PASCAL語言的語法結構以及文字來說明和描述各模組在各個描象階段中之資料處理過程及流程。

COMPRESS_FILE模組 (程式一) 為將雷達觀測之原始資料檔 (IN_FILE) 轉換成壓縮檔案之處理流程，透過該模組將會產生另一個壓縮檔 (OUT_FILE) 以備錄存 (ARCHIVE) 於磁帶上，此檔案為原始資料之壓縮碼 (壓縮原理參前) 其檔案頭 (FILE HEADER) 與原始檔案頭相同未作壓縮處理但將會被標記為“已壓縮”(參表一)；然造成壓縮過程錯誤的原因乃發生於程式一裡之 COMPRESS_DATA 模組內，茲將其錯誤及修正後之演繹邏輯分述於程式二及程式三而其壓縮格式如表二所示。

```

Pseudocode
begin
  open input and output files
  {input 為雷達觀測之原始資料檔, output 為錄至磁帶之壓縮資料檔}

  read file-header
  check header
  {讀取原始資料檔案頭並檢查該檔案是否已壓縮及是否為雷達資料檔}

  write header to output file
  {檔案頭永不壓縮但會標註“已壓縮”, 參表一}

  initiate variables

  while not end of input data or again loop
    if not again then
      read input-data, if end of data set flag
    end

    COMPRESS_DATA ()

    if output buffer is full then
      write buffer to output-file
      update pointers
    end
    else
      set again = false
    end
  end loop

  if any rest-data then
    put data to the output-file
  end

  close input and output files
end

```

程式一 . COMPRESS_FILE

Pseudocode

```
begin
  initiate variables
  {大於三個資料量才作壓縮檢查處理否則視為不同資料不作壓縮}
  if number of input data > 3 then
    save the three last positions data
    insert instead 'end of data'-marks in those positions
    {每一批資料處理暫存區之大小為31個BLOCK, 於暫存區最後
    位址加放001以示結束該暫存區內之資料處理}

    while not done
      count number of different bytes
      count number of same bytes

      if end of data not reached then
        if number of same bytes > 3 then
          OUTPUT_DIFFERENT_BYTES ()      {參程式四}
          OUTPUT_SAME_BYTES ()          {參程式五}
        end
      end
    else
      OUTPUT_DIFFERENT_BYTES ()
      OUTPUT_SAME_BYTES ()
      set done flag
    end
  end {while}

  get the saved rest-data
  OUTPUT_DIFFERENT_BYTES ()
end
else
  OUTPUT_DIFFERENT_BYTES ()
end
end
```

程式二. COMPRESS_DATA

Pseudocode

```
begin
  initiate variables

  if number of input data > 3 then
    save the three last positions data
    insert instead 'end of data'-marks in those positions

    while not done
      count number of different bytes
      count number of same bytes

      if end of data not reached then
        if number of same bytes > 3 then
          OUTPUT_DIFFERENT_BYTES ()
          OUTPUT_SAME_BYTES ()
        end
      end
    else
      if number of same bytes < 4 then
        OUTPUT_DIFFERENT_BYTES ()
      else
        OUTPUT_DIFFERENT_BYTES ()
        OUTPUT_SAME_BYTES ()
        set done flag
      end
    end
  end {while}
```

```

    get the saved rest-data
    OUTPUT_DIFFERENT_BYTES ()
end
else
    OUTPUT_DIFFERENT_BYTES ()
end
end
end

```

程式三. COMPRESS_DATA

```

Pseudocode
begin
  if diff-byte-counter > 0 then
    if diff-byte-counter > 63 then
      write 'type 2' - header to output-string {參表二}
    else
      write 'type 0' - header to output-string {參表二}
    end
    copy data from input- to output-string
    clear diff-byte-counter
  end
end
end

```

程式四. OUTPUT_DIFFERENT_BYTES

```

Pseudocode
begin
  if same-byte-counter > 0 then
    if same-byte-counter > 63 then
      write 'type 3' - header to output-string {參表二}
    else
      write 'type 1' - header to output-string {參表二}
    end
    write databyte to output-string
    clear same-byte-counter
  end
end
end

```

程式五. OUTPUT_SAME_BYTES

(三)資料之回復 (CHECK AND RECOVER)

由以上之資料結構、壓縮原理以及演譯分析過程中可明確了解其壓縮發生錯誤之源由及發生錯誤位置，程式六（極座標資料檔案回復）及程式七（直角座標資料檔案回復）為解壓縮後檔案之資料檢查與回復流程。

於極座標資料檔案內之資料掃描檢查係利用每一波束 (BEAM) 前四個位元組 (與該仰角開始掃描之時間差，請參圖二) 作為該組方位角資料展開後位址檢查之指標，而直角座標資料檔案則以各高度層之檔案頭識別字 "PHERAWIS" 作為檢查指標，如發現展開後之資料檔發生位移時 (資料漏失) 則計算壓縮時造成錯誤位置並記錄

於錯誤表 (BUG TABLE) 內以作為回復工作時之依據。

程式六及程式七乃採 TWO PASS 方式處理，其中 PASS ONE 為做資料掃描檢查工作如發現壓縮錯誤則產生一資料錯誤對照表以做為 PASS TWO 復原檔案工作之參照表，而 PASS TWO 則依據 PASS ONE 所建立之錯誤對照表進行回復工作，如果該錯誤對照表為空的則示該檔案未發生壓縮時邏輯錯誤；此外於檢查掃描中如發現因壓縮時有誤導致展開後資料缺失位移情形則於 PASS TWO 回復時將漏失之資料填入 (插入) 無資料 (NO DATA) 之資料值 (VALUE)，並將整組資料由缺失位址位移至正確位置，資料儲存格式請參圖一、二、三。

Pseudocode

```
begin
- Open input file
  {解壓縮極座標資料檔案}
  Read file header
  {讀取檔案資料頭相關資訊, 參表一}

P   Initial all index and pointer variable
A   Loop
S   read whole elevation data into blocks address
    {以每一仰角資料作為檢查單位, 參圖一}
    examine each azimuth scan time
    {每一個BEAM前四個BYTE為與該仰角開始掃描之時間差如圖二示
    以此四個BYTE作為CHECK INDEX}

    if found byte shift (missing) then
      set have bug flag
      put information to the bug table
      {諸如仰角、計算壓縮時發生錯誤BLOCK及BYTE位置 . . . 等}

    update all pointer
  End loop

- Close input file

- if bug table not empty then
  Open input and output file
  {input 為解壓縮錯誤檔, output 為回復檔}
  Read/Writes file header
  {HEADER 不壓縮}

P   Initial all index and pointer variable
A   Loop
S   read whole elevation data into input buffer
    {以一個仰角資料放入暫存區作為搬移單位,
    因各仰角皆以BLOCK為單位結束}

    check bug table {由 PASS ONE 所建立}

    if found then {如果該仰角資料有誤}
      shift input buffer data to the output buffer
      put no data value to the data missing position
      {修正位移位置並將漏失資料位置填入NO DATA 之資料值}
    else write input buffer to the output buffer

    write output buffer to the output file blocks address
    update all pointer
  End loop

  Close input and output file
- end
end
```

程式六. RECOVER_POLAR_FILE

Pseudocode

```
begin
- Open input file
  {解壓縮直角座標檔案}
  Read file header
  {讀取檔案資料頭相關資訊, 參表一}
```



```

P   Initial all index and pointer variable
A
S   Loop
S   read whole xyz volume header data
    {檢查各層資料頭位置之地址, 參圖三}
    examine each picture header ID
    {以每一定高層資料作為檢查單位並以 "PHERAWIS" 作為檢查碼}

    if found byte shift (missing) then
D   set have bug flag
N   put information to the bug table
E   {諸如該層之起始位置、計算發生錯誤 BLOCK 及 BYTE 位置...等}

    update all pointer
  End loop

- Close input file

- if bug table not empty then
  Open input and output file
  {input 為解壓縮錯誤檔, output 為回復檔}
  Read/Write file header
  {HEADER 不壓縮}

P   Initial all index and pointer variable
A
S   Loop
S   read each level picture data into input buffer
    {以一層之資料放入暫存區作為搬移單位, 因各層皆以 BLOCK
    為單位結束即各層資料可視為獨立}

    check bug table    {由 PASS ONE 所建立}

    if found then      {如果該層資料有誤}
      shift input buffer data to the output buffer
      put no data value to the data missing position
      {修正位移位置並將漏失資料位置填入 NO DATA 之資料值}
T   else write input buffer to the output buffer
W
O

    write output buffer to the output file blocks address
    update all pointer
  End loop

  Close input and output file

- end
end

```

程式七. RECOVER_XYZ_FILE

四、結果

圖四與圖五為民國 76 年 7 月 27 日亞力士颱風之水平與垂直最大回波分布圖，其中圖四.(1)、圖四.(2)分別為雷達原始資料更正前與更正後之定高面平面位置圖 (CAPPI)，如圖四.(1)所示。在圖面外圍有一不連續之環狀圈，此乃因資料登錄過程壓縮邏輯錯誤，導致重取 (RESTORE) 被壓縮之極座標資料時無法完全回復原來之資料 (部分資料將遺失)；又因資料係以連續之方式存放，以致自錯誤之資料點起，往後存放之資料都有一固定量之位移 (每個波束的資料同時往前移相同個存放位置，後續之仰角亦然)，所以會產生環狀之現象。圖四.(2)為資料經回復軟體 (程式六) 處理後之結果，

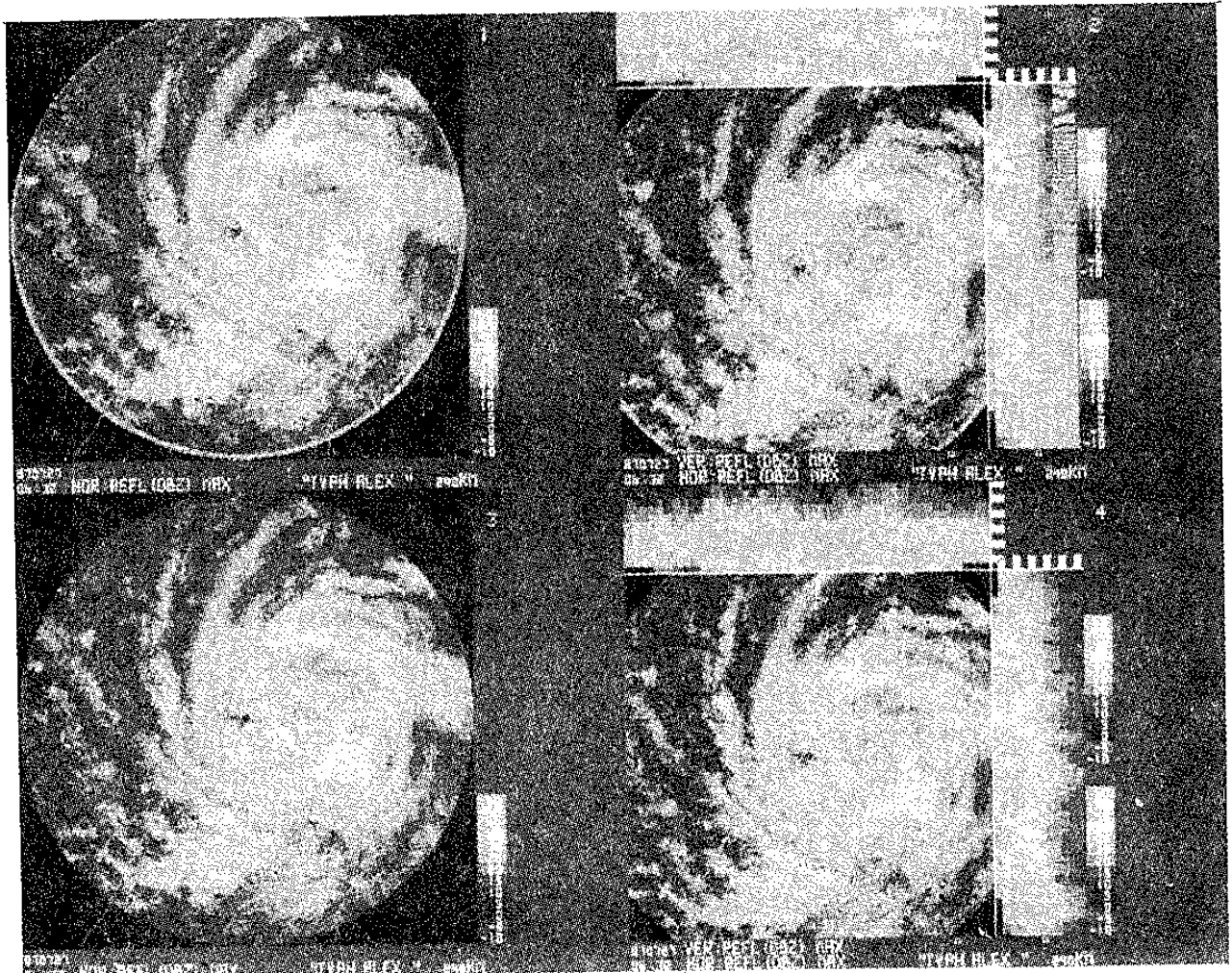
周圍環狀已不存在，且圖中每個相點 (pixel) 皆重返正確之位置。

圖五為垂直剖面最大回波圖，圖五.(1)顯示其回波之垂直分布極為反常，高度愈高顯示之回波值愈大，因垂直剖面最大回波圖係將該水平 (東西或南北) 方向之最大回波值投影在南北或東西向的剖面高度層上，反常現象發生之最低高度則視被壓縮之極座標資料產生錯誤之仰角與方位角而定。如圖五.(1)最低高度層之錯誤發生在仰角 1.5° 時之第 309 個方位角 ($309 \times 0.86^\circ = 265.7^\circ$) 的第 120 RANGE GATE，存放此 RANGE GATE 之資料位置將被下一個方位角之時間差資料值所填補，即後續資料均往前位移一個位元組，而此 RANGE GATE 之水平高度為 3.99 公里，因此反常現象的最低高度出

現在 3.5公里之剖面上。圖五.(2)為雷達資料經修正後之圖形，圖中可清楚的分辨出回波強度之垂直分布，亦可發現眼牆及雲牆之垂直發展較高。

若直角座標之原始資料因被壓縮而造成錯誤，則當重取被壓縮之資料時，錯誤高度層以上之各層將無法製

作 CAPPI 圖（該各高度層檔案頭之識別碼 "PHERANIS" 因位移而發生錯誤），當然也不能製成剖面圖。若將此檔案經由程式七除錯，則檔案頭之識別碼及各相關資料將重新歸位而得一正確之直角座標。



圖四.(1) CAPPI 最大回波強度分布圖 (修正前)
 圖四.(2) CAPPI 最大回波強度分布圖 (修正後)

圖五.(1) 垂直剖面最大回波強度分布圖 (修正前)
 圖五.(2) 垂直剖面最大回波強度分布圖 (修正後)