UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Unidad Académica de los Ciclos Profesional y de Posgrado del Colegio de Ciencias y Humanidades

PRONOSTICO METEOROLOGICO ESTADISTICO DE TEMPERATURA MINIMA EN EL VALLE DE MEXICO

T E S I S

Para obtener el Grado de
MAESTRA EN CIENCIAS DE LA COMPUTACION

p r e s e n t a

ISABEL (Irene) QUINTAS Pereira

Asesor de Tesis: M. en I. Jorge Sánchez Sesma

México, D. F. 1995

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

Pronóstico meteorológico estadístico

de temperatura mínima en el Valle de México

Isabel Quintas

ESTE TRABAJO FUE REALIZADO GRACIAS AL APOYO DEL **INSTITUTO** MEXICANO DE TECNOLOGIA DEL AGUA (IMTA) **Y A LA UNIVERSIDAD** AUTONOMA METROPOLITANA (UAM)

INDICE

Introducción

El Servicio Meteorológico Nacional (SMN), necesita de herramientas numéricas y gráficas que permitan la respuesta oportuna a las demandas de información sobre el estado del tiempo.

Como demandas de información meteorológica asociadas, se consideran, tanto los pronósticos como las mediciones y estimaciones, a tiempo real, de variables meteorológicas.

Para el pronóstico meteorológico, el SMN cuenta con un modelo dinámico barotrópico que depende fuertemente de las condiciones iniciales y de frontera, y que no ha sido evaluado ni utilizado sistemáticamente. Aunque la precisión de estos métodos ha mejorado en forma paralela a las capacidades de cálculo numérico de las computadoras, en México existen fuertes limitaciones debido a dos aspectos: la falta de monitoreo en los niveles superiores y la distorsión ocasionada por la orografía a los flujos atmosféricos.

Para cubrir parte de estas necesidades se ha comenzado a instalar una red de monitoreo automático que cuenta con 600 estaciones en superficie, doce radares meteorológicos digitales y diez estaciones de radiosondeo, cuya información llega de forma automática al SMN.

Por otro lado, el SMN recibe cada doce horas información del National Meteorological Center de los Estados Unidos de América (NMC de EUA), que incluye datos analizados y pronosticados a 12 horas, 24 horas y hasta 5 días, obtenidos con diferentes modelos. Actualmente esta información se recibe de diferentes maneras, no todas en forma digital; por ejemplo llegan mapas con isolíneas, imágenes de satélite, datos numéricos.

Tanto la información analizada como la pronosticada, correspondiente a México es de gran escala, y debe realizarse algún tipo de procedimiento, que permita a partir de esos datos, obtener la información local.

En este contexto es que surge la necesidad de este trabajo. Sería deseable contar con un sistema al que le lleguen, en tiempo casi real, tanto los datos analizados, como los de salida de los modelos del NMC, y los datos monitoreados sobre el territorio mexicano, que ayudara al personal del SMN obtener estimaciones más reales, para realizar el pronóstico meteorológico.

Este sistema estaría constituido por varios módulos: uno de adquisición de datos, un segundo de comunicación con el usuario, un núcleo con los modelos matemáticos de cada una de las variables a pronosticar, y un último

técnicas que se pueden emplear para el modelado de las variables regionales, se encuentran las técnicas estadísticas.

En el presente trabajo se plantea la exploración y evaluación de algunas técnicas estadísticas para mejorar el pronóstico de variables meteorológicas, resultado de los modelos dinámicos que se reciben del National Meteorological Center de los Estados Unidos, y como caso concreto, las funciones que modelan las temperaturas mínimas diarias en la ciudad de México.

Se prueban una serie de alternativas, registradas en la bibliografía, y variaciones de ellas, en la búsqueda de la depuración de una técnica que permita llegar al *mejor modelo* operativo.

En el capítulo uno se presenta el problema, los antecedentes encontrados en la bibliografía, las técnicas utilizadas en el pronóstico estadístico, y las fuentes de información o datos meteorológicos, que representan el mayor problema en este tipo de estudios.

En el capítulo dos se presentan los conceptos matemáticos y estadísticos utilizados en las diferentes técnicas empleadas para realizar los modelos de pronóstico.

En el capítulo tres se detallan los pasos realizados para obtener diferentes modelos. Se explica como se obtiene la muestra de datos y se buscan patrones de temperatura en la región. Se utilizan la técnica de regresión múltiple sobre los datos de la muestra, para obtener un primer modelo, para luego aplicar las técnicas de agrupamiento y el análisis de componentes principales y obtener así modelos más aproximados. Por último se utilizan los procedimientos que dieron mejores resultados, para un segundo caso de estudio, incluyendo un número mayor de variables y mayor covertura geográfica.

El siguiente capítulo toca los detalles computacionales y algunos de los problemas encontrados, ya que hubo que utilizar los recursos disponibles, muy limitados para la envergadura del problema a estudiar.

En el capítulo cinco se hace el análisis y evaluación de los modelos obtenidos, se señalan las deficiencias encontradas. Se dan ciertas recomendaciones generales para llegar a los modelos operativos requeridos por el Servicio Meteorológico Nacional.

Por último, en los apéndices se encuentran los códigos de los módulos utilizados en las diferentes etapas de construcción de los modelos. Hubo variaciones para algunas alternativas, ya que durante el proceso de modelar se fueron haciendo modificaciones sobre la marcha. Los códigos presentados son los utilizaron para el primer caso.

1 Pronóstico meteorológico estadístico

Existen, fundamentalmente, dos tipos de métodos para realizar el pronóstico meteorológico: dinámico y estadístico.

Los métodos dinámicos, al integrar las ecuaciones de movimiento en el tiempo, a partir de ciertas condiciones iniciales, permiten estimar a 24, 48 y 72 horas las configuraciones de las variables meteorológicas. En general la precisión de dichos métodos ha mejorado en las últimas décadas, sin embargo, para México existen fuertes limitaciones debido a dos aspectos: la falta de monitoreo de niveles superiores en la región, y la distorsión que le imponen las sierras a los flujos atmosféricos.

Por otra parte, los métodos estadísticos permiten, a partir de relaciones encontradas entre las variables medidas o pronosticadas con métodos dinámicos, estimar los valores futuros de alguna variable de interés. Aunque estas técnicas tambien presentan limitaciones, existen planteamientos que no han sido probados y que podrían ser útiles para el pronóstico en ciertas regiones de México.

Se describen, a continuación, las bases de la técnica utilizada para el pronóstico estadístico de variables meteorológicas en México.

Planteamiento del problema

Las técnicas estadísticas permiten obtener reglas, por regresión lineal u otros métodos, para el pronóstico de variables meteorológicas a partir de *configuraciones meteorológicas* previas.

No obstante, para definir las configuraciones o situaciones meteorológicas se requiere de numerosos datos, pues cada configuración está constituidas por campos de diferentes variables (viento, presión, temperatura y humedad), y sus variaciones en el tiempo.

Las condiciones atmosféricas en zonas tropicales son caracterizadas básicamente por flujos, a diferencia de lo que acontece en latitudes medias y altas donde esto se da por medio de la distribución de presiones. Por ello, para caracterizar las condiciones meteorológicas sinópticas, es decir, a escalas de miles de kilómetros, y especialmente durante el verano, se deberían utilizar los flujos en superficie y en altura (850 y 200 mb). Para estos datos, se cuenta con los calculados por el Centro Nacional de Huracanes (CNH), de los EUA.

Otro aspecto fundamental en el pronóstico meteorológico estadístico, es el definir la configuración *media* de los sistemas meteorológicos. Esta configuración nos permitirá definir las anomalías de una manera más *dinámica*, como variaciones alrededor de una configuración más estable, entorno de la cual puede oscilar.

En este trabajo, se modelaron dos situaciones diferentes. Como primera aproximación, se utilizaron como variables predictoras, solamente las temperaturas a 850 mb, su variación en los dos días consecutivos previos, considerando el cambio respecto a la media en toda el área, para esos dos días. En el segundo caso, se consideraron como variables predictoras, las temperaturas de los dos días previos a 850 mb y las componentes de viento del día anterior, también a 850 mb.

Técnicas estadísticas utilizadas en meteorología

Existen varias técnicas estadísticas utilizadas en pronóstico meteorológico, aunque la que más se usa es la de regresión lineal múltiple. Otras tratan de clasificar los eventos según cierto grado de similitud, o búsqueda de patrones característicos entre las configuraciones probables, de algunas variables.

Algunos métodos utilizan el análisis de componentes principales, o también llamadas, funciones empíricas ortogonales (EOF). Básicamente, se trata de describir las configuraciones utilizando los vectores característicos de la matriz de coverianza de los datos, para luego buscar las funciones de regresión entre la variable a predecir y las primeras EOF.

En la literatura se menciona una clasificación de las técnicas de regresión múltiple, dependiendo del origen de las variables. Dividen a los procedimientos estadísticos utilizados [Rouss83], [Mozer93], [Glahn72],[Klein59] en los de *pronóstico perfecto* (perfect prog) y en los *modelos estadísticos sobre la salida (MOS*, Model Output Statistics).

Las técnicas de pronóstico perfecto son aquellas en que se busca una relación entre las variables locales a predecir y los valores *observados* de las variables meteorológicas predictoras; pero como casi no se dispone de valores observados, se utilizan los valores calculados por los modelos numéricos globales de análisis, de datos observados, como variables predictoras.

Para el caso de los modelos MOS, *la relación estadística se establece usando las salidas de algún modelo de pronóstico global y de gran escala, como variables predictores* [Rouss83], incorporándole información local.

Otras técnicas utilizadas, pero que no necesariamente están asociada a regresión lineal múltiple, son la búsqueda de análogos y las técnicas de agrupamiento (cluster). Básicamente se trata de analizar la salida de los

modelos numéricos, buscando los caso más parecidos ocurridos en el pasado y esperar un comportamiento similar. Se reportan diferentes procedimientos utilizados para buscar estas analogías [Kruiz81] y [Mozer93].

Todas ellas requieren la definición o búsqueda dentro de un universo factible, de las variables base del pronóstico o variables predictoras. Es aquí donde interviene el conocimiento físico del problema, para la selección de las variables apropiadas.

Antecedentes

En la bibliografía especializada se encuentran artículos donde se utilizan diversas técnicas estadísticas en el pronóstico meteorológico o para interpretar y mejorar los datos obtenidos de los sistemas globales de pronóstico y análisis, que sólo dan información a gran escala, y así obtener pronósticos locales.

Básicamente todas estas técnicas utilizan modelos de correlación, variando en la forma en que seleccionan sus datos iniciales: algunos utilizan un solo tipo de variable, otros utilizan distintas variables o campos simultaneamente; los datos pueden haber sido obtenidos por observaciones directas, de superficie y altura, o ser los datos previamente analizados por un sistema global; o una combinación de ambos.

Algunos autores utilizan los datos como variables de las ecuaciones de regresión múltiple, mientras que otros primero buscan los componentes principales para luego buscar los coeficientes de los modelos de regresión u otras técnicas de análisis de semejanzas.

Por ejemplo, ya en 1959 , Klein [Klein59], utilizó un método mixto, dinámico y estadístico para realizar pronóstico de temperatura a cinco días en cierto número de ciudades de los Estados Unidos.

Para cada ciudad realizó los mapas de correlación entre la temperatura de superficie y la circulación de los niveles superiores, considerando la altura barométrica de 700 mb sobre una malla de 70 puntos, para los cuatro días anteriores.

Realiza regresión lineal simple entre la temperatura en la superficie y la altura barométrica del punto más correlacionado. Utiliza la técnica de paso a paso para incorporar nuevas variables predictoras, calculando la correlación con cada uno de los puntos de la malla. Como sólo disponían de una calculadora de mesa, el autor comenta que no podía introducir un gran número de variables en la ecuación de regresión.

Pero utilizando la altura barométrica, sólo pudo explicarse una parte de la varianza de la temperatura superficial. Deben incorporarse otros parámetros para mejorar los resultados, para lo que incorpora la tempertura media local. El artículo dice que se obtuvo la ecuación de regresión para cada ciudad logrando mejorar los pronósticos globales.

El artículo de Rousseau [Rouss83] hace una descripción de los trabajos realizados en Francia, para adaptar los pronósticos locales con técnicas estadísticas, a partir de los datos globales. Reseña los pronósticos de temperaturas máximas y mínimas, ocurrencia de precipitación a 12 horas y cantidad de precipitación en 24 horas.

Poseen una base de datos importantísima con los archivos de salida del modelo de pronóstico del Centro Europeo de Pronóstico Meteorológico Mundial, ECMWF, y los archivos de las observaciones sinópticas de la red meteorológica francesa. Utilizaron datos sobre una malla de 10 por 10 puntos separados unos 380 Km y durante períodos de 15 años.

En el caso de la determinación de las temperaturas máximas y mínimas, (aquí se calcularán las temperaturas mínimas) utilizaron como variables predictoras los campos de altura barométrica, temperatura, vientos y algunas variables derivadas como vorticidades y advección de estos campos a varios niveles, totalizando 4800 variables predictoras.

Para cada uno de estos campos, con técnicas de componentes principales se reducen a los 10 primeros coeficientes. Luego, menciona que calculan una variable canónica por campo, le incorpora una variable de persistencia, para finalmente buscar los diez campos más representativos, por el procedimiento de regresión lineal múltiple por pasos.

El autor refiere que se determinaron las funciones correspondientes a 72 localidades obteniendose, para temperaturas, un error medio no superior a 1.8°C.

Mozer y Zehnder [Mozer93] utilizan técnicas de agrupamiento y análogos para estudiar la formación de ciclones en el Pacífico norte. Su interés se centra en tratar de identificar las circulaciones a gran escala en la región asociadas a las perturbaciones iniciales, que provocan la formación de ciclones.

Utiliza las técnicas de agrupamiento para diferenciar entre grupos que presentan patrones homogeneos en las distribuciones de temperatura, vientos y altura barométrica cercanas a la región de génesis, porque a diferencia de otras técnicas multivariadas de agrupamiento, no se requiere conocer a priori las características de los grupos a formar.

conocer a priori las características de los grupos a formar.

Para filtrar el ruido en los datos, que pudieran esconder a las anomalías que interesa detectar, se les quitó el promedio estacional, y sólo se consideraron los datos posteriores a 1985, ya que en dicho año hubo cambios en la forma de inicialización de los datos, utilizada para el análisis global de datos (ECMWF). Para reducir el número de variables, primero realizan un análisis de componentes principales.

Weare [Weare86] por su parte, propone agregarle a la técnica de regresión lineal múltiple, restricciones adicionales que toman en cuenta las interrelaciones espaciales de las variables (dependencias) que se manifiestan también en el dominio de las funciones empíricas ortogonales. La mejoría es modesta pero logra para los casos tratados (dos) funciones de regresión que dependen de solamente uno o dos de los primeros componentes principales.

El trabajo que deberíamos realizar aquí, es similar al mencionado por Rousseau, pero en México tenemos limitaciones muy severas: no se cuenta con bases de datos tan completa; actualmente nuestra fuente de información más confiable es el CD-ROM METEO, ya mencionado. Las series de tiempo de datos observados en estaciones meteorológicas, cubren pocos años, los últimos, y algunas no están capturadas digitalmente.

La otra limitación al presente trabajo fueron los recursos de cómputo disponibles: se debía realizar en una computadora PC, y no se disponía de una librería matemática poderosa, limitándose al software público que se consiguiera.

Información base para el pronóstico

La información meteorológica necesaria para la aplicación del método debe de cubrir, en el espacio, a México y sus alrededores, tanto en datos de superficie como en altura y, en el tiempo, debería de abarcar más de 10 años para ser una muestra representativa confiable.

Para la realización del primer caso se utilizó la información disponible en ese momento, que era la contenida en el CD-ROM denominado CD-METEO. En éste CD-ROM se cuenta con datos de presiones, alturas barométricas y temperaturas a diferentes niveles, colectados y procesados por el National Meteorological Center (NMC) de los EUA.

En cuanto a datos medidos, se obtuvieron del Observatorio Meteorológico de Tacubaya, las series de tiempo de las temperaturas mínimas observadas, correspondientes a los meses de invierno (diciembre, enero y febrero) desde 1970 a 1982.

Meteorological, Center Grid Point Set, version 2, que contiene información posterior a 1962. Se utilizaron los datos del período comprendido entre 1978 a 1989.

Estimar la temperatura mínima en la Cd. de México es un primer paso en el proceso de pronósticar condiciones favorables a la concentración de contaminantes atmosféricos en esa ciudad, ya que una de las variables más importantes, es la temperatura ambiente.

Procedimiento empleado

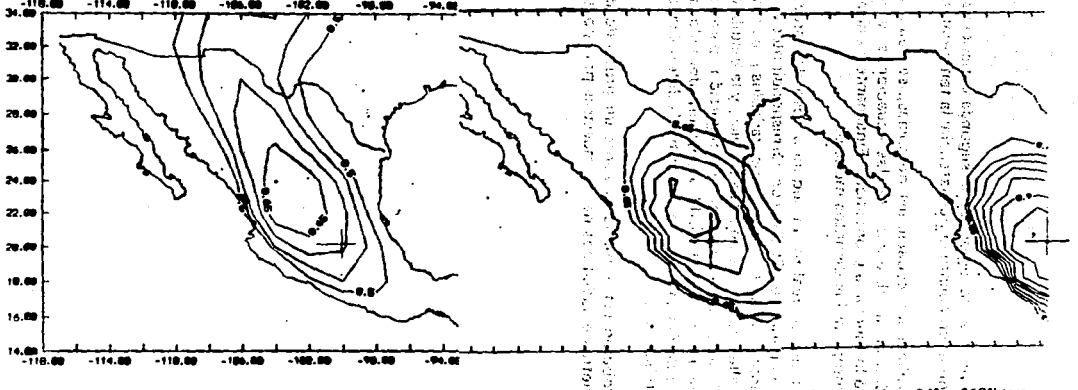
Como la cantidad de información disponible era muy grande y era necesario acotar el problema por limitaciones del equipo disponible y del software, se realizaron algunos estudios preliminares. Una de las pruebas consistió en correlacionar los valores de temperatura del punto a pronosticar, la ciudad de México, con las temperaturas de los puntos sobre una malla de dos grados, que cubriera todo el territorio nacional. Después de obtener los valores, se dibujaron las líneas correspondientes a la correlaciones del día a pronosticar y de los tres días anteriores.

De estas gráficas, inmediatamente se pudo observar que la temperatura de la ciudad de México depende especialmente de la temperatura sobre la parte continental, y se nota un desplazamiento de norte a sur , disminuyendo el coeficiente de correlación máximo, a medida que los días estan más separados del día a pronosticar. Estas observaciones permitieron la selección de un número reducido de puntos, localizados al norte de punto a pronosticar, y sobre la parte continental.

Siguiendo una combinación de los métodos mencionados en la literatura y las observaciones anteriores, se procesaron más de 360 casos o configuraciones térmicas de la zona norte del país, para pronósticar la temperatura mínima en invierno en la ciudad de México. Estas muestras corresponden a los días invernales de los años 1970 a 1976. Para cerciorarse de la validez del procedimiento, se utilizó otro conjunto de configuraciones correspondientes al período 1983-1989.

Se obtuvieron las anomalías, esto es, la variación de la temperatura respecto a la media de los dos días previos, en todos y cada uno de los casos, después se calcularon los vectores y valores característicos correspondientes. Se calcularon las contribuciones lineales de cada uno de los vectores propios en cada uno de los casos. Se agruparon los casos utilizando las técnicas conocidas como de "cluster". Para algunos de esos grupos, los más significativos, se calcularon las ecuaciones de regresión correspondientes.

Se analizaron diferentes alternativas, en la primera sólo se incluyeron como variables predictoras, la información correspondiente a las anomalías de



Coeficiente de correlación entre la temperatura del DF, y las temperaturas sobre la toda la región, (latitud de 14° a 34° y longitud de -94° a -118°) tres días antes, un día antes y el día que se uiere pronosticar.

temperatura en la región. En la segunda variante caso se agregó una variable para tomar en cuenta la persistencia: la temperatura mínima del día anterior.

Para validar las ecuaciones obtenidas, se tomó otro grupo de datos, no incluidos en la obtención de las ecuaciones, y con cada una de ellas se calculó la función a pronosticar. Finalmente se hizo un análisis de los errores para comparar los distintos modelos.

Por último, se realizó un segundo caso de estudio, aplicando las mismas técnicas para unos 600 casos correspondientes al período de 1978 a 1989. Para este caso se tomaron como variables predictoras la temperatura, sobre una malla regular, los dos días previos, y la distribución de vientos a 850 mb. Se calcularon las ecuaciones de regresión, para pronosticar la temperatura mínima en la ciudad de México, considerando también a la temperatura mínima del día anterior, ya que se había determinado la importancia del efecto de persistencia.

Se probaron estos modelos sobre un grupo de datos y se calcularon los parámetros estadísticos de las muestras para poder comparar resultados.

2 Conceptos matemáticos y estadísticos

Estructura de la información

Al igual que en otras ciencias, la información necesaria para describir a una situación meteorológica, está compuesta por varias variables, o se puede hablar de *datos multivariados*.

Una configuración meteorológica puede estar dada por los valores de parámetros como la temperatura, humedad, presión en distintos puntos y su evolución en el tiempo.

Para estudiar los fenómenos es necesario no solo conocer sus valores, sino que puede interesar, como se relacionan estas variables entre si en situaciones similares, que permitan identificar a las configuraciones en estudio, de cierta manera. O sea, se trata de buscar una estructura de estos datos multivariados.

Algunas técnicas de análisis multivariado pueden ayudar a encontrar esta estructura que caracterice situaciones particulares, dentro de un conjunto grande de muestras, de muchas variables cada una. *El análisis de componentes principales busca un número mínimo de variables, combinación lineal de las variables originales, que puedan explicar las variaciones de la muestra, pues contienen casi la misma información.*

Definiciones

En esta sección se darán algunas definiciones y la notación a utilizar en el resto del texto.

Vector dato

El vector de datos es el conjunto de valores que adquieren cada una de las variables que definen una situación o entidad o caso de estudio. Los vectores de datos serán vectores fila y se representan como

$$\mathbf{x}' = (x_i) = [x_1, x_2, x_3]$$

Matriz de la muestra

Un conjunto de casos conforman una matriz de la muestra, donde cada fila corresponde a un dato, o situación meteorológica, y cada columna corresponde a una variable.

$$\mathbf{X} = (x_{i,j})$$

La dimensión de la matriz es de m x p donde m es el número de filas o casos de estudio y p es el número de variables de cada vector dato.

Angulo entre dos vectores

El coseno del ángulo entre dos vectores u y v , se obtiene del producto interno entre vectores

$$\cos\theta = \frac{u'v}{(u'u)^{1/2}(v'v)^{1/2}}$$

Producto de matrices

En el análisis multivariado, es importante el producto de matrices con su transpuesta. A estos productos se los llama [Reyme93] *momentos* mayor y menor de la matriz.

El *momento mayor* se define como el producto de la matriz posmultiplicada por su transpuesta. Si **X** es una matriz de m por p, su momento mayor es una matriz cuadrada y simétrica de orden m por m.

$$\mathbf{A} = \mathbf{XX'}$$

donde cada elemento a_{ij} de la matriz es el producto interno de los renglones (casos) i y j

$$a_{ij} = \sum_{j}^{p} x_{ij}x_{ij}$$

El momento menor se define como la premultiplicación de la matriz por la transpuesta.

$$\mathbf{B} = \mathbf{X'X}$$

En este caso, la matriz resultante es de orden p por p, el número de variables. Cada elemento a_{ij} de la matriz es el producto interno de las columnas (variables) i y j

$$b_{ij} = \sum_{j}^{m} x_{ji}x_{jj}$$

Parámetros estadísticos en notación matricial

Para poder desarrollar el análisis factorial formalmente, se presentan las ecuaciones básicas de la estadística descriptiva en notación matricial.

Valor medio

El valor medio de los elementos de cada columna de una matriz X de n por p es

$$\bar{x} = l'X / l'l$$

donde l es la matriz unitaria de orden p.

Desviaciones

La diferencia entre el valor de la variable y el valor medio es la desviación o anomalía de la variable. El valor medio de las desviaciones es, por supuesto, cero.

$$y_{ij} = x_{ij} - \bar{x}_j$$

Varianza

La varianza es una medida de la dispersión de los valores individuales alrededor de la media. Para la variable j, y utilizando las desviaciones

$$s_j^2 = \sum_n^N y_{ij}^2 / N$$

que en notación matricial es

$$s_j^2 = y'_j y_j / l'l$$

donde l es la matriz unitaria.

Covarianza

La covarianza expresa la relación entre dos variables, y está dada por la suma promedio de sus productos. La varianza antes definida es un caso particular de la covarianza. En el caso de una matriz de datos, donde cada columna corresponde a una variable, y los renglones a las muestras, la matriz de covarianza, está formada por el producto de las columnas o monento menor, expresado en término de las desviaciones.

característica, que tendrá k raíces o valores propios.

$$\lambda_1, \lambda_2, \ldots\ldots, \lambda_k$$

Si la matriz R es real y simétrica, las k raíces serán siempre reales, aunque no necesariamente todas sean diferentes. Incluso puede haber valores iguales a cero.

Obtenidos los valores propios se calcular los vectores propios; la solución no es única, ya que si *v* es solución, tambien lo es *cv*. Por convención se eligen los vectores normalizados.

Asociado con cada valor propio hay un vector propio, ortogonales entre sí. En el caso de las raíces múltiples, se elije el vector ortogonal a los demás.

Si se forma una matriz diagonal L con los valores propios, y se acomodan los vectores propios como columnas de la matriz U, entonces

$$R\,U = U\,L$$

Pero como U es cuadrada y ortonormal, $U'U = UU' = I$ entonces, posmultiplicando

$$R = U\,L\,U'$$

entonces R puede expresarse en términos de sus valores y vectores propios

$$R = \lambda_1 u_1 u_1' + \ldots + \lambda_p u_p u_p'$$

o pre y pos multiplicando la matriz R se muestra que U es la matriz que reduce a R a la forma diagonal

$$L = U'\,R\,U$$

El conjunto de vectores propios asociados a los valores característicos distintos de cero forman una base ortonormal que expanden el espacio de los vectores de R.

Una interpretación geométrica de los valores y vectores propios, vp, de la matriz de correlación permite decir que:

1) La dirección del vector propio asociado al mayor vp determina la dirección de máxima varianza de los vectores dato. El vector asociado con el segundo vp, da la dirección de mayor varianza ortogonal a la primera.

2) Los vectores propios son vectores linealmente independientes, combinación lineal de las variables originales. Pueden ser vistos como nuevas variables, no

Parámetros estadísticos en notación matricial

Para poder desarrollar el análisis factorial formalmente, se presentan las ecuaciones básicas de la estadística descriptiva en notación matricial.

Valor medio

El valor medio de los elementos de cada columna de una matriz X de n por p es

$$\bar{x} = l'X / l'l$$

donde l es la matriz unitaria de orden p.

Desviaciones

La diferencia entre el valor de la variable y el valor medio es la desviación o anomalía de la variable. El valor medio de las desviaciones es, por supuesto, cero.

$$y_{ij} = x_{ij} - \bar{x}_j$$

Varianza

La varianza es una medida de la dispersión de los valores individuales alrededor de la media. Para la variable j, y utilizando las desviaciones

$$s_j^2 = \sum_{n}^{N} y_{ij}^2 / N$$

que en notación matricial es

$$s_j^2 = y'_j y_j / l'l$$

donde l es la matriz unitaria.

Covarianza

La covarianza expresa la relación entre dos variables, y está dada por la suma promedio de sus productos. La varianza antes definida es un caso particular de la covarianza. En el caso de una matriz de datos, donde cada columna corresponde a una variable, y los renglones a las muestras, la matriz de covarianza, está formada por el producto de las columnas o monento menor, expresado en término de las desviaciones.

$$S = Y'Y / I'I$$

Se trata de una matriz simétrica, donde los elementos de la diagonal principal corresponden a las varianzas de cada variable.

Desviación estandar

Se define como la raiz cuadrada de la varianza

$$s_j = (y'y)^{-1/2} N^{-1/2} = |y| N^{-1/2}$$

Valores normalizados

Las variables normalizadas tienen valor medio cero y desviación estandar unitaria, y el módulo de los vectores de las variables es $N^{1/2}$. La expresión para estandarizar la matriz de datos es

$$Z = Y.D^{-1/2}$$

donde D corresponde a una matiz formada por los elementos de la diagonal de la matriz de covarianza S, y Y es la matriz de las desviaciones.

Correlación

El coeficiente de correlación de Pearson entre dos variables, se define como el cociente de la covarianza de las variables entre el producto de sus desviaciones estandar.

$$r_{ij} = s_{ij} / s_i s_j$$

Y la matriz de correlación se representa como

$$R = Z'Z / N$$

Es una matriz cuadrada y simétrica en la cual cada elemento se obtiene del momento menor de la matriz de las variables normalizadas, dividido por el tamaño de la muestra. Otra manera de expresar al coeficiente de correlación, y que explicita su interpretación geométrica es

$$r_{ij} = z_i z_j / N = \cos\theta$$

que indica que la correlación entre las variables está dada por el ángulo formado por

los vectores de las variables normalizadas en el espacio de las muestras.

Rango de una matriz

Toda matriz $X_{(N \times p)}$, puede descomponerse en el producto de dos factores $A_{(N \times s)}$ y $B_{(s \times p)}$. Una forma de definir el rango de la matriz en función de sus componentes [Reyme93] es el menor orden s, filas o columnas, de las matrices cuyo producto da la matriz deseada. Por lo tanto el orden de una matriz no puede ser mayor que su dimensión más pequeña.

El rango de una matriz de datos no puede ser mayor que el número de sus filas (datos) y sus columnas (variables). Entonces el rango de la matriz momento derivada, no puede exceder la dimensión menor de la matriz de datos.

Una matriz básica es aquella en que el rango es igual a su orden menor y por lo tanto no puede ser expresada como producto de matrices de menor orden.

Valores y vectores propios

Los vectores propios de una matriz cuadrada forman una base del espacio de las columnas de la matriz, y por lo tanto el número de vectores es igual al rango de la matriz. Esta base posee características especiales, cuya interpretación depende del campo de aplicación. En particular, interesa discutir el caso de los valores y vectores propios de las matrices simétricas obtenidas como producto menor de una matriz de datos.

Un vector propio de la matriz R, es un vector v, tal que el producto de la matriz por el vector es proporcional al vector; la proporcionalidad esta dada por el escalar lambda:

$$R\,v = v\lambda$$

o sea

$$(R - \lambda I)v = 0$$

Esto implica que el vector v es ortogonal a todas las filas de la matriz $(R - I)$, y como no es el vector nulo, para que el sistema tenga solución, el determinante debe ser igual a cero:

$$|R - \lambda v| = 0$$

Esto no lleva a una ecuación polinómica, de grado k en lambda, llamada ecuación

característica, que tendrá k raices o valores propios.

$$\lambda_1, \lambda_2, \ldots\ldots, \lambda_k$$

Si la matriz R es real y simétrica, las k raices serán siempre reales, aunque no necesariamente todas sean diferentes. Incluso puede haber valores iguales a cero.

Obtenidos los valores propios se calcular los vectores propios; la solución no es única, ya que si *v* es solución, tambien lo es cv. Por convención se eligen los vectores normalizados.

Asociado con cada valor propio hay un vector propio, ortogonales entre sí. En el caso de las raices múltiples, se elije el vector ortogonal a los demás.

Si se forma una matriz diagonal L con los valores propios, y se acomodan los vectores propios como columnas de la matriz U, entonces

$$R\,U = U\,L$$

Pero como U es cuadrada y ortonormal, $U'U = UU' = I$ entonces, posmultiplicando

$$R = U\,L\,U'$$

entonces R puede expresarse en términos de sus valores y vectores propios

$$R = \lambda_1 u_1 u_1' + \ldots + \lambda_p u_p u_p'$$

o pre y pos multiplicando la matriz R se muestra que U es la matriz que reduce a R a la forma diagonal

$$L = U'\,R\,U$$

El conjunto de vectores propios asociados a los valores característicos distintos de cero forman una base ortonormal que expanden el espacio de los vectores de R.

Una interpretación geométrica de los valores y vectores propios, vp, de la matriz de correlación permite decir que:

1) La dirección del vector propio asociado al mayor vp determina la dirección de máxima varianza de los vectores dato. El vector asociado con el segundo vp, da la dirección de mayor varianza ortogonal a la primera.

2) Los vectores propios son vectores linealmente independientes, combinación lineal de las variables originales. Pueden ser vistos como nuevas variables, no

correlacionadas y que toman cuenta de la varianza en los datos en forma decreciente.

3) La suma de los cuadrados de las proyecciones de los vectores dato sobre los vectores propios, es proporcional a la varianza en esa dirección. Esta varianza es igual al vp asociado y por lo tanto, la raiz cuadrada del vp puede ser usada como la desviación estandar de la nueva variable, el vector propio.

Si R puede expresarse como combinación linéal vectores, dicha matriz puede aproximarse utilizando los k primeros terminos, con $k<p$. La bondad de la aproximación estará dada por

$$\frac{\lambda_{k+1}+.....+\lambda_r}{\lambda_1+\lambda_2+.....+\lambda_r}$$

Análisis de componentes principales

El análisis de componentes principales, que consiste básicamente en reducir la dimensión de un conjunto de datos que incluyen un gran número de variables, reteniendo la mayor cantidad de información posible. Para esto realiza transformaciones de la base del espacio de las variables, para obtener una base ortonormal, de tal manera que el primer eje se oriente en la dirección de máxima variación de las variables, el segundo, sea normal al primero, o no correlacionado, y en la dirección de máxima variación restante, etc. De tal manera que los datos u objetos puedan ser descritos con una buena aproximación, tomando en cuenta solo las primeras componentes.

Sea x un vector de p variables aleatorias, se trata de encontrar una función lineal de los elementos de x que tenga máxima varianza $a'1x$, donde a_1 es un vector de p componentes

$$a_1.x = \sum_{j=1}^{p} a_{1j}\, x_j$$

Luego se busca una función a'_2x no correlacionada con a'_1x que tenga la máxima varianza no considerada aún. Y así se obtienen p funciones lineales no correlacionadas, correspondientes a los p componentes principales. Por la forma en que son elejidos, los m primeros componentes principales toman en cuenta la mayor parte de la varianza de los datos, para $m << p$.

Para encontrar los componentes principales se construye la matriz de covarianza Σ de las variables x. El k-ésimo componente principal es $z_k = a'_kx$ donde a_k es el k-

ésimo vector propio de Σ correspondiente al k-ésimo valor propio más grande. La varianza de z_k es el valor propio correspondiente.

Para hallar los componentes principales se trata primero de maximizar la varianza de $a'_1\mathbf{x}$

$$\max \text{var}[a'_1\mathbf{x}] = a'_1\Sigma a_1 \qquad \text{donde } \Sigma \text{ es la covarianza de } \mathbf{x}.$$

para poder obtener un resultado es necesario imponer una restricción de normalización $a'_1a_1 = 1$, y utilizando los multiplicadores de Lagrange para maximizar

$$a'_1\Sigma a_1 - \lambda (a'_1a_1 - 1)$$

derivando respecto de a se obtiene

$$(\Sigma - \lambda I_p)a_1 = 0$$

donde λ es un valor propio de Σ y a_1 es el vector propio correspondiente. Para decidir cual de los p vectores propios es el que maximiza, partiendo de la expresión original

$$a'_1\Sigma a_1 = a'_1\lambda a_1 = \lambda a'_1 a_1 = \lambda$$

entonces λ debe ser el mayor posible y a_1 entonces es el vector propio correspondiente. El k-ésimo componente principal $a'k\mathbf{x}$, entonces, será el asociado con el k-ésimo vector propio de la matriz de covarianza.

Para obtener el segundo componente principal $a_2\mathbf{x}$ que maximiza $a'_2\Sigma a_2$ pero no correlacionado con $a_1\mathbf{x}$ o sea que $\text{cov}[a_1\mathbf{x}, a_2\mathbf{x}] = 0$

$$\text{cov}[a_1\mathbf{x}, a_2\mathbf{x}] = a'_1\Sigma a_2 = a'_2\lambda a_1 = \lambda a'_2 a_1 = \lambda$$

ahora minimizando, sujeto a las dos restricciones, con dos coeficientes de Lagrange, λ y μ

$$a'_2\Sigma a_2 - \lambda (a'_2a_2 - 1) - \mu a'_2 a_1$$

y derivando respecto de a_2

$$\Sigma a_2 - \lambda a_2 - \mu a_1 = 0$$

se llega a $(\Sigma - \lambda I_p)a_2 = 0$ Nuevamente para obtener la maximización se debe tomar el mayor valor propio posible, en este caso λ_2, y así sucesivamente.

3 Descripción de la técnica utilizada

Información meteorológica

Para alcanzar el objetivo de este trabajo, que consiste en encontrar funciones que permitan predecir algunas de las variables meteorológicas en los lugares de interés, partiendo de los datos de gran escala, se necesitaban conjuntos de datos muestra de dos tipos diferentes, para el mismo período de tiempo: uno con las variables predictoras y otro con los valores de la variable a predecir en el intervalo muestra.

Para la variable a predecir se utilizaron los datos de temperatura mínima de la ciudad de México, medidas en el Observartorio de Tacubaya; y para la información regional se utilizaron los datos globales del National Meteorological Center (NMC), de los Estados Unidos.

Se disponía del CDROM CD-Meteo versión 1, que contiene la información sinóptica de presión, temperatura y altura barométrica sobre una malla que cubre todo el hemisferio norte hasta los 15° grados de latitud, a las 0 y 12 h de Greenwich, con una resolución de 450 km en la dirección este-oeste y de 250 km en la dirección norte-sur. En general, para las latitudes bajas, la información es algo pobre, dado que no hay suficientes datos medidos en esa zona.

CD-METEO

CD-Meteo tiene información de temperatura, presión y altura barométrica a las 0Z y 12Z (6 h y 18 h locales respectivamente) La presión a nivel del mar, la temperatura a 500, 700 y 850 mb y la altura barométrica a cuatro diferentes niveles de presión: 250, 500, 700 y 850 mb, desde el año 1946 al año 1989, sobre una malla que cubre el hemisferio norte desde los 15°N. Existen lagunas en la información y, como ya se dijo, los valores obtenidos para las latitudes bajas no son muy confiables.

Datos medidos

Los datos medidos se obtuvieron de los registros que tiene el Observatorio de Tacubaya. Se contó con un archivo de las temperaturas mínimas de los meses invernales: diciembre, enero y febrero, para el período que abarca desde diciembre de 1970 a febrero de 1982. También se utilizaron los registros desde diciembre de 1978 a febrero de 1989.

Definición de la configuración meteorológica mínima

Para realizar el pronóstico de temperatura mínima sobre la ciudad de México, despues de realizar pruebas para observar la correlación entre los datos de los días previos, se consideró necesario contar con la distribución de temperaturas de los dos días anteriores sobre una región que cubre ,primordialmente, el norte del país, ya que en invierno son preponderantes los movimientos de las masas de aire frío provenientes del norte.

Para esto se obtuvieron los valores en catorce puntos distribuidos sobre el territorio nacional (ver figura 1) en los días de los meses de diciembre, enero y febrero, desde diciembre de 1983 a febrero de 1989.

punto	latitud	longitud
1	18	-100.0
2	20	-98.0
3	20	-102.0
4	22	-98.0
5	22	-101.5
6	22	-104.0
7	26	-98
8	26	-101
9	26	-104
10	26	-107
11	29	-97
12	29	-101
13	29	-106
14	29	-109

Cuadro 1

Figura 1 - Puntos seleccionados

Se leyeron las temperaturas a 850 mb, correspondientes a las 12 Z de CD-Meteo, para los 14 puntos seleccionados. Se creó el archivo que contiene la fecha, hora y los catorce datos en cada registro correspondiente a los meses invernales, desde 1983 a 1989.

Vector de datos

El vector de datos fue construido con la temperatura de los catorce puntos en dos días consecutivos: el dia i y el dia i+1.

$T_{1,i}\ T_{2,i}\cdots\cdots\cdots T_{14,i}\ \ T_{1,i+1}\ T_{2,i+1}\ \cdots\cdots\cdots\cdots T_{14,i+1}$ (1)

Por cada par de días consecutivos se generó un vector de datos válidos, guardando la fecha del primer día para luego poder relacionarlo a los datos medidos.

Como la temperatura mínima depende de las perturbaciones locales, a cada punto de la muestra se le resta la temperatura promedio de los dos días de todos los puntos.

$$t_{k,i}=T_{k,i}-\sum_{1}^{14}(T_{j,i}+T_{j,i+1})/28$$

El conjunto de vectores dato, así formado, constituyen la matriz de perturbaciones **T** en la que cada fila corresponde a un caso y cada columna a una de las 28 variables.

Se obtuvieron 377 casos de estudio o vectores. La primera etapa del método consistió en encontrar los patrones climatológicos (vectores propios) invernales. Estos casos son los utilizados para tratar de obtener los patrones de anomalías de temperatura que determinan las temperaturas mínimas en el DF.

Patrones de temperatura

Los 377 casos obtenidos en la etapa anterior son otros tantos vectores de 28 elementos cada uno, correspondientes a las perturbaciones de temperatura. Para obtener los patrones de temperatura que caracterizan el período invernal, se utilizó el análisis de componentes principales, y que en la literatura de meteorología se conoce como EOF o funciones ortogonales empíricas.

El primer paso consiste en construir la matriz **A**, que se obtiene de correlacionar cada uno de las 28 variables de los vectores de datos, consigo mismo y los demás.

Si **T** es una matriz en la que cada fila corresponde a un caso y cada columna corresponde a una de las 28 variables , la matriz de correlación es el producto interno menor.

$$A=T'.T$$

$$a_{ij} = \sum_{1}^{numcasos} t_{k,i} * t_{k,j}$$

Esto nos da una matriz simétrica de 28x28, de la cual se deben obtener los valores y vectores propios. Dado el orden de la matriz, se decidió utilizar el algoritmo de Jacobi, verificando que no hubiera inestabilidades numéricas. La verificación consiste en calcular la ortogonalidad de los vectores propios obteniéndose valores satisfactorios.

$$\begin{aligned} v(i) * v(j) &< 10^{-8} && \text{si } i = j \\ &= 1 \pm 10^{-5} && \text{si } i = j \end{aligned}$$

Los valores propios obtenidos se muestran en el cuadro 2. Los vectores propios son vectores linealmente independientes formados por combinaciones lineales de las variables o casos originales, por lo tanto, tienen la misma dimensión, en este caso 28, y forman una base ortonormal de los vectores del espacio de las muestras.

Por lo tanto todo vector, como por ejemplo una distribución de temperaturas con la estructura definida en (1), puede escribirse como combinación lineal de los vectores propios.

Como A es una matriz simétrica y positiva, sus valores propios son todos reales. Los vectores característicos correspondientes contribuyen a explicar el vector a describir en una proporción que está relacionada con las magnitudes relativas de los valores propios. Una forma de medir que porcentaje de la varianza explica la componente k-ésima es

$$\%\sigma = \lambda_k / \sum_{i}^{n} \lambda_i * 100$$

En el cuadro 2 se presentan los valores propios, su porcentaje de aportación individual y los porcentajes acumulados. Si los valores propios son ordenados de manera tal que

$$\lambda_k \geq \lambda_{k+1}$$

sólo se hace necesario utilizar los primeros vectores característicos o modos principales para obtener la aproximación deseada, dadas las diferencias entre las magnitudes relativas de los valores propios.

Analizando el cuadro 2 se puede observar que el vector propio correspondiente al valor propio mayor explica prácticamente el 50% de la varianza de las muestras. A este se lo llama primer modo.

n	Valor Propio	% explicación de la varianza	% explicación var. acumulada
1°	199.039	49.95	49.95
2°	71.714	19.99	67.95
3°	47.549	11.93	79.88
4°	19.106	4.79	84.67
5°	15.808	3.96	88.63
6°	11.990	3.00	91.63
7°	9.501	2.38	94.01
8°	5.514	1.38	95.38
9°	3.922	0.98	96.37
10°	3.745	0.94	97.31
11°	2.609	0.65	97.96
12°	2.246	0.54	98.50

Cuadro 2 - Valores propios y su participación porcentual

Se observa que la contribución de cada uno disminuye rápidamente, y que tomando en cuenta la contribución de los primeros ocho modos, se tiene la explicación de 95% de la varianza.

Graficando estos primeros modos, se puede tratar de explicar las fuentes o patrones típicos que condicionan las temperaturas del Valle de México.

El primer modo corresponde al avance desde el norte, de una masa de aire frío (figura 2); el segundo modo parece corresponder al avance de masas de aire frío del noroeste (figura 3).

Para verificar la representatividad de los datos, se obtuvo una segunda muestra de 260 casos que cubren cuatro años (del 12/70 al 1/74), y se comparó con la primer muestra.

Se trata de dos muestras, la primera de cuatro años, la segunda de seis, separadas por 13 años. También permite garantizar la estabilidad numérica del algoritmo de obtención de los valores y vectores propios, ya que hay coincidencia en todos ellos, no sólo para los valores principales.

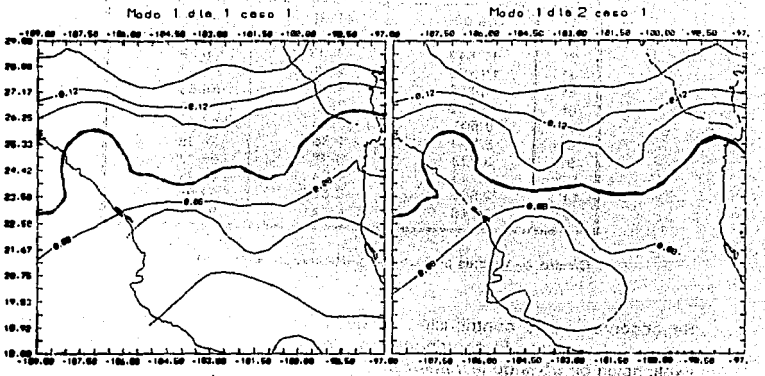


Figura 2 - Primer modo

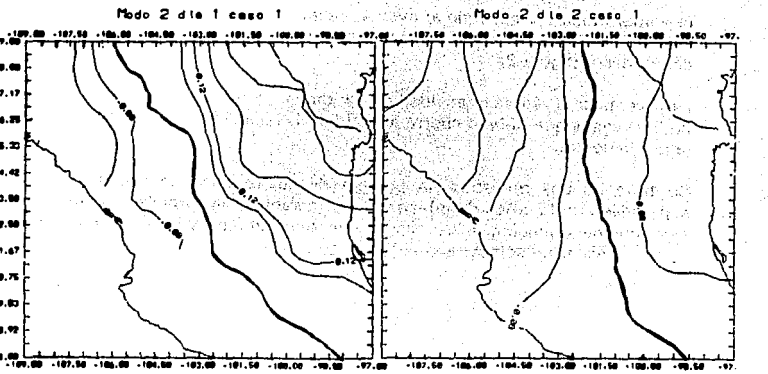


Figura 3- Segundo modo

A continuación se dan los coeficientes del primer modo para ambas muestras, y en el cuadro 3 se comparan los valores propios. Dichos valores muestran la estabilidad de los patrones de temperatura seleccionados.

valor propio 200.247
vector propio .3108 .1876 .2449 .1336 .1015 .1907 .0801 -.0308 -.0694 .0276 -.1533 -.2464 -.3239 -.2060 .2917 .1411 .2208 .0731 .0573 .1716 .0132 .0936 -.1043 .0254 -.2095 -.2891 -.3468 -.1978

valor propio 199.039
vector propio .3116 .1891 .2467 .1342 .1033 .1916 .0809 -.0288 -.0680 .0277 -.1531 -.2386 -.3155 -.2039 .2908 .1414 .2201 .0709 .0555 .1711 .0087 -.0989 - .1071 .0266 -.2166 -.2959 -.3468 -.1978

	Valores propios	propios muestras
años	70-74	83-89
casos	260	377
	200.247	199.039
	73.097	71.714
	47.407	47.749
	18.963	19.106
	16.408	15.808
	11.472	11.990
	9.490	9.501
	5.629	5.514
	3.889	3.922
	3.349	3.745
	2.656	2.609
	2.281	2.246
	1.738	1.727
	.876	.863
	.802	.763
	.551	.533
	.489	.516
	.402	.407
	.359	.342
	.155	.149
	.116	.118
	.079	.082
	.073	.075
	.040	.040
	.037	.037
	.033	.035
	.019	.021
	.000	.000

Cuadro 3- Valores propios para dos muestras

Obtención de datos medidos de la ciudad de México

De Observatorio Meteorológico de la ciudad de México, se obtuvo el un archivo con las temperaturas mínimas de los meses invernales, desde diciembre de 1970 a febrero de 1982. Como estas fechas se traslapaban con las utilizadas en la primera muestra, se trabajo con datos del período 1970-1975.

Puesto que se desea pronosticar la temperatura mínima de la ciudad de México con las perturbaciones de la variable temperatura en los dos dias anteriores, se relacionaron los vectores de las perturbaciones indicados en (1), con la temperatura mínima del día siguiente medida en la ciudad de México.

Se construyeron nuevos vectores, agregándole a los datos de variación de temperatura en los catorce puntos escogidos para el día i y el día i+1, como se indicó en (1), la temperatura mínima del dia i+2 relativa al promedio de los dos dias anteriores $t_{minr\ i+2}$ y el valor real de temperatura mínima $T_{min\ i+2}$, como se muestra en (2).

$$t_{1,i}\ \ t_{2,i}\ldots t_{14,i}\quad t_{1,i+1}\ \ t_{2,i+1}\ \ldots\ldots t_{14,i+1}\ \ t_{minr\ i+2}\ \ T_{min\ i+2} \tag{2}$$

Obtención de coeficientes

Los vectores dato originales $\mathbf{w}_j$ pueden expresarse como combinación lineal de los vectores propios $\mathbf{v}_k$ calculandos.

$$\omega_j=\sum\alpha_k v_k$$

Para el cálculo de los coeficientes, sólo es necesario realizar el producto punto entre los vectores correspondientes, ya que los $\mathbf{v}_k$ son ortogonales

$$\alpha_k=\omega_j^t.v_k$$

Esto se realiza para cada uno de los 260 casos dato que se introdujeron al programa. Para cada vector se determinó el módulo y la contribución a este de la primer componente, las dos primeras componentes, las 5, 10 y 11 componentes. En el siguiente cuadro se muestran los resultados estadísticos de estos valores para toda la muestra calculados como el error cuadrático.

$$\varepsilon_n=\left(\sum_1^{28}(\alpha_k v_k)^2-\sum_1^{n}(\alpha_k v_k)^2\right)/\sum_1^{28}(\alpha_k v_k)^2$$

	media	varianza	desviación
primer compon e1	53.29	648.29	25.46
2 componentes e2	37.12	455.58	21.34
5 componentes e5	16.04	145.37	12.06
10 componentes e10	3.73	13.32	3.65
11 componentes e11	1.88	27.02	5.20

Cuadro 4- Estadísticos de los errores

El error de la primera componente, es el porcentaje del módulo de los vectores v_k no explicado por esta componente; su valor es del 53.29% con una desviación estándar del 25.46%. Si se consideran las dos primeras componentes, el error baja al 37% con una desviación del 21.34%.

En el cuadro anterior se puede observar que el considerar solamente las 10 primeras componentes nos lleva a tener errores de menos del 5%. Esto permite reducir el número de componentes del vector de 28 a 10 sin cometer un error apreciable, estadísticamente, para el conjunto de los casos.

Ecuaciones de regresión

Para realizar el pronóstico, se deben encontrar la ecuaciones de regresión entre los vectores dato originales y los datos registrados. Como vectores dato originales se consideraron dos situaciones: los vectores originales y los vectores formados por los 10 primeros coeficientes de la descomposición lineal de los primeros, sobre los vectores característicos.

Se obtuvieron las funciones de regresión considerando como variable dependiente en un caso, los valores registrados de temperatura y en otro la variación de temperatura respecto de la media de los dos días anteriores. Se graficaron las curvas correspondientes a los valores observados y valores calculados.

Se utilizó el algoritmo de regresión múltiple por pasos (Apéndice A), ya que interesa tener una función con el número mínimo de variables que expliquen el comportamiento de manera satisfactoria.

Las ecuaciones de regresión obtenidas, se concentran en el cuadro 6 y los parámetros estadísticos en el cuadro 7. Las figuras 4 a 7 muestran los valores calculados y observados. Las cuatro funciones de regresión fueron:

Función 1- La perturbación de temperatura o valor relativo, como función de las 28 valores de temperatura (perturbaciones) sobre los puntos elegidos.

Función 2- El valor de la temperatura observada, como función de los 28 valores de temperatura (perturbaciones) sobre los puntos elegidos. (figura 5)

Función 3- La perturbación de temperatura observada como función de los 10 primeros coeficientes de la combinación lineal de los vectores propios (figura 6).

Función 4- El valor de temperatura observada como función de los diez primeros coeficientes de la combinación lineal de vectores propios (figura 7).

Aunque los resultados distan de ser satisfactorios, se observa que las funciones de regresión tienden a explicar una mayor parte de la varianza cuando se trata de determinar las temperaturas observadas, que cuando se consideran las perturbaciones. Esto podría deberse a que tal vez la media considerada sobre todo el dominio los dos días previos, no tiene relación con la temperatura a pronosticar.

También se observa una mejoría, aunque no tan marcada, al considerar las ecuaciones de regresión obtenidas utilizando los coeficientes de la representación de los vectores en las componentes principales: además de aumentar la suma de los cuadrados de la parte explicada por la regresión, hay una disminución del error cuadrático medio[1].

[1] Ver cuadro 6

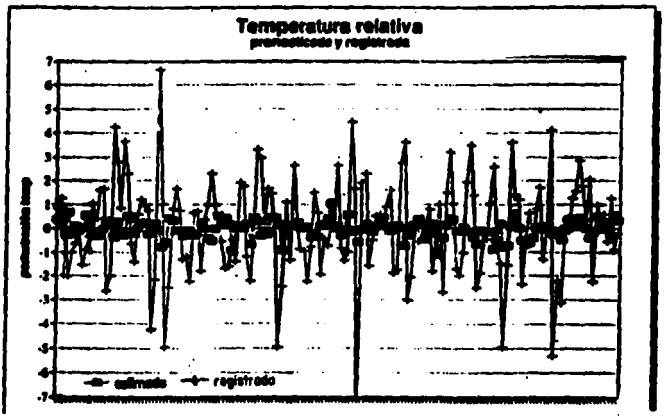


Figura 4- Variación de temperatura mínima observada y estimada en el Valle de México, utilizando como predictores las perturbaciones de temperatura en 14 puntos, los dos días anteriores. (función 1)

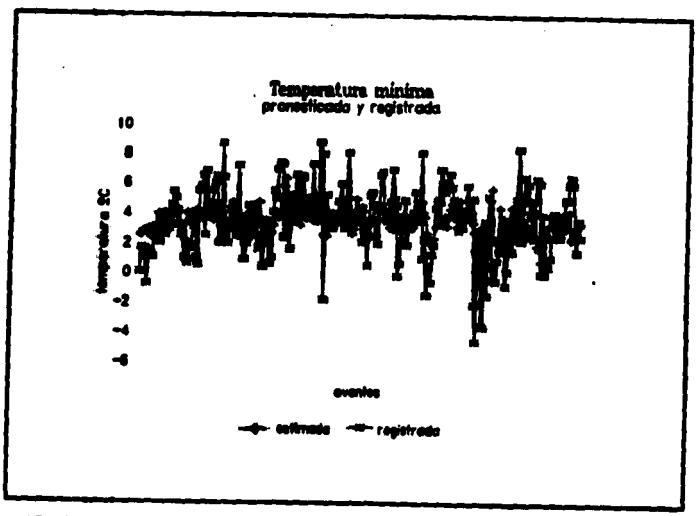


Figura 5- Temperatura mínima observada y estimada en el Valle de México, utilizando como predictores las perturbaciones de temperatura en 14 puntos, los dos días anteriores. (función 2)

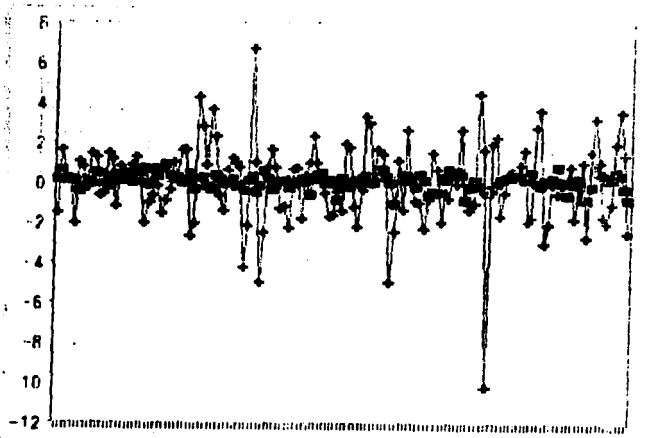


Figura 6 . Variación de temperatura observada y estimada, siendo los predictores los diez primeros coeficientes de la descomposición de los datos en sus componentes principales. (función 3)

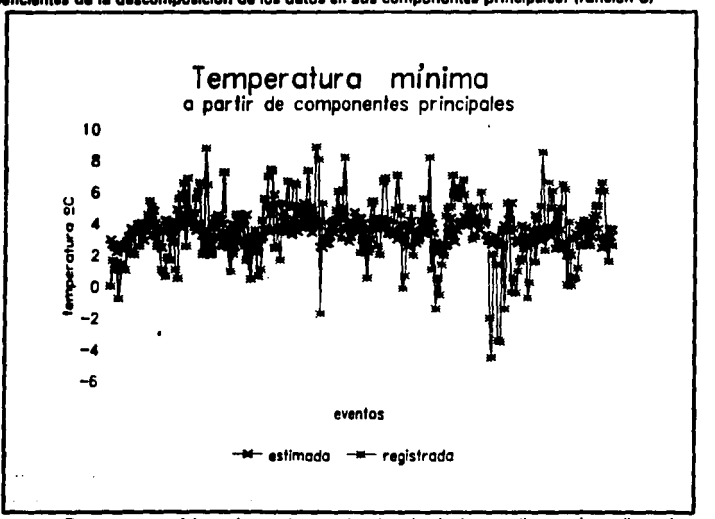


Figura 7 - Temperatura mínima observada y estimada, siendo los predictores los diez primeros coeficientes de la descomposición de los datos en sus componentes principales. (función 4)

Agrupamiento (cluster)

Para mejorar las aproximaciones obtenidas con las ecuaciones de regresión, se decidió utilizar el método estadístico de agrupación de casos análogos. Se consideró la representación de los vectores dada por los coeficientes de su descomposición en los modos principales, por las consideraciones ya expuestas.

Se trataron de relacionar los vectores con la misma dirección, por lo que se optó por el coseno del ángulo o producto interno entre los vectores como medida de la similitud.

$$Similitud(X,Y) = \sum_i (X_i Y_i) / \sqrt{\sum (X)^2 \sum (Y)^2}$$

Los vectores inicialmente están separados y se van agrupando uno por vez. Se buscan los más similares y se agrupan. Para introducir un nuevo vector, cada grupo existente se representa por la dirección promedio de sus elementos.

Aplicando el algoritmo de agrupamiento, la muestra de 260 casos quedó dividida en un gran número de casos con pocos elementos cada uno, dependiendo de que valor del coseno se tomara para decir cuales casos pertenecen al mismo grupo. Sería deseable tener un pequeño número de grupos que aglutinaran a todos los casos, pero nuestra muestra no mostró ese comportamiento.

Considerando como miembros de un mismo grupo a aquellos elementos para los cuales el coseno es mayor o igual a 0.9, o sea que cada elemento de un grupo difiere de su proyección en la dirección media del grupo en menos del 10% de su módulo, la muestra quedo subdividida como se indica en la primer columna del cuadro 5. Hubiese sido deseable obtener un menor número de grupos que englobaran a más casos. En la segunda columna del mismo cuadro se muestran los resultados obtenidos para coseno mayor o igual a 0.89, y en la tercera, los grupos correspondientes a coseno mayor o igual a 0.85.

cos >= 0.9	cos >= 0.89	cos >= 0.85
1 grupo de 18 casos	1 grupo de 19 casos	1 grupo de 28 casos
2 14	6 10 a 14	4 16 a 18
1 13		5 10 a 14
1 10		4 7 a 8
		8 4 a 6
17 de 3 a 7 casos	24 de 3 a 9 casos	10 2 o 3
62 de 1 o 2 casos	42 de 1 a 2 casos	7 de 1 casos

Cuadro 5- Grupos formados para diferente valor de coseno

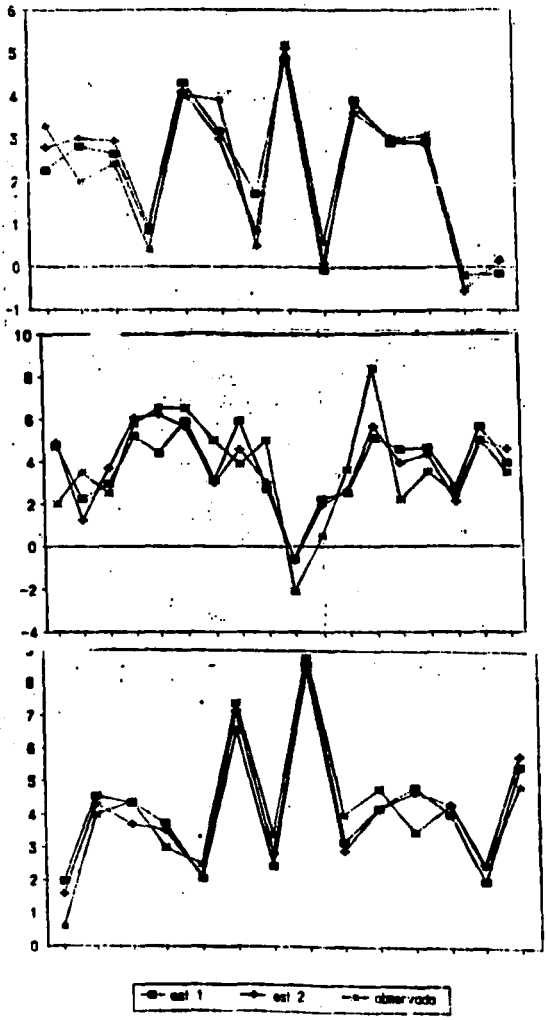
Con coseno de 0.9, poco más del 25% de los casos quedaron en grupos de diez o más. Habría que tener 22 funciones de regresión en las que caerían el 72% de los casos, el 28% restante habrá que buscar por alguna de las técnicas de análogos entre los 62 grupos restantes. En el segundo caso, para coseno mayor de 0.89, hay 31 grupos de más de tres casos, para los cuales habría que tener sus respectivas funciones de regresión y que representan poco más del 80% de los casos de la muestra, y quedarían el 20% para resolver por técnicas de análogos.

Por supuesto podría relajarse el criterio, por ejemplo agrupar a aquellos para los cuales el coseno fuese mayor o igual a 0.85. Tendremos menos grupos con más elementos, pero no tan semejantes entre sí. Con 32 funciones de regresión se cubre el 97% de los casos.

En el capítulo de análisis de resultados volveremos sobre este problema y trataremos de explicar porque tenemos esta disparidad dentro de la muestra. Aparece otro problema al tratar de pronosticar casos no incluidos entre los originales, ya que muchos no caen en los grupos previos.

Se tomaron los tres grupos más grandes con el criterio de coseno mayor de 0.9, esto es el de 18 casos, y los dos de 14 casos cada uno y se calcularon las ecuaciones de regresión, que corresponden a las funciones 5, 6 y 7 que aparecen en el cuadro 6 y cuyos valores estadísticos correspondientes, se muestran en el cuadro 6. En las figuras 8, 9 y 10 se grafican los valores estimados y los observados para las funciones de regresión de cada grupo.

Sin lugar a dudas hay un cambio cualitativo en las funciones de predicción despues de realizar los agrupamientos que cuando se tenía la muestra completa, pero esta ventaja no es tan significativa al introducir nuevos casos a la muestra, para fines de pronóstico, como se discutirá más adelante.



Figuras 8 - 9 - 10 - Temperaturas mínimas observadas y estimadas para tres grupos de datos obtenidos de la muestra para los cuales el coseno del ángulo entre los diferentes vectores es mayor a 0.9 En cada caso se muestran dos funciones de aproximación.

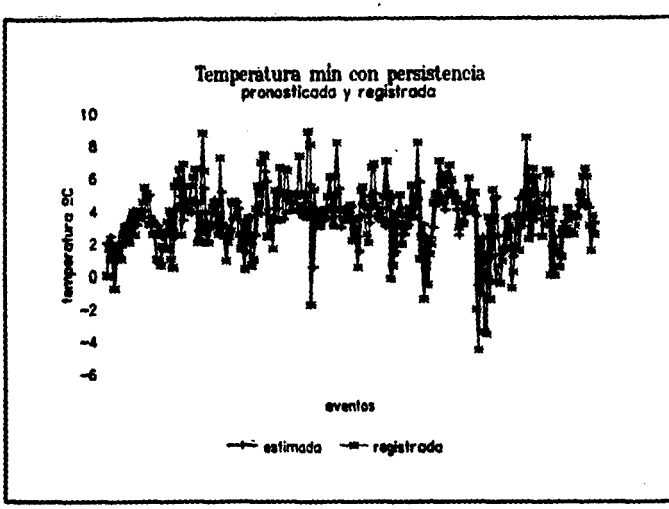
Persistencia

Una de las características de los fenómenos meteorológicos es la inercia. En algunos artículos se menciona la incorporación de alguna variable que tome en cuenta esta persistencia.

Se buscaron las funciones de regresión sobre los 28 coeficientes iniciales a los que se le agregó la temperatura promedio correspondiente a los dos días, y la temperatura observada en el DF el día previo al que se va a pronosticar.

En este caso, la temperatura promedio no pudo mejorar la estimación, pero la inclusión de la temperatura del día anterior produce una mejora sustancial, como puede verse comparando los resultados de la función 8 de los cuadros 6 y 7 con la función 2 que no la incluye. La figura 13 muestra los valores calculados con esta ecuación y los observados.

La función 9 de los cuadros mencionados, corresponde a la ecuación de regresión obtenida utilizando los coeficientes de los diez factores principales y la temperatura mínima observada el día anterior. En este caso, la introducción de la temperatura promedio afecta al pronóstico, pero la temperatura mínima introduce mayor información. La mejoría se observa comparando los estadísticos de la función 4 con la 9. Los resultados pueden observarse en la figura 14.



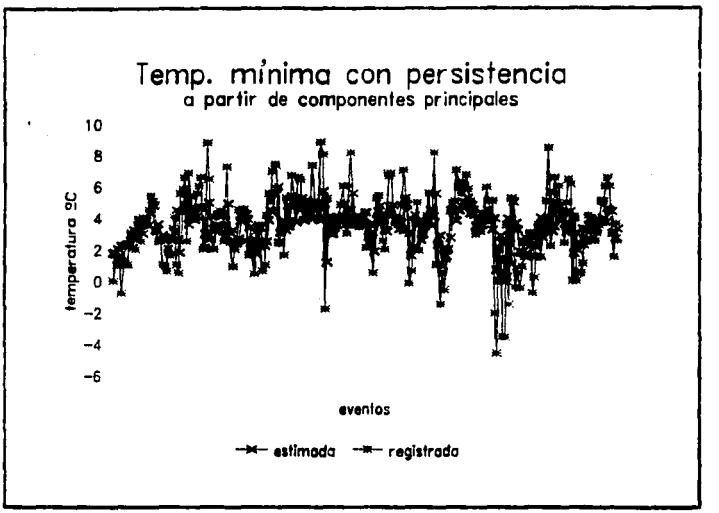


Figura 11 - Temperatura mínima observada y estimada como función de las perturbaciones de temperatura en 14 puntos, los dos días anteriores agregando la temperatura del día anterior para tomar en cuenta la persistencia. (función 8)

Figura 12 - Temperatura mínima observada y estimada como función de los diez primeros coeficientes de la descomposición de los datos en sus componentes principales, más la temperatura del día anterior para tomar en cuenta el factor de persistencia. (función 9)

n°	Ecuación de regresión
1	$-.381 + .099*X_{22} + .146*X_8 + .081*X_1 - .115*X_6 - .056*X_{16}$
2	$4.384 - .511*X_{20} + .331*X_{10} - .12*X_{12} + .136*X_{11} - .092*X_6$
3	$0.047 + .118*X_5 + .346*X_4 - .081*X_1 - .082*X_7 - .18*X_8 - .048*X_3 + .084*X_{16} - .041*X_6 - .015*X_7$
4	$3.536 - .077*X_3 + .065*X_2 - .045*X_4 - .10*X_{10}$
5	$4.348 + .130*X_6 + .224*X_7$
6	$-1.918 + 1.211*X_1 - 3.709*X6 + .942*X_3 + 1.514*X_9 - 0.69*X_5 + .541*X_7 + .365*X_8 + .497*X_2 + .178*X_{20}$
7	$11.429 + .821*X_2 + .37*X_7 + .386*X_6 + .324*X_5 - .24*X_{10}$
8	$2.186 + .432*T_3 - .066*X_{14} - .598*X_{20} + .493*X_{19} + .332*X_{23} - .236*X_{22}$
9	$1.958 + .431*T_2 - .054*X_2 + .022*X^2$

Cuadro 6- Ecuaciones de regresión

n°	R^2	F	Suma C regres	sigma regres	Suma C residuo	sigma residuo	Serr	ecm	emin	emax
1	0.289	.185	34.97	6.97	1168.1	4.59	-.2	4.5	.0	9.6
2	0.368	.0000	166.9	33.38	1063.2	4.35	1.0	2.9	.0	6.3
3	0.035	.432	42.24	4.69	1160.7	4.64	.00	4.5	.0	9.7
4	0.322	.0000	125.52	31.14	1104.5	4.50	-.0	3.1	.0	7.2
5	0.68	.0874	97.45	9.6	41.8	3.79	-.0	0.2	.0	2.0
6	0.850	.0364	45.59	6.51	8.0	1.34	-.01	0.0	.1	1.4
7	0.878	.0001	35.41	11.8	4.9	0.5	.0	0.0	.1	1.2
8	0.535	.0000	352.80	58	877.3	3.6	.5	2.4	.0	5.9
9	0.508	.0000	318.28	106.1	911.8	3.7	5.7	2.5	.0	6.6

Cuadro 7- Parámetros estadísticos para cada uno de los modelos

con

R^2 : factor cuadrático de correlación múltiple
F : nivel de significancia de F
S regresión : parte (cuadrática) explicada por la regresión
S residuos : suma cuadrática de los residuos
Sigmas : valores anteriores entre gl
Serr : suma de los errores
ecm : error cuadrático medio
emin : error mínimo
emax : error máximo

Validación de las funciones de pronóstico

Sin embargo el problema del pronóstico aún no ha sido resuelto. Hasta aquí se trabajó con valores conocidos de las variables predictoras y los correspondientes valores de la variable a predecir, para determinar las ecuaciones de regresión. En esta etapa se tomo otro conjunto de datos, que no había sido utilizado para encontrar las funciones del modelo.

Se utilizaron los vectores datos correspondientes a 110 días, para cada uno de ellos se encontró la descomposición en función de los componentes principales. Utilizando las funciones 2, 4, 8 y 9 se pronosticaron la temperatura mínima para estos días prueba. Los resultados pueden verse en las figuras 15 a 18. En el cuadro 8 se condensa la información sobre la desviación estándar para los 110 días de prueba así como el valor correspondiente para los días (250) de la muestra utilizada para encontrar las funciones. En la última columna se muestran el número de casos en que el error de la temperatura pronosticada respecto a la observada, fue menor a 1°C.

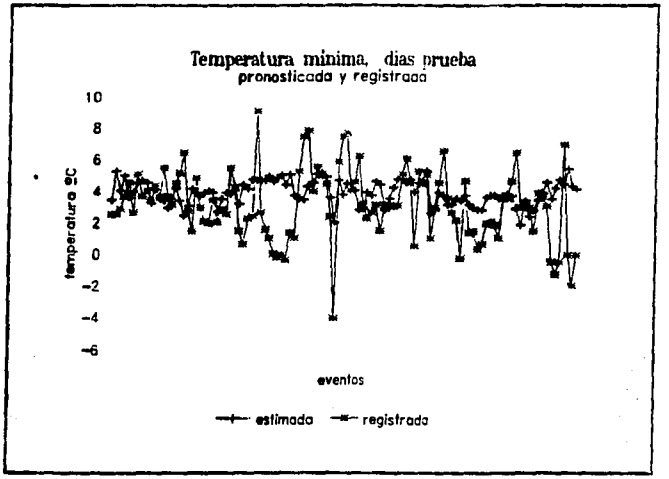
Para los mismos días, se trató de obtener el pronóstico utilizando las funciones de regresión correspondientes a los grupos. Se tomó una muestra de 100 casos y se corrió el mismo procedimiento de agrupación para ver a que grupo correspondía cada caso.

Considerando los grupos formados de tal manera que los miembros de cada grupo no difieran entre sí en un coseno de menos de 0.85, de los 100 casos, 59 pertenecieron a los 10 grupos más grandes; seis más correspondían a grupos pequeños y los otros 35 casos forman un ángulo cuyo coseno es menor que 0.85 respecto a los 250 casos iniciales. Esto quiere decir que nuestra muestra en este sentido no representa a todo el universo.

Si se considera algún otro criterio de similitud que agrupe más a los eventos, seguramente los nuevos casos caerían en alguno de los grupos, pero empeorando el pronóstico.

En la figura 19 se ve uno de estos casos: corresponde al grupo cuya ecuación de regresión aparece como función 5, en el cuadro 6. Los primeros 18 puntos corresponden a días base y los últimos 9 son los días de prueba que caen dentro de dicho grupo.

Los modelos obtenidos reducen la incertidumbre en el pronóstico. Sin disponer de otro modelo, la mejor medida estadística de la variable a pronosticar, es la temperatura promedio estacional. Sin embargo la meteorología estudia las variaciones alrededor de los valores promedio, o sea las anomalías. Una característica fundamental de los sistemas meteorológicos es su inercia, por lo que se puede esperar que el valor de la variable sea el mismo que en el período anterior.



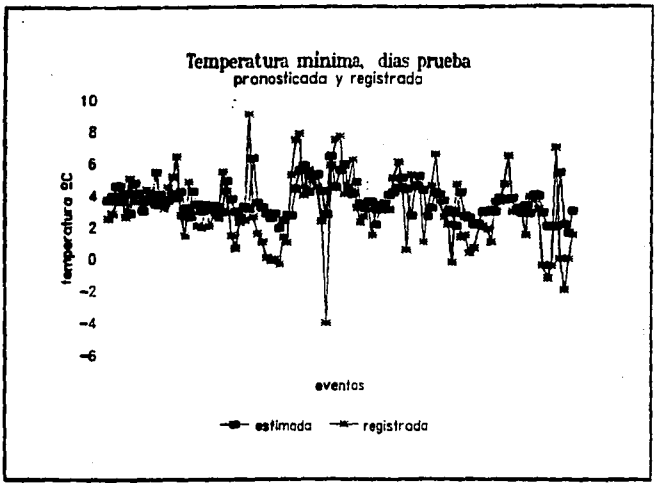
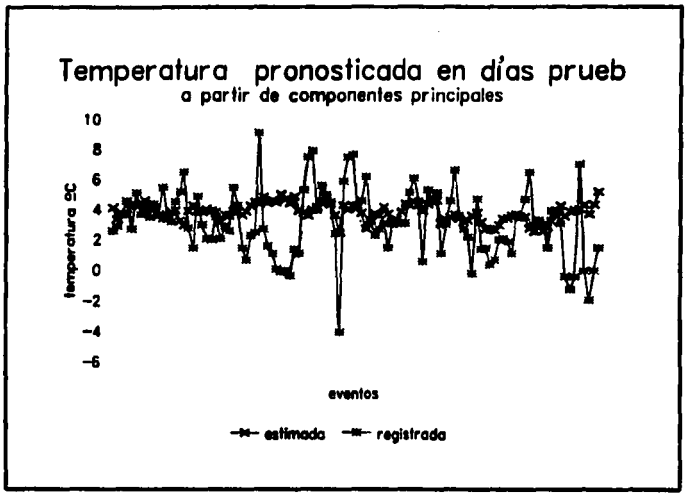


Figura 13 - Días pronosticados utilizando como predictores las perturbaciones de temperatura en 14 puntos, los dos días anteriores. (función 2)

Figura 14 - Días pronosticados utilizando como predictores las perturbaciones de temperatura en 14 puntos, los dos días anteriores, más la variable de persistencia. (función 8)



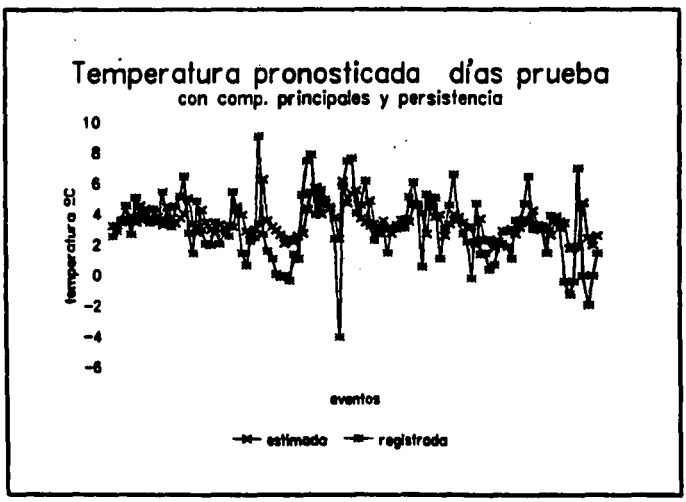


Figura 18 - Días pronosticados utilizando como predictores los coeficientes de la descomposición en componentes principales. (función 4)

Figura 19 - Días pronosticados utilizando como predictores los coeficientes de la descomposición en componentes principales, más la variable de persistencia. (función 9)

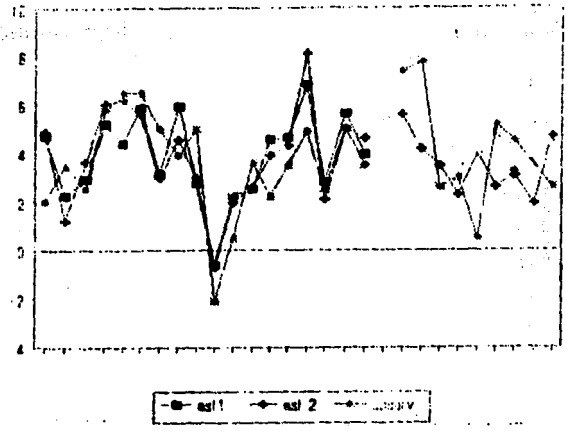


Figura 17 - Temperaturas mínimas observadas y pronosticadas para los días base y días de prueba correspondientes a uno de los grupos anteriores.

Temperatura mínima del día 3 respecto a la temperatura	desviación estandar 110 casos	desviación estandar 289 casos base	nº casos sobre 110 error < 1ºC
promedio	2.00	2.22	30
del día anterior	2.19	2.29	61
función sobre 28 perturbaciones (f3)	1.90	1.71	43
función sobre 28 perturbaciones + temp día anterior (f6)	1.92	1.90	43
función sobre 10 coeficientes (f4)	2.20	1.75	44
función sobre 10 coeficientes + temperatura día anterior (f5)	2.01	1.89	42
función del grupo de la muestra 18 casos base, 9 de prueba (f8)	1.88	1.62	2/11

Cuadro 8- Comparación de la desviación estandar para cada modelo

En el cuadro ocho se comparan las diversas situaciones. Para los 250 días base del estudio se calculó la desviación estándar de la variable a pronosticar, respecto a la temperatura mínima promedio, por un lado, y a la temperatura del día anterior por el otro. Ambas desviaciones practicamente coinciden con 2.2 grados. Para los días de prueba ambas desviaciones son menores, lo que indica que la muestra no es suficiente. Los 250 días corresponden al período 70-74 y los días de prueba a los dos años siguientes. Habría que ampliar el período de estudio a por lo menos diez años para considerar ciertas periodicidades.

Otro detalle interesante lo muestran el número de veces en que la diferencia entre el valor a pronosticar y el pronóstico es menor a un grado: la temperatura del día anterior da valores muy próximos en casi el 50% de los casos, contrariamente a la desviación estándar, que muestra que la temperatura promedio es un mejor pronóstico. Esto puede explicarse debido a que la temperatura mínima del día anterior toma en cuenta la persistencia, que es una de las características climatológicas, pero ignora todas las anomalías.

Todos los modelos presentados en el cuadro, correspondientes a las funciones f2, f8, f4 y f10 mejoran la desviación estándar de la muestra. Es muy clara la contribución de la temperatura del día anterior como puede apreciarse entre las desviaciones producidas por las funciones f2 y f8 calculadas con las perturbaciones reales, y las funciones f4 y f9 basadas en los componentes principales.

Como era de esperarse, la estimación en el caso de los datos de prueba presentan una desviación mayor que en los datos de muestra. Esta diferencia debe disminuir utilizando una muestra que incluya un período mayor de años. Aún así, en el caso de la función f8 que incluye a las perturbaciones de los datos y a la persistencia, la desviación estándar es de 1.8°, similar a lo que reporta Rousseau [Rouss83] quien declara que *se obtuvieron las funciones correspondientes a 72 localidades, con un error medio no superior a 1.8°*.

Falta hacer un comentario sobre la estimación pero con datos agrupados. Si se observan los valores de la desviación de la tabla, parecería ser la mejor alternativa, pero no lo es así como ya se señaló anteriormente. Dependiendo del criterio considerado, el número de grupos cambia. Cuando se tomo coseno mayor a 0.85 para definir el agrupamiento de un conjunto de 100 datos prueba, 59 casos correspondieron a grupos con más de 10 elementos en la muestra original, seis más caerían en grupos de pocos elementos y el resto, 35% de ellos, no corresponden a ningún grupo o caso aislado de la muestra original, y por lo tanto no se podrían estimar.

Segundo caso de estudio: modelo con malla rectangular

El modelo considerado sólo cubre la región continental, por lo que no se consideran las posibles influencias de perturbaciones provenientes de los océanos, tanto del Pacífico como de la región del Golfo. Otra limitación proviene del tipo de datos considerados, se pronostica temperatura como función de los valores de la temperatura los dos días previos, sobre la serie de puntos de la malla.

En este caso se aumenta la región cubierta: se utiliza una malla regular, que abarca desde los 15° hasta los 35° de latitud norte, cada 5°, y desde los -95° a -115°, cada 4°, dando un total de 30 puntos.

También se introdujo además de la variable temperatura, la variable vientos. Para estos puntos se consideran los valores de la temperatura de los dos días anteriores, a las cuales se les resta la temperatura promedio de ambos días, para quedar sólo las variaciones alrededor de dicho valor. Además se consideran los valores de las componentes horizontales del viento, U y V en las direcciones oeste-este y sur-norte respectivamente. Estos valores se toman para el día previo, totalizando 120 variables por dato o evento: los 30 valores de temperatura de los dos días anteriores, más los 30 valores de la componente U del viento, más los 30 valores de la componente V del viento.

Se incluyen los datos correspondientes a once inviernos (diciembre, enero y febrero) del período comprendido entre 1978 a 1989. Se obtuvieron 613 datos o elementos, de los cuales se utilizaron 503 como datos base, dejandose los restantes para los casos prueba.

Los datos de temperatura se extrajeron del CD_Meteo con el programa extractor Ptz que aparece en el apéndice C. Los datos de vientos se obtuvieron de un CD-ROM del NMC, para el que hubo que adaptar el programa de extracción. Los datos de la temperatura mínima del día previo y del día a pronosticar se obtuvieron de un archivo facilitado por el SMN; en este último archivo los valores de los primeros cinco años, estan redondeados al entero más cercano.

Con los procedimientos utilizados en el caso anterior se construyeron dos funciones de regresión, la primera considerando los 60 valores de temperatura, más el valor de la temperatura mínima del día anterior, y la segunda incluyendo los valores de temperatura y los de viento, además de la temperatura mínima del día anterior.

Las funciones obtenidas fueron:

$$f1 = 2.519 + 0.408*T - 0.074*X_{46} - 0.149*X_{43} + 0.082*X_{44} + 0.159*X_{13} - 0.069*X_{16}$$

$$f2 = 2.48 + 0.417*T + 0.112*X_{47} - 0.119*X_{15} - 0.179*X_{103} - 0.107*X_{56} - 0.101*X_{44} + 0.099*X_{91} - 0.19*X_{31} + 0.316*X_{11} - 0.218*X_{16}$$

En el siguiente cuadro se dan los parámetros estadísticos de las ecuaciones anteriores, de manera similar a los del cuadro 7. La única diferencia es la definición del parámetro C que indica la relación entre la suma de los cuadrados de la parte explicada por la regresión entre la suma de los cuadrados del residuo.

nº	R^2	F	C	S_error	e_absm	ecm	emin	emax
1	0.68	0.0	0.808	0.00	1.6	2.115	0.0	8.5
2	0.60	0.0	0.53	-0.04	1.7	2.079	0.0	6.3

Cuadro 9- Parámetros estadísticos de cada modelo

Del análisis de los valores, se observa que con ambas funciones mejora el pronóstico, respecto de los obtenidos para la malla reducida. En el caso de utilizar solo temperaturas, el error cuadrático medio disminuyó de 2.43 a 2.11. El error máximo aumentó, pero debe recordarse que esta muestra es mucho mayor y por lo tanto puede incluir días con mayores anomalidades. Es importante señalar aquí que la muestra de temperaturas mínimas observadas tiene una media de 5.05 con un error cuadrático medio de 2.59 y un error medio absoluto de 2.0.

En el caso de incluir los vientos se obtienen mejoras en todos los parámetros, aunque estas no son muy notables; sin embargo lo que llama la atención es que se obtienen resultados similares y aún mejores pero se nota una marcada dependencia de las variables correspondientes a vientos. (La variables X_1 a X_{60} corresponden a temperaturas y las siguientes 60 a vientos). Esto indica claramente que deben introducirse otras variables que pueden ser fundamentales en la determinación de la temperatura mínima. Las figuras 18, 19 y 20 presentan los valores para los 503 casos base.

Ambas funciones fueron probadas sobre una muestra de 95 casos. Se pronosticó la temperatura mínima para cada caso y luego se comparó con la temperatura observada en esos casos. Los valores del error cuadrático medio, el error absoluto medio y el error máximo se deterioran menos del 10% respecto a los obtenidos con la muestra base. (figura 21 y 22)

Variables pronosticadoras	S_error	ecm	emin	emax	e_absm	nº casos de 95 error < 1°C
temperatura	0.601	2.21	0.0	7.3	1.7	57
temperatura + viento	0.329	2.22	0.0	6.9	1.6	59

Cuadro 10- Comparación de la desviación estándar para cada modelo

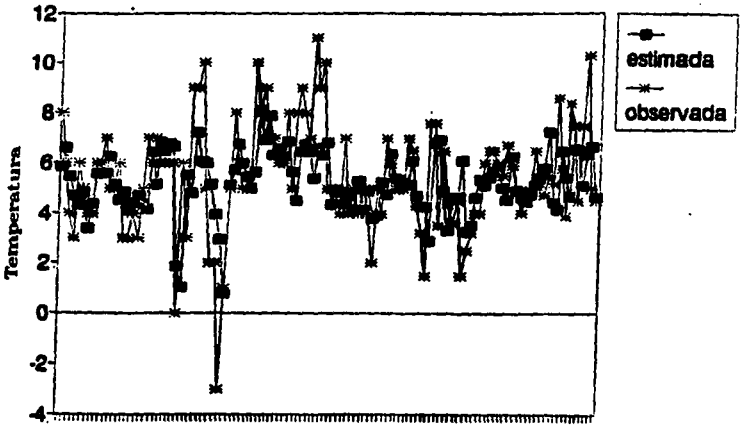
Una mejora notable es la cantidad de casos en que el pronóstico tiene un error menor a 1°. Mientrás que en las funciones obtenidas con 28 datos de temperatura, solamente en el 40% de los casos se obtenian errores menores a 1°, al agregar más variables se llega al 59 y 62% de los casos respectivamente, sin que aumente el error máximo de una manera significativa.

Siguiendo el procedimiento realizado en el primer ejemplo, se trataron de obtener los componentes principales correspondientes a este caso (120 variables), pero la rutina Eigen que se utilizó, basada en el algoritmo de Jacobi no da valores satisfactorios para calcular los vectores propios de matrices de orden superior a unas pocas decenas.

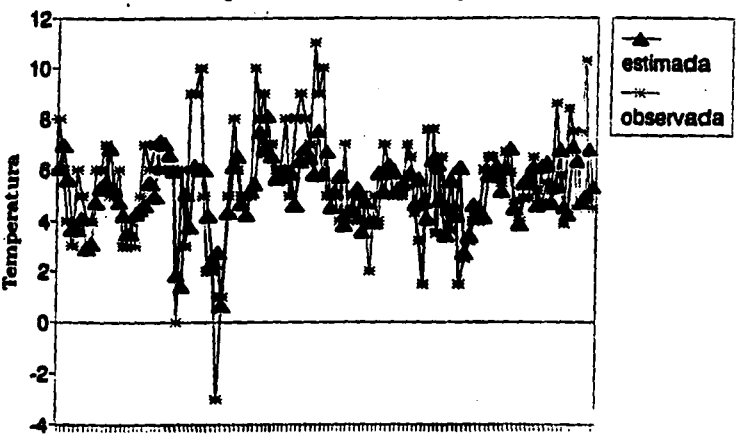
En cuanto a la técnica de agrupamiento, se quería buscar separar los datos en días más fríos y días más cálidos, encontrando las funciones discriminante que separaran los eventos. Pero no es posible encontrar estas funciones partiendo de los 120 datos, por limitaciones tanto de recursos, como de interpretación física. Es necesario encontrar cuales son las variables significativas que definen estos grupos. Para reducir el número de variables y tratar a partir de estas, de formar los grupos, es necesario sacar las componentes principales para tener un conjunto reducido de variables, pero que explique de manera satisfactoria la variación de los datos. Esta parte de la prueba no se pudo realizar por la limitación ya mencionada de los recursos de software.

temperatura mínima estimada

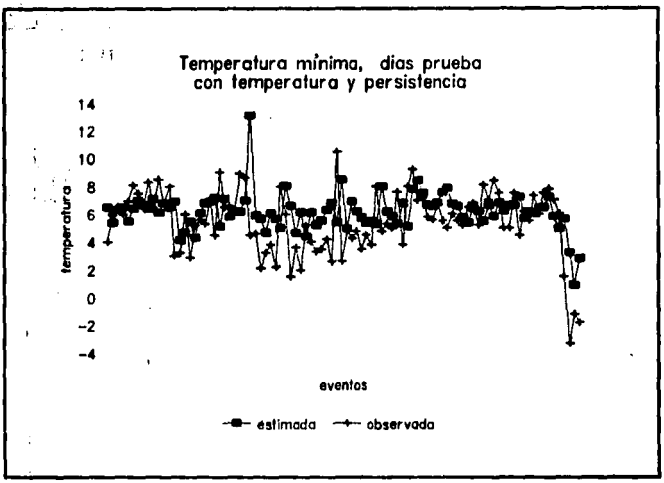
con temperaturas y persistencia



con temperatura, vientos y persistencia



casos



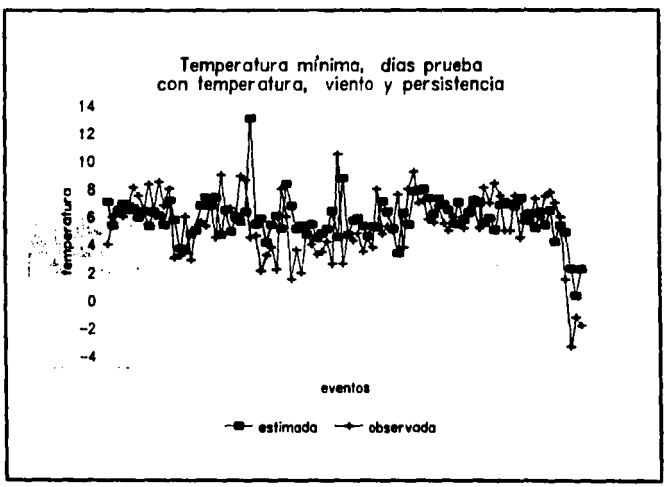


Figura 21 y 22 Días de prueba para el caso de malla completa, utilizando como pronosticadores a) sólo la temperatura b) la temperatura y los vientos

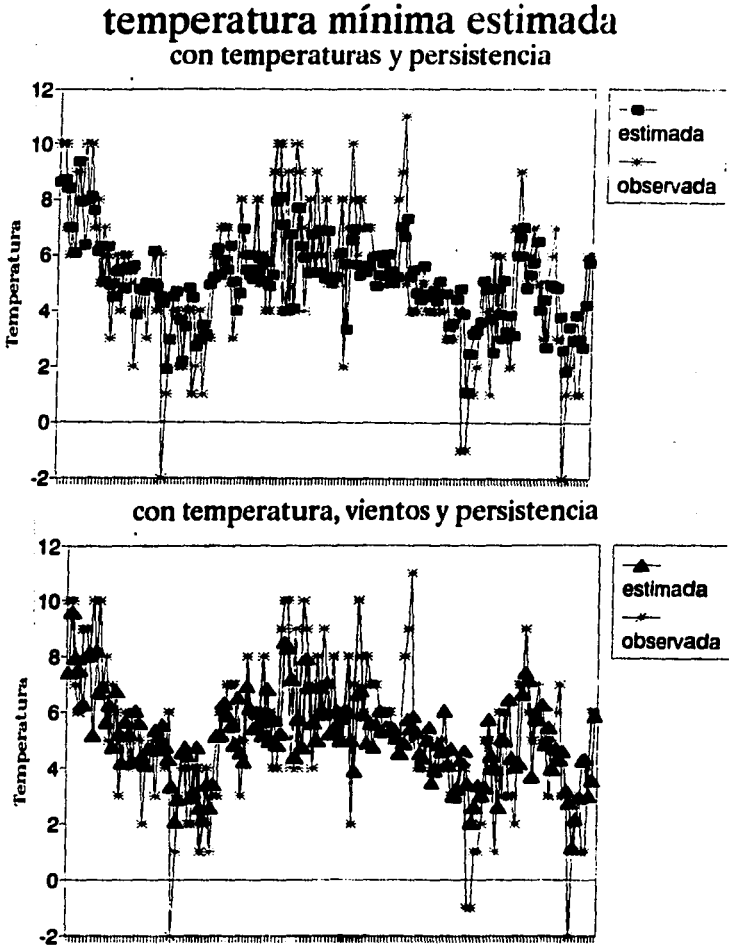
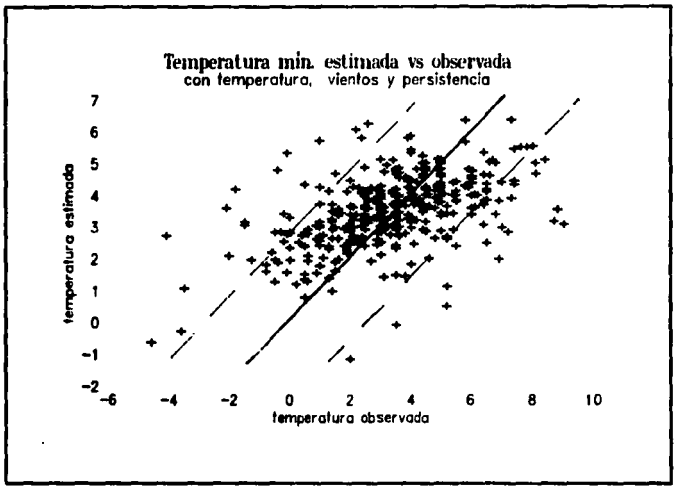
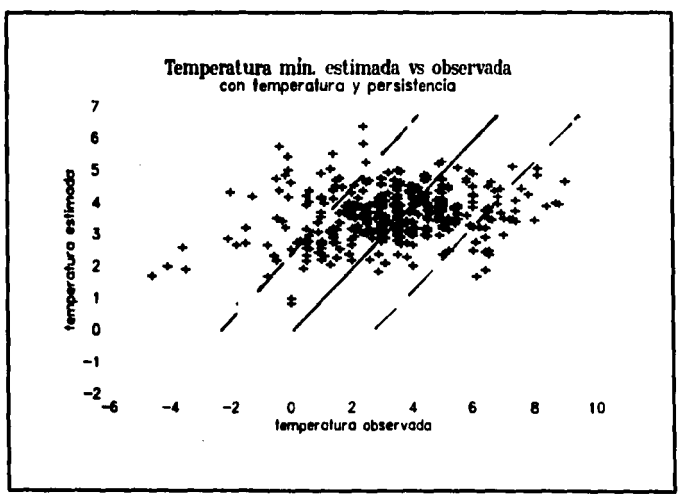


Figura 20 - 21 - 22 - Temperaturas mínimas observadas y pronosticadas para los días base, en el caso de malla completa, utilizando como pronosticadores a) sólo la temperatura b) la temperatura y vientos





Figuras 23 y 24 Diagramas xy de temperaturas observadas y pronosticadas para los días base a) con vientos y temperatura b) sólo con temperatura

4 Aspectos computacionales

Introducción

En esta etapa del trabajo, se pretende encontrar las técnicas más apropiadas para construir los modelos matemáticos de las variables meteorológicas a pronosticar. Esto requiere de la experimentación de diferentes técnicas y alternativas de cálculo, lo que no permite definir previamente un sistema que realice determinada función.

La otra componente fundamental son los datos. Estos se encuentran en fuentes muy diversas, por el origen, los medios de soporte y los formatos. Cada fuente cubre algún período.

Básicamente existen datos del Centro Nacional de Meteorología, NMC, y del Centro Nacional de Huracanes, NHC, de los EUA, del Centro Europeo de Pronóstico Meteorológico Mundial, ECMWF, y de datos provenientes del Observatorio Meteorológico de Tacubaya y las 63 estaciones nacionales distribuidas en todo México.

Actualmente se está terminando de instalar una red automática de monitoreo con 600 estaciones que facilita el almacenamiento y utilización de la información. La red empezó a funcionar este año, incluye a 12 radares digitales y toda la información está disponible en Tacubaya.

La información empieza a distribuirse en CD-ROM, discos ópticos y en cintas de alta densidad, pero también está en cintas de viejo formato de baja densidad, discos flexibles o simplemente hay que accesarla a través de las redes de las computadoras de algunas universidades americanas. El Servicio Meteorológico Nacional está realizando algunos intentos de captura y respaldo de información. Ya dispone de algunos datos en medios magnéticos de los años recientes, y otra se encuentra compilada en hojas de datos que contienen promedios diarios, mensuales, etc.

Los tipos de datos también son muy diversos; están los datos sinópticos provenientes de los modelos numéricos que dan el valor de las variables meteorológicas a distintos niveles, sobre una malla de puntos extensa. Hay datos provenientes de estaciones de monitoreo, de radiosondeo e imágenes fotográficas y de infrarrojo provenientes de satélites.

La cantidad de esta información generada diariamente, es muy difícil de almacenar. Pero, para poder hacer investigación es necesario contar con información extensa y confiable de grandes períodos de tiempo. Indudablemente, es un reto el decidir que información guardar, como compactarla, donde guardarla y las políticas de acceso a ella. Inclusive sería necesario contar con información de que es lo que está disponible y sus características.

Al momento de iniciar esta investigación se contaba con el CD-ROM METEO, ya mencionado, y se esperaba contar con nueva información en el transcurso del año. Para leer dicho disco existía el programa PTZ.C como subproducto de un sistema que despliega datos de manera gráfica, para un día dado.

No existía documentación de la estructura y formato de los archivos dentro del disco, por lo que había que adaptar este programa para obtener la información requerida. Dicha información debe ser validada ya que a veces se obtiene información distinta a la especificada y en otra aparece un mensaje diciendo que no está disponible la información de ese día. Algunos meses después se obtuvo la versión 2 con los programas de lectura y la estructura de los archivos.

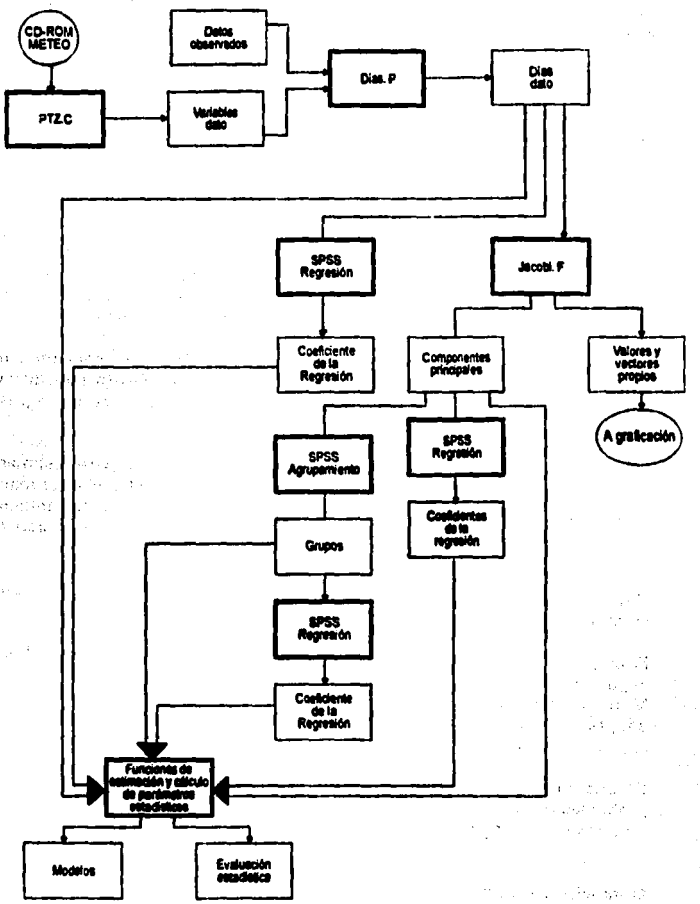
En cuanto a la parte de cálculo numérico, hubiese sido deseable contar con bibliotecas como NAG o IMSL, pero no fue posible tener acceso a ese tipo de herramientas y hubo que utilizar software del dominio público. Para el caso del cálculo de los valores y vectores propios, se probaron rutinas de EISPACK y de Numerical Recipes en Fortran y en Pascal, pero daban muy malos resultados para rangos mayores de 10 o 12. La rutina Eigen que se utilizó estaba en Fortran y esto determinó el lenguaje a utilizar en el módulo. Esta rutina hace uso de una costumbre de Fortran, de pasada de parámetros por referencia, asociando una matriz a un vector unidimensional, lo que obliga a compilar el código cada vez que cambia la dimensión.

Para los algoritmos estadísticos de regresión múltiple y agrupamiento, no se tenía certeza de cuales darían mejores resultados; había que experimentar. SPSS ofrecía una gran variedad de opciones, por ejemplo permite realizar regresión múltiple directa sobre todas las variables, o decirle cuales se pretende que introduzca, o realizar regresión por pasos hasta que no exista ninguna otra variable significativa dentro de cierto intervalo de confianza.

El intentar aprovechar estos recursos disponibles, y las alternativas que se querían probar y que incluso se fueron ensayando durante el trabajo, no permitía escribir un sistema como un todo.

Descripción del sistema

En el siguiente esquema se muestran los bloques o módulos en que se dividió el procedimiento de construcción del modelo. La primera parte se encarga de la lectura de datos. El primer módulo es el programa PTZ.C que lee los datos del CD-ROM y genera un archivo ASCII. No toda la información es correcta. Estos datos deben ser validados para armar los vectores de datos. De esto último se encarga el módulo Dias.p que toma los datos anteriores, busca días consecutivos válidos, y verifica si en el archivo de los valores de temperatura mínima observada se encuentra el dato correspondiente. Si se reúnen estas condiciones genera un vector dato. El conjunto de estos forman el archivo de *datos*, muestra base del estudio.


CD-ROM METEO
Datos observados
Dias. P
Dias dato
PTZ.C
Variables dato
SPSS Regresión
Jacobi. F
Coeficiente de la Regresión
Componentes principales
Valores y vectores propios
Agrupamiento
SPSS Regresión
Coeficientes de la regresión
Grupos
SPSS Regresión
Coeficiente de la Regresión
Funciones de estimación y cálculo de parámetros estadísticos
Modelos
Evaluación estadística

Estos datos se dividen en dos grupos, el primero se utiliza para obtener las funciones de regresión, y se deja un segundo grupo para probar que tan buena es la estimación con las funciones encontradas.

Los datos pasan por diferentes procesos en la búsqueda de la técnica que mejor estime a la variable a predecir; el primero es obtener la ecuación de regresión a partir de estos datos. Utilizando SPSS se obtienen los coeficientes de la ecuación, así como los parámetros estadísticos que la evalúan.

El segundo procedimiento calcula los componentes principales de los vectores dato, y expresa a los datos originales como combinación lineal de los componentes principales. De esta parte se encarga el módulo Jacobi.F que calcula los valores y vectores propios de la matriz de covarianza de los datos. Este módulo también realiza los cálculos estadísticos que permiten decidir cuantos componentes principales se considerarán, dentro del error tolerable. Con los nuevos datos, expresados por los coeficiente de los componentes principales considerados, se calculan las ecuaciones de regresión.

Un tercer procedimiento es tratar de agrupar los datos en conjuntos más homogéneos, utilizando las técnicas estadísticas de agrupamiento, para después encontrar las funciones que mejor modelen a cada grupo, utilizando regresión múltiple.

Todos los modelos de regresión, se utilizan finalmente para estimar la temperatura mínima. Para cada uno de los modelos se calculan el error mínimo y máximo y el error cuadrático medio de la estimación, para los dos conjuntos de datos: los utilizados como base para definir las ecuaciones de regresión y los datos de prueba.

Hay que señalar que al comienzo del trabajo no se tenía claro que alternativas se analizarían y cuales serían las pruebas a realizar con los resultados.

El esquema que se presenta corresponde al primer caso presentado. Para el segundo caso, los datos de vientos tuvieron que leerse de otro disco, el National Meteorological, Center Grid Point Data Set, version 2, para lo cual hubo que adaptar los programas que contenía el disco, para obtener los datos sobre la retícula y los días de interés.

En cuanto al uso de SPSS, que hace difícil tener un sistema único, permite probar las distintas técnicas de una manera muy interactiva e ir observando los resultados. En las conclusiones de este capítulo se volverá sobre este tema.

Obtención de datos

La fuente de datos para este trabajo fue el CD-ROM CD-METEO impreso en México, a partir de los datos proporcionados por el NMC. La base de datos del NMC venía con una serie de programas en Fortran para leer sus archivos. Al

imprimir el CD-ROM, aquellas rutinas se tradujeron a C, quedando incorporadas en el programa PTZ.C, que permite obtener para una fecha dada, el valor de una de las variables sobre los puntos de una retícula.

No se contaba con información acerca del contenido y organización de los archivos de CD-METEO. Dicha información se dedujo de la lectura del código de PTZ. El CD-ROM cuenta con tres directorios: uno con los archivos de valores mensuales, otro con los datos diarios y un tercero de índices a los datos diarios, para permitir el acceso directo a los registros de los datos buscados.

CD-METEO cuenta con información sinóptica a las 0h y 12h, sobre una retícula, para el período comprendido entre 1946 a 1989, de las variables presión a nivel del mar, temperatura a 700 y 850 mb y altura barométrica de 200 mb, 500 mb y 700 mb. Ciertos comentarios del código indican que también hay datos de componentes u y v del viento, aunque el programa PTZ no está habilitado para leerlas.

El programa está dividido en tres módulos. El primero dependiente de la implementación, contiene al programa principal, la rutina que verifica la existencia de la malla, rutinas para desempacar la información sinóptica e interpolarla para la rejilla solicitada, y por último una rutina que realiza la salida de los datos de la malla.

El segundo módulo contiene las rutinas de acceso a la información de NMC. El tercer módulo está relacionado con la determinación de los índices dependiendo de las fechas.

Para obtener los valores de la temperatura a 850 mb, en los catorce puntos seleccionados sobre el territorio nacional, se adaptó el programa PTZ.C pues este sólo permitía leer el valor de las variables climatológicas presión, temperatura o altura barométrica, para un día y hora señalada, sobre una malla regular.

Era necesario leer datos sobre un conjunto de puntos que no forman parte de una retícula regular, se necesitaban los datos para los días consecutivos de los meses invernales. También era muy importante validar la información ya que la base de datos CD-METEO, tiene a veces corrimientos en la información, esto es, como la información está guardada en forma indexada en el disco, cuando se trata de leer cierto dato, calcula el índice correspondiente a su dirección. En caso de no existir ese dato hay una bandera que indica la ausencia, pero por error, en ciertos casos aparece otra información ocupando ese registro o trae el dato más cercano.

Se realizaron modificaciones en las rutinas del módulo dependiente de la aplicación. La versión modificada aparece en el apéndice C; se llamó PTZinv.C Este programa lee los valores de la variable temperatura a 850 mb a las 12Z, que corresponden a las 6 h en México, para los puntos seleccionados, creando

un archivo con un registro por cada día que logra encontrar. Se incluye la fecha a efectos de validación y para relacionarlos con los archivos de temperaturas medidas en el Observatorio de Tacubaya.

Con este programa se obtuvieron dos muestras: la primera con los datos correspondientes al período 1983-1989 y la segunda al período 1970-1982, de donde se obtuvieron 637 días dato.

Para el segundo caso de estudio hubo que leer datos de velocidad del viento, de otro disco, para lo que fue necesario adaptar otro programa, que no se documentará pues fue un trabajo similar. El leer los valores de los datos observados para los días de la muestra requirió de pequeñas rutinas dependientes de la fuente.

Generación de vectores

El vector de datos correspondientes a la configuración mínima contiene 28 elementos, que como ya se señaló, consisten en la temperatura a 850 mb y a las 12Z para los catorce puntos señalados, dos días consecutivos.

$$T_{1,l} \quad T_{2,l} \cdots\cdots\cdots T_{14,l} \quad T_{1,l+1} \quad T_{2,l+1} \cdots\cdots\cdots T_{14,l+1} \qquad (1)$$

Como interesan las perturbaciones o anomalías de temperatura, respecto de la media, se calcula el valor promedio sobre los 14 puntos y para los dos días, y se le resta a cada uno de los valores.

$$t_{k,l} = T_{k,l} - \sum_{1}^{14}(T_{j,l}+T_{j,l+1})/28$$

El módulo que realiza esta función aparece en el apéndice D y se llama Dias.pas. Se procesaron los dos conjuntos de datos obtenidos de CD-METEO y se obtuvieron los vectores correspondientes a dos muestras de datos válidos, la primera con 260 casos del período 70-74 y la segunda con 377 casos del período 83-89.

El módulo lee del archivo generado por PTZinv la fecha y los datos de un día, del siguiente, verifica si las fechas son consecutivas y corresponden a las 12Z, calcula la temperatura promedio de ambos días en la región y guarda en el un archivo los vectores dato generados y en otro la fecha correspondiente al día uno.

En la segunda parte del trabajo, se utiliza el mismo programa para generar vectores datos idénticos a (1) más una componente correspondiente a persistencia que se representó por la temperatura mínima observada el día anterior. En el caso final, se realiza el mismo tratamiento pero se incluyen más puntos de la retícula.

Debo mencionar que no se realizaron rutinas más generales ya que al comienzo no se conocía como iba a evolucionar el modelo: sólo después de obtener resultados, se decidía que modificaciones realizar en cuanto a variables, retícula, días a promediar.

Cálculo de vectores propios

La parte más importante consiste en la determinación de los vectores propios de la matriz de correlación de los datos, que es una base del espacio vectorial del conjunto de los datos. Los vectores dato son luego representados en la nueva base.

El programa que realiza estas funciones se llama Jacobit.for y su código se encuentra en apéndice E. Lee los vectores de las muestras en una matriz **T** en la que a cada fila corresponde un caso y a cada columna una de las variables de temperatura. **T** es una matriz de Nx28, con N igual al número de los vectores de datos.

Luego calcula la matriz de correlación **A** como el producto interno menor

$$A = T'.T$$

con

$$a_{ij} = \sum_{1}^{numcasos} t_{k,i} * t_{k,j}$$

La matriz que se obtiene es de 28x28. Después de haber realizado algunas pruebas para asegurar la convergencia de distintos algoritmos, se decidió utilizar el algoritmo de Jacobi ya que se trata de una matriz de orden moderado. Esta elección definió el lenguaje del módulo, ya que se disponía de una rutina de biblioteca en Fortran que era la que daba mejores resultados.

Cabe aclarar que aún para esta dimensión hay problemas de inestabilidad numérica importantes; por ejemplo se trataron de utilizar las rutinas que provee Numerical Recipes de Cambridge University Press, y se verificaron los resultados; los vectores propios obtenidos no resultaban ortogonales, obteniéndose como valor del producto interno de los vectores $v_i.v_j = 10^{-1}$.

Este error podría ser aceptado, ya que los datos del problema tienen un error de ese orden, sin embargo se prefirió utilizar una rutina más estable.

El cálculo de los valores y vectores característicos lo realiza la rutina Eigen, utilizando el método de Jacobi de diagonalización de la matriz.[1] Se obtienen todos las valores y vectores propios. Por ser **A** una matriz de correlación es simétrica y positiva, por lo tanto tiene n valores reales y en nuestro caso particular, no hubo raíces múltiples.

El peor inconveniente de esta rutina es la pasada de parámetros de matrices como vectores, tal como se realizaba en Fortran, por lo que es el usuario quien debe cuidar de asignar los datos correctamente. Esto obliga a tener que compilar para cada dimensión, no pudiendo modificar ésta, con un valor dato.

La rutina *Comprueba* verifica la ortogonalidad de los vectores, realizando el producto punto entre ellos y calcula la relación

$$\mathbf{A}x=\lambda x$$

obteniéndose resultados satisfactorios de al menos cuatro cifras significativas correctas.

Después la rutina *Coeficientes* expresa a cada uno de los vectores originales como combinación lineal de los vectores propios. Los coeficientes para cada vector w_j (renglón de la matriz T) se obtienen realizando el producto punto

$$\alpha_k=v_k^{/}\circ\omega_j$$

donde v_k es el vector propio k-ésimo.

Finalmente se determina la contribución de cada componente al módulo del vector w_j, y se calcula su contribución a la explicación de la varianza. Para esto determina el error en el módulo de cada vector dato, si se consideran sólo una componente, dos, cinco, etc, y finalmente realiza la estadística del error medio, la varianza y la desviación estándar para todos los casos de la muestra. Los resultados aparecen en el cuadro 4 del capítulo 3.

Como salida se generan dos archivos que se utilizarán para hacer los modelos de regresión: uno con los diez primeros coeficientes y la temperatura mínima observada y otro con los datos originales y la temperatura mínima (coefymin.dat, dataymin.dat)

---

[1] Método adaptado por Von Newman para computadoras grandes y que aparece en 'Mathematical Methods for Digital computers' editado por Ralston & Wilf, Wiley and Sons, 1962

Obtención de las funciones de regresión

Para calcular las funciones de regresión se decidió utilizar la paquetería SPSS, porque ofrece varias opciones del algoritmo de regresión y cálculo de parámetros estadísticos con los que se podía afinar el modelo. SPSS permite utilizar los métodos directos para meter todas las variables en la función de regresión, o seleccionar cuales variables meter o sacar, indicando en cada paso los valores de R^2 (coeficiente de correlación múltiple) y del valor de significación F. También dispone de los métodos por pasos, hacia adelante y hacia atrás (*stepwise, forward, backward*).

Aunque se realizaron algunas otras pruebas, como interesaba encontrar las funciones de regresión que explicaran la varianza con un número reducido de variables, se optó por el método de introducción de variables por pasos. A continuación se transcribe aproximadamente el programa utilizado para realizar las funciones de regresión. Generalmente se realizan varias corridas con diferentes criterios intentando mejorar el coeficiente de regresión R^2, permitiendo la introducción de más variables.

```
DATA LIST FILE\'Datminab.dat'
      FREE/ t1 TO t28 tminrel tminab
REGRESSION
      /VARIABLE = t1 TO t28 tminab
      /CRITERIA = PIN(.30) POUT(.34)
      /DEPENDEN = tminab
      /METHOD   = STEPWISE.
```

```
DATA LIST FILE\'Coeminab.dat'
      FREE/ t1 TO t10 tminrel tminab
REGRESSION
      /VARIABLE = t1 TO t24 tminab
      /CRITERIA = PIN(.25) POUT(.30)
      /DEPENDEN = tminab
      /METHOD   = STEPWISE.
```

Estos programas se corrieron primero para todo el conjunto de datos, y utilizando como variable dependiente las temperaturas observadas, pero a las que se le restaba el valor promedio de los dos días anteriores, después se correlacionaron con los valores medidos directamente. Lo mismo se realizó con los coeficientes obtenidos para la representación de los datos en la base de los vectores propios.

Luego se separaron los datos en grupos más homogéneos, como se indicará en el próximo inciso y se encontraron las funciones de regresión de cada grupo. En los cuadros 5 y 6 del capítulo 3 se muestran las ecuaciones obtenidas y los parámetros estadísticos de los modelos. En las figuras 6 a 12 se muestran las

gráficas de los datos observados y de los datos calculados con las funciones de regresión.

Procedimiento de agrupación

Para mejorar las funciones de regresión, se consideró necesario utilizar alguna técnica de agrupamiento de casos similares, para disminuir la disparidad de la muestra. Como medida de similitud, se utilizó la dirección de los vectores. Como en el caso de las funciones de regresión, se decidió utilizar nuevamente SPSS que ofrece funciones de agrupamiento o *cluster*.

Se utilizó como medida de similitud el producto punto entre vectores dato. Los algoritmos anexan un elemento a un grupo ya formado, cuya dirección promedio sea la más parecida al elemento a introducir. Los grupos formados varían algo si se utiliza el método Baverage, que minimiza la distancia promedio dentro del grupo, o Waverage que minimiza la distancia promedio entre cada par de elementos del grupo. Las instrucciones fueron:

```
CLUSTER alfa1 TO alfa10
      /METHOD = WAVERAGE
      /MEASURE = COSINE
      /PLOT HICECICLE
      /PRINT SCHEDULE
```

El procedimiento comienza con todos los casos separados y los va agrupando hasta terminar en un solo caso. El criterio para decidir hasta cuando agrupar, es un compromiso entre el número de grupos con los que sería deseable quedarse y la homogeneidad dentro del grupo. Cuanto menos grupos se deseen, estos serán menos homogéneos.

El cuadro 7 del capítulo anterior, presenta los resultados de este procedimiento. Se obtuvieron algunas de las funciones de regresión de los grupos obtenidos con coseno > 0.89 que resultaron aproximaciones excelentes. Sin embargo, para validar resultados se agruparon los datos hasta coseno > 0.85 puesto que sólo así el 60% de los casos coincidió con alguno de estos grupos.

Módulo de estimación y validación

Con SPSS se obtienen los coeficientes de las funciones de regresión para cada una de las alternativas estudiadas. En este módulo se utilizan las ecuaciones obtenidas para estimar el valor de la variable a pronosticar, la temperatura mínima, con el juego de datos base y con los conjuntos de datos de prueba.

Para ambos conjuntos calcula el error cuadrático medio y los errores mínimo y máximo, y el número de casos en que el error es menor a un grado. Estos

valores permiten comparar estadísticamente los distintos modelos. Los datos de salida son los que se muestran en las gráficas como valores estimados y valores observados y se resumen en los cuadros correspondientes.

Observaciones y comentarios

En este trabajo se probaron distintas técnicas para la obtención de los modelos estadísticos de variables meteorológicas. Aún no se tiene perfectamente definido el modelo; se ve la necesidad de utilizar más variables y de contar con períodos extensos de datos confiables.

Computacionalmente sería deseable contar con un sistema que ayude en la creación de un modelo de una manera más automática. Inclusive se llegó a plantear la posibilidad de escribir un programa que encontrará el *mejor modelo*. Pero hay una serie de alternativas a decidir, en el procedimiento para llegar al modelo, que si no se definen previamente, lo hacen crecer en complejidad de manera exponencial.

Independientemente de lo anterior, para poder escribir un sistema más integrado, se necesita contar con una biblioteca estadístico-matemática como las bibliotecas IMSL o NAG, con la documentación correspondiente.

A continuación se van a discutir cada uno de las alternativas que deben resolverse durante el modelado.

1 - La lectura de datos depende de la fuente de información; un mismo modelo puede requerir de varias fuentes distintas, y en cada caso habrá que ver con que información se cuenta sobre los formatos de los archivos, o si estos traen algún programa de lectura.

2 - La siguiente decisión importante es cual es la estructura del vector de datos. En el primer caso se tomaron las temperaturas de dos días consecutivos sobre 14 puntos. A estos datos se les restó la temperatura promedio de los 28 elementos para obtener las anomalías. Aún en el caso de tratar de automatizarlo con una red regular, hay que decidir la superficie a cubrir y el espaciamiento de la red:
¿Que región y cuántos grados de separación de los puntos en latitud y longitud?
¿Cuantos días previos se consideran, dos, tres ?
¿Las anomalías hay que definirlas respecto a la temperatura de esos días, o del mes, o de la temporada?
¿ Y cuando se tienen variables diferente como en el segundo caso, como se equiparan los datos? ¿ Como se definen las anomalías? ¿Sería conveniente normalizar las variables?

3 - Después de calcular los vectores propios y analizando la contribución de cada uno de ellos a la explicación de la varianza de los datos, se define el

número de componentes con los que se van a representar los datos. Aquí hay que definir el porcentaje de error tolerado.

Se utilizó como único criterio para decidir con cuantos componentes quedarse, el porcentaje de la varianza explicada, sin tomar en cuenta que algún componente puede no ser importante en cuanto a la covarianza de las variables independientes, pero si serlo en la determinación de la variable independiente [Jolli86]. Esto se trata en el apéndice B.

4 - En el procedimiento de agrupamiento, SPSS da la posibilidad de elegir entre seis algoritmos diferentes con cinco criterios para medir la distancia. Cada alternativa da diferentes agrupaciones. Se probaron sólo dos o tres. El problema siguiente es con cuantos grupos quedarse.

Probar aquí todas las alternativas es imposible, ya que se requiere para cada grupo de cada caso, encontrar las ecuaciones de regresión y aplicarlas a los datos del grupo para determinar las estadísticas de errores, después ver cuales de los días prueba caen en esos grupos y estimar con las ecuaciones correspondientes. El peor problema a resolver es que hacer con los casos que no corresponden a ningún grupo. En este momento no se puede contestar esta pregunta. Habrá que haber trabajado con muestras mucho mayores para tener respuestas.

Otra alternativa es utilizar el análisis discriminante para separar los elementos de la muestra en conjuntos homogéneos bajo algún criterio predefinido. Se pretendía utilizar en el segundo caso de estudio, pero no se pudo realizar este análisis pues no se pudo trabajar con las 120 variables, dadas las limitaciones del software disponible.

5 - El método seleccionado para obtener las funciones de regresión fue el de *por paso*. En cada corrida, sin embargo hay que determinar el valor mínimo del parámetro F para que una variable sea considerada en la función o sea removida, y esto se decide viendo como se modifica el valor de R^2 en cada iteración.

No creo que sea posible, ni siquiera deseable contar con un sistema que busque el modelo óptimo para cada variable. En problemas de ingeniería, con este grado de dificultad, he visto el uso de algoritmos con la técnica Montecarlo para evitar el análisis de todas las alternativas y tratar de ir por los modelos posibles. En este tema, siento que debe ser el experto en meteorología, quien ayude en cada caso, a tomar las decisiones, y no utilizar técnicas aleatorias.

Para la construcción de los modelos para cada variable, es imprescindible trabajar dentro de un grupo donde esté el especialista en meteorología junto con un asesor en estadística. Una vez definido el modelo para cada variable, por ejemplo la temperatura mínima y máxima, se podría construir una herramienta que permitiera sistematizar el procedimiento, para calcular la función correspondiente a cada punto de interés sin olvidar las particularidades en cada caso.

Este procedimiento de modelación debe repetirse para cada tipo de variable meteorológica: probabilidad de precipitación, viento, cantidad de lluvia, etc. Para cada variable habrá que adaptar la herramienta, ya que las variables predictoras serán diferentes.

Dentro del contexto en que se realizó este trabajo, que como se mencionó en la introducción, es el de satisfacer la demanda del SMN de contar con herramientas numéricas y gráficas para dar respuesta a las demandas de información inmediata, y a los estudios e investigaciones que se deban seguir desarrollando, creo muy importante realizar ciertas tareas prioritarias: realizar una base de datos para saber con que información se cuenta y como accesarla, esté esta en medios electrónicos o no; definir que información se debe guardar, como se va a procesar, definir normas en cuanto a formatos, métodos de compactación y medios de almacenamiento. Si esto no se realiza en forma inmediata, se seguirá perdiendo información irrecuperable.

También es necesario analizar cual es el flujo de información actualmente, ya que el SMN posee una red interna y se comienza a interconectar con la red de monitoreo, aunque aún parecen no estar definidas las bases de información globales y cada departamento trabaja con sus propios criterios.

Para la realización de una herramienta realmente operativa, que ayude a mejorar el pronóstico a 24 y tal vez 48 horas, del cual formarían parte los modelos encontrados con las técnicas que aquí se estudiaron, se necesita diseñarlo junto con los futuros usuarios, para que cubra sus necesidades reales, y no ocurra como con el modelo barotrópico que ya poseen y no utilizan.

Una característica muy importante es la necesidad de estar conectado a una red que le suministre los datos de entrada en tiempo casi real. El diseño de este sistema es un reto interesante.

5 Análisis y evaluación de resultados

El objetivo de este trabajo fue probar el uso de diversas técnicas estadísticas mencionadas en la literatura, utilizadas para realizar el pronóstico regional de las variables meteorológicas. En especial interesaba aplicar la técnica de las funciones empíricas ortogonales y la técnica de agrupamiento, utilizadas en diferentes artículos para encontrar patrones que ayuden a entender, y así estimar las variables meteorológicas.

El análisis de componentes principales mostró que puede ser usado para reducir el número de variables, especialmente cuando se necesitan cientos de ellas.

La reducción del número de variables a unos pocos componentes principales, en este caso los diez primeros explicaron más del 90% de la varianza, permiten obtener resultados equiparables.

Pero esta ventaja no se obtiene gratuitamente, la determinación de los componentes principales implica la obtención de los vectores propios de la matriz de covarianza. Y las dimensiones de la matriz dependen del número de variables. Si se tiene que trabajar con más de un centenar de variables, como ocurre en el segundo caso de este trabajo, donde fue necesario emplear temperaturas a 850 mb de los dos días previos y las componentes horizontales u y v de los vientos del día anterior sobre una malla de 30 puntos, se llega a las 120 variables. La determinación de los vectores propios para una matriz de este orden ya es relativamente compleja.

Además los modos obtenidos para el caso de las temperaturas no son tan explícitas como se esperaba. Es probable que los patrones obtenidos con variables tales como vientos o trayectorias de huracanes sean más definidos, ya que se trata de fenómenos también más identificables.

También es algo problemática la interpretación de vectores propios que incluyan diferentes tipos de variables como temperatura y vientos, ¿ o habrá que considerarlos por separado como lo hacen algunos trabajos ? [Rouss83], [Mozer93]. En el segundo caso, donde se tenían más de 120 variables, se podrían calcular los vectores propios para cada uno e los campos por separado: temperatura, viento en ambas direcciones; para cada uno elegir algunos de los componentes principales, para luego buscar las funciones de regresión sobre estas nuevas variables, que se podrían reducir a 20 o 30 solamente.

También queda otra indeterminación cuando se utilizan diferentes variables: como se están usando anomalías respecto de la media (¿cuál media?), no se puede

normalizar cada una de las variables. Esto obliga a tener que utilizar la matriz de correlación y no la de covarianza de los datos, para independizarse de las escalas. El utilizar variables normalizadas o no, cambia la estructura de la matriz y lleva a diferentes vectores propios. Al no utilizar variables con media cero, lleva a tener valores principales muy diferentes, de tal manera que con los primeros se explica gran parte de la varianza. Si a cada variable se le resta la media, los primerosa valores principales son del mismo orden de magnitud, y es necesario entonces considerar un número mayor.

En cuanto a las técnicas de clasificación utilizadas, mostraron que al agrupar los eventos, se pueden obtener funciones que ajusten mucho mejor a los datos, sin embargo también se constató que los criterios matemáticos usados para clasificar no son los adecuados para el pronóstico, ya que no queda bien definida la pertenencia a los grupos, para la totalidad de los datos.

En la medida en que se lograr clasificar a los eventos tomando en cuenta consideraciones físicas que los agrupen en un número aceptable de casos, se podrá llegar a mejorar las funciones de pronóstico. Esto se podría realizar utilizando análisis discriminante sobre los componentes principales. Habría que decidir cuales son los componentes principales realmente determinantes para clasificar los datos según un criterio físico.

Conclusiones

Este es el primer trabajo que se hace en México intentando hacer pronóstico utilizando técnicas estadísticas, con la intención de apoyar otros esfuerzos que se han realizado en el SMN, como la red de monitoreo digital.

Los resultados aún no son satisfactorios, pero con todas las limitaciones que se realizó, se logró disminuir la incertidumbre del pronóstico.

El uso de estas técnicas pero abarcando una mayor área física, un período de tiempo mayor, e incorporando las variables físicamente significativas, permitiría obtener los modelos operativos necesarios.

También puede concluirse que este procedimiento puede irse sistematizando, aunque en todo momento deberá haber la flexibilidad para incorporar modificaciones, intrínsecas a la tarea de modelar. Esta experiencia resalta la necesidad de contar con mayores recursos, tanto de equipo de computo, como en la disponibilidad de los datos y la necesidad de conformar un grupo de trabajo con habilidades en los aspectos físicos, meteorológicos y en estadística.

Habría que hacer intentos en otras direcciones; hay que probar con las técnicas estadísticas del análisis multivariado. También, por las características del problema, las redes neuronales pueden ser una alternativa que debe ser explorada para la estimación de parámetros meteorológicos, ya que se cuenta con gran cantidad de datos para enseñar a la red. Es un tema abierto, complejo, al que este trabajo pretende aportar la experiencia realizada.

Apéndice A

Método de regresión múltiple por pasos (stepwise)

En el modelado del problema se utilizó la técnica de regresión lineal múltiple con el procedimiento de selección de variables por pasos (stepwise regression) para la optención de la mejor ecuación de regresión. En este apéndice se sumariza la técnica empleada en los capítulos 3 y 4.

Se desea establecer el comportamiento de la variable dependiente o variable a pronosticar Y, en términos de un conjunto de variables independientes, llamadas predictores X_1, X_2,....,X_k de las que se tienen datos. La respuesta Y puede depender en forma directa de las variables predictoras, o de alguna variable derivada de las anteriores Z_i, en forma logarítmica, cuadrática, inversa o alguna otra función.

Se trata de encontrar la función

$$Y=\beta_0+\sum_{i}^{n}\beta_i*X_i$$

que mejor describa a la variable a pronosticar. A este proceso se lo conoce como selección de la mejor función de regresión, e implica un compromiso entre:

- buscar la función que mejor ajuste a los datos incluyendo todas las posibles combinaciones de los X_i

- encontrar la ecuación que incluyendo el menor número de X_i, aproxime a la variable a pronosticar de manera satisfactoria, ya que cada variable incluida en la ecuación deberá ser monitoreada.

Existen varios procedimientos estadísticos para seleccionar la función. Se podrían mencionar entre los más utilizados (1) la búsqueda de todas las funciones de regresión aplicando los criterios de R^2 o, s^2; (2) los métodos que tratan de quedarse con el menor número de variables en la regresión, ya sea por eliminación hacia atrás o la introducción de una variable por vez, como la regresión por pasos.

Considerar todos las posibles ecuaciones de regresión en el caso de los fenómenos meteorológicos es prácticamente imposible dado que el número de variables predictoras a considerar es de algunas decenas a miles de variables. (El número de ecuaciones posibles es 2^n, donde n es el número de variables)

También por simplicidad se considera que las Z_i son directamente las variables X aunque en ciertos casos puede ser, por ejemplo el gradiente de una de las variables datos.

El método de eliminación hacia atrás es más económico que el de buscar todas las ecuaciones posibles ya que calcula solamente la ecuación de todas las variables para luego ir eliminando aquellas menos significativas.

El método de regresión por pasos es, sin embargo, el más apropiado para este tipo de problemas donde hay un gran número de variables, altamente correlacionadas algunas de ellas, pues intenta encontrar la ecuación que incluya el mínimo número de variables necesarias (predictores) para explicar el comportamiento de la variable a pronosticar.

Este método va introduciendo variables de una en una, hasta que la ecuación de regresión es satisfactoria de acuerdo al criterio especificado. El orden de inserción lo determina el coeficiente de correlación parcial más grande, correspondiente a las variables que aún no están en la ecuación.

El procedimiento es el siguiente. Primero se selecciona la variable X más correlacionada con Y , por ejemplo X_1, y se calcula la regresión lineal

$$Y = F(X_1)$$

Se verifica si la variable es significativa. Si no lo es, el mejor modelo será

$$Y = \bar{Y}$$

En el caso en que la variable sea significativa, se debe seleccionar la segunda variable predictora a entrar al modelo de regresión evaluando los coeficientes de correlación parcial de las variables X_i que aun no están en la regresión. Matemáticamente es equivalente a calcular la correlación entre los residuos de la regresión $Y = f(X_1)$ y los residuos de cada una de las regresiones $X_i = g_i(X_1)$, correspondiente a la parte de las variables fuera de la regresión, explicada por la variable ya incluida en la ecuación. Si la variable X_2 es la que presenta el coeficiente de correlación parcial mayor, se obtiene la segunda ecuación de regresión lineal

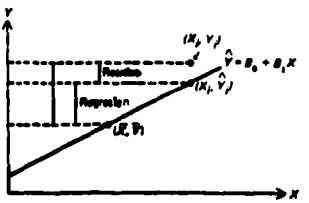
$$Y = f(X_1, X_2)$$

Para mejorar el nivel de significancia, debe haber aumentado el valor de R^2; también se examinan los valores de la F parcial de ambas variables predictoras de la ecuación. El menor de los valores de F se compara con un valor previamente estipulado para que una variable predictora permanezca en la ecuación. De acuerdo al valor de F, dicha variable es rechazada o aceptada. Este paso se realiza al terminar cada iteración.

Nuevamente se evaluan los coeficientes de correlación parcial, de las variables aún fuera de la ecuación; se introduce la siguiente variable, después de probar si su contribución a la ecuación de regresión pasa un valor mínimo de la prueba de F parcial. Se calcula la ecuación de regresión correspondiente. El proceso termina cuando ya no se puede rechazar ninguna variable y la siguiente candidata a entrar en la ecuación no pasa la prueba de F.

Este método es uno de los más económicos en cuanto a recursos computacionales, ya que solo introduce a aquellas variables que mejoran el modelo en cada etapa. La posibilidad de modificar los criterios de F para la introducción o el rechazo de las variables predictoras, modifica el número de estas en la ecuación, paralelamente a la confiabilidad de la regresión.

La calidad del modelo se mide con los parámetros estadísticos utilizados. A continuación se da la interpretación del coeficiente R_2 y del estadístico F.



$$Y_i - \bar{Y} = (Y_i - \hat{Y}_i) + (\hat{Y}_i - \bar{Y})$$

Figura A.1 - Componentes en la regresión lineal

El coeficiente R^2 es el coeficiente de correlación entre el valor estimado de la variable a pronosticar y el valor observado. Si R^2 vale uno, la estimación es perfecta, pero si el valor es cero solamente indica que no existe una relación lineal entre las variables.

Generalmente los modelos no son tan buenos predictores de la población total, como lo fueron de la muestra. El estadístico R^2 *ajustado* intenta reflejar la bondad de la ecuación en toda la población.

Geométricamente, figura A.1, es la proporción de la variación de los cuadrados de la variable dependiente explicada por el modelo.

$$R2 = 1 - \frac{\text{Suma cuadrados de los residuos}}{\text{Suma total de cuadrados}}$$

Para calcular el estadístico F se divide a la suma de los cuadrados por los grados de libertad correspondientes, obteniendose los valores cuadráticos medios. El cociente de los cuadrados medios de la regresión entre los del residuo tiene una distribución F, con p y N-p-1 grados de libertad, lo que permite calcular el grado de significancia de la prueba F. Si este valor es pequeño, la hipótesis de no relación lineal entre X y Y se rechaza.

$$F = \frac{\text{Valor cuadrático medio regresión}}{\text{Valor cuadrático medio residuo}}$$

permite obtener el nivel de significancia de F.

Apéndice B

Componentes principales y regresión lineal múltiple

En la regresión lineal múltiple, uno de los problemas más importantes es la multicolinearidad entre las variables predictoras, lo que provoca que algunos de los coeficientes estimados para la ecuación de regresión tengan demasiada varianza, provocando errores o inestabilidad en la ecuación de regresión.

Una de las técnicas para contrarrestar este problema es la regresión sobre los componentes principales (CP), donde se utilizan como variables predictoras a dichos componentes. Como los CP no están correlacionedos, no hay multicolinearidades entre ellos, con lo que se simplifica el cálculo de los coeficientes de la regresión. Si se utilizan todos lo CP, el modelo obtenido es equivalente al que se llega a partir de las variables originales, pero si se utilizan sólo algunos de ellos, los coeficientes son estimados evitando los problemas derivados de la multicolinearidad de las variables.

Regresión sobre los componentes principales

Considerando el modelo de regresión

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

donde **y** es un vector de n observaciones de la variable dependiente, **X** es una matriz de *nxp*, cuyo elemento (i,j) es el valor de la i-ésima observación de la variable predictora *j*, β es el vector de los coeficientes de la regresión y ϵ es el vector de error.

Por otro lado, los CP de cada observación están dados por

$$\mathbf{Z} = \mathbf{X}\,\mathbf{A}$$

donde el elemento (i,k) de **Z** es el peso del k_ésimo CP en la observación i; **A** es una matriz de *pxp*, cuyas columnas son los vectores principales de la matriz $\mathbf{X'X}$[1]

[1] Hasta aquí no se hizo ninguna restricción sobre la matriz X; esta puede estar formada por los datos originales, las desviaciones de los mismos respecto a su media o los valores normalizados correspondientes. La matriz **A** será entonces en algún caso, la matriz de covarianza o la matriz de correlación.

Como **A** es una matriz ortogonal, $\mathbf{X}\beta$ se puede escribir como $\mathbf{XAA'}\beta = \mathbf{Z}\gamma$ donde $\gamma = \mathbf{A'}\beta$ y la ecuación de regresión se transforma en

$$\mathbf{y} = \mathbf{Z}\gamma + \epsilon$$

Si se utilizan los CP para reducir el número de variables, el modelo queda

$$\mathbf{y} = \mathbf{Z}_m \gamma_m + \epsilon_m$$

donde $\mathbf{Z}_m$ es ahora una matriz de *nxm*.

Sin embargo hay algún problema en la elección de con cuales CP quedarse. Para eliminar la varianza debida a la multicolinearidad, dado que los CP explican en orden decreciente, la varianza de los datos, no correlacionada con los anteriores, es necesario quitar a los menos importantes, pero sin eliminar a aquellos que están altamente correlacionados con la variable dependiente. Esto no siempre es fácil de satisfacer.

El que un componente explique poca de la varianza, no significa que no sea importante en el modelo de regresión. Jolliffe [Jolli86] hace mención de un estudio en meteorología, la generación de monzones, donde cierto componente explica muy poco de la variación en x, pero es muy importante para la determinación de la variable dependiente.

Análisis de discriminantes y técnicas de agrupamiento con CP

El análisis discriminante trata de separar las observaciones en distintos grupos bien definidos y de encontrar las reglas que permitan asignar las futuras observaciones a uno de los grupos, según cierto criterio. Las técnicas de agrupamiento (cluster), también tratan de dividir las observaciones en grupos pero que no se conocen *a priori*.

En el análisis discriminante, el uso más obvio de los CP es sustituir las variables originales para disminuir el número de variables, y por lo tanto reducir la dimensionalidad del problema. Como los primeros componentes explican la mayor parte de la varianza, esto se puede utilizar para representar ahora los grupos en R^2 y tratar de visualizar la estructura de los grupos. Esto tiene inconvenientes, ya que supone que son los mismos CP los más importantes para todas las observaciones, y es probable que sean diferente CP los que caractericen cada grupo ya que la matriz de covarianza de cada uno puede tener distinta estructura.

En cuanto a encontrar la función discriminante, el problema es similar a la búsqueda de la ecuación de regresión, en el que la variable dependiente es un entero que indica la pertenecia al grupo. Si se eliminan CP para disminuir la dimensión del problema, hay

que recordar que el mejor subconjunto para encontrar la función discriminante no es necesariamente el que explica la mayor parte de la varianza. Aun con estos inconvenientes, el uso de los CP en vez de la variables originales es importante, ya que la contribución de cada CP es independiente de los otros debido a su ortogonalidad.

En las técnicas de agrupamiento, se requiere de una medida de la similaridad o diferencia entre los pares de observaciones de los miembros de cada grupo. esta medida puede por ejemplo ser una distancia euclidiana en el espacio de las variables. Si en vez de calcular esta medida sobre las variables originales, se lo hace sobre los CP más importantes, se obtiene una aproximación que permite determinar la pertenencia al grupo basandose en un grupo menor de variables.

Puede presentarse el mismo problema que en el análisis discriminante, donde algún CP haya sido eliminado previamente, aunque el problema aquí es menor ya que la matriz de covarianza se calcula sobre todas las observaciones, y la medida de similitud considerará las m componentes que se decidan introducir.

Otro problema a considerar es el uso de variables medidas en unidades no compatibles, lo que puede aparecer en las medidas de la distancia como efectos perturbadores, asignandole mayor importancia relativa a una que a otra. Es preferible normalizar las variables; en el caso de los CP, todos son vectores unitarios.

Apendice C

```
/* Programa para sacar un listado de los archivos de Presión, Tempera- */
/* tura o nivel de CD_METEO.                                           */
/* ******************************************************************* */
/* Autor:              Marco Antonio Sosa Chiñas.                      */
/* Adaptado por Isabel quintas                                         */
/* ******************************************************************* */

/*=====================================================================*/
/
Dependencias de implementación.                                        */
/                                                                      *
=====================================================================*/

/* ANSI C ------------------------------------------------------------ */
#include <stdio.h>     /* fopen() fread(), printf(), scanf().          */
#include <conio.h>     /* getch().                                     */
#include <math.h>      /* pow()                                        */
#include <string.h>    /* strcpy(), strcat().                          */
#include <ctype.h>     /* toupper()                                    */
/* Borland ----------------------------------------------------------- */
/* #include <dos.h>    / * getvect(),setvect()                         */
#include <process.h>   /* exit(), execl()                             */
/* Específicos
de implementación. ------------------------------------------------- */
#include "misc.h"      /* ubyte.                                       */
#include "ptz_grid.h"  /* get_grid(), interp(), PTZ_N_PTS.             */

/* ------------------------------------------------------------------ */
/* En PTZ estas están declaradas en el módlo PTZ_GUI.                 */
/* ------------------------------------------------------------------ */
char    cdun ;                      /* Unidad lógica del CD-ROM.      */
date_hr         grid_date ;                 /* Fecha de la malla.     */
int     ipl ;                       /* Nivel de presión.              */
char    cfc ;                       /* Identificacion de variable.    */
int  ndias ;                /* num dias consecutivos.                 */
/* Tamaños varios. ---------------------------------------------------- */
#define DRAW_MESH   512
#define LOND        1

float           grid[PTZ_N_PTS] ;   /* Malla desempaquetada.          */
float           r[DRAW_MESH] ;              /* Interpolaciones a long=cte. */
int             daily_mesh = 1 ;
char            map_fname[80] ;
/* ----------------  SLP   T500    T700    T850    Z200    Z500    Z700    Z850 */
static float    bs[] ={ 920.0, -55.0,   0.0, -71.0,   0.0, 4200.0, 2000.0, 1000.0,
                        750.0,   0.0, -144.0, -144.0, 9150.0, 4000.0, 1700.0,  144.0 } ;
static float    rs[] ={ 0.003, 0.001,   0.0, 0.002,   0.0, 0.035, 0.025, 0.010,
                        0.008,   0.0, 0.004, 0.004, 0.080, 0.040, 0.040, 0.040 } ;
static int      id_var = 0 ;
static char     *def_dd = ".:\\ptz\\diario\\" ;
static char     *def_md = ".:\\ptz\\mensual\\" ;
/* ------------------------------------------------------------------ */
static char     caller[80] ;  Ya no llama a CD_METEO de regreso*/
```

```
/                                                                          *
=========================================================================*/
/*   Rutinas   locales.                                                    */
=========================================================================  */
int halt ( char *msg ) ;
int  get_mesh ( int daily ) ;
int print_grid ( void ) ;


int main ( int argc, char *argv[] )
/* ======================================================================  */
/* Cuerpo del programa.                                                    */
/*=========================================================================*/
{int  dia, more ;          /* Control de continuación.                     */
/* ----------------------------------------------------------------------  */
/* Uso y fraseo de parámetros.                                             */
/* ----------------------------------------------------------------------  */
if ( argc < 9 )
  return halt ( "PTZ_TXT cd_drive año mes dia hora nivel {200|500|700|850|1013} variable {P|T|Z}
ndias\n" ) ;
/* ----------------------------------------------------------------------  */
cdun = argv[1][0] ;
/* ----------------------------------------------------------------------  */
if ( sscanf ( argv[2], "%d", &more ) < 1 )
  return halt ( "Error en año. Sintáxis.\n" ) ;
else
  if ( ( 46 > more ) || ( more > 89 ) )
    return halt ( "- Error en año. Fuera de rango [46..89].\n" ) ;
grid_date.year = more ;
/* ----------------------------------------------------------------------  */
if ( sscanf ( argv[3], "%d", &more ) < 1 )
  return halt ( "- Error en mes. Sintáxis.\n" ) ;
else
  if ( ( 1 > more ) || ( more > 12 ) )
    return halt ( "Error en mes. Fuera de rango [1..12].\n" ) ;
grid_date.month = more ;
/* ----------------------------------------------------------------------  */
if ( sscanf ( argv[4], "%d", &more ) < 1 )
  return halt ( "- Error en día. Sintáxis.\n" ) ;
else
  if ( ( 1 > more ) || ( more > 31 ) )
    return halt ( "- Error en día. Fuera de rango [1..31].\n" ) ;
dia = more ;
/* ----------------------------------------------------------------------  */
if ( sscanf ( argv[5], "%d", &more ) < 1 )
  return halt ( "- Error en hora. Sintáxis.\n" ) ;
else
  if ( ( 0 != more ) && ( more != 12 ) )
    return halt ( "- Error en hora. Datos disponibles para 0 y 12.\n" ) ;
grid_date.hour = more ;
/* ----------------------------------------------------------------------  */
if ( sscanf ( argv[6], "%d", &more ) < 1 )
  return halt ( "- Error en nivel. Sintáxis.\n" ) ;
else
  if ( ( more != 200 ) && ( more != 250 ) && ( more != 500 ) && ( more != 700 ) && ( more != 850
) && ( more != 1013 ) )
    return halt ( "- Error en nivel. Datos disponibles para 200, 500, 700, 850 o 1013.\n" ) ;
ipl = more ;
```

```
if ( ipl == 250 ) ipl = 200 ;
/* ------------------------------------------------------------------ */
cfc = argv[7][0] ;
cfc = toupper ( cfc ) ;
if ( ( cfc != 'P' ) && ( cfc != 'T' ) && ( cfc != 'Z' ) )
  return halt ( "- Error en tipo de variable. Disponibles P, T o Z.\n" ) ;
/* ------------------------------------------------------------------ */
sscanf ( argv[8], "%d", &more );
ndias = more ;
nmc_directories ( def_md, def_dd, def_dd ) ;

/* ------------------------------------------------------------------ */
/* Loop de despliegue.                                                */
/* ------------------------------------------------------------------ */
for ( grid_date.day =dia; grid_date.day<=dia+ndias; grid_date.day++)
  {
   ipl = 850 ;
   if ( get_mesh ( daily_mesh ) )
    {                    /* Si lo encontró. ------      */
     switch ( print_grid2() )
      {
       case 0:       /* Ok.          */                  break ;
       case 1:    printf ( "- Error al escribir\n" ) ;  break ;
      } /* switch */
    } /* if */

   if ( grid_date.day == 31 ) break;
  } /* for  */
 return halt ( NULL ) ;
} /* main() */

int kbd_keycode ( void )
{
int       key ;
key = getch() ;
if ( key == 0 ) key = getch() ;
return key ;
}

int halt ( char *msg )
/* ================================================================== */
/*Manda mensaje (modo texto) y regresa 0.                             */
/* ================================================================== */
{
if ( msg != NULL )
  {
  printf ( msg ) ;
  printf ( "\nOprima cualquier tecla para continuar ..\n" ) ;
  (void) kbd_keycode() ;
  }
return 0 ;                              /* Nunca llega aqui.          */
/* ------------------------------------ ------------------------------ */
} /* halt() */

int get_mesh ( int daily )
/* ================================================================== */
/* Pide las características de la malla diario o mensual (daily = =0).  */
/* ================================================================== */
```

```
{
int     nerr ;                  /* Código de error.                    */
int     back_day ;              /* Día.                                */
int     pl ;                    /* Nivel de presión.                   */
/* ------------------------------------------------------------------ */
switch ( cfc )
 {
 case 'P': id_var = ( daily ) ? 8 : 0 ;                    break ;
 case 'T': if ( daily )
             id_var = (ipl==700) ? 10: (ipl==850) ? 11: -1 ;
           else
             id_var = (ipl==500) ? 1 : (ipl==850) ?  3: -1 ;
           break ;
 case 'Z': if ( daily )
             id_var = (ipl==200) ? 12: (ipl==500) ? 13: (ipl==700) ? 14: 15 ;
           else
             id_var = (ipl==500) ?  5: (ipl==700) ?  6: (ipl==850) ?  7: -1 ;
           break ;
 }
if ( id_var < 0 )
 {
 printf ( "- Combinación no disponible\n" ) ;
 return 0 ;
 }
base = ba[id_var] ;
resol = rs[id_var] ;
if ( daily == 0 )
 {
 back_day = grid_date.day ;
 grid_date.day = 0 ;
 }
pl = ( cfc == 'P' ) ? 1013 : ipl ;      /* Nivel de presión.           */
nerr = get_grid ( PTZG_FIND, &grid_date, pl, cfc, grid, cdun ) ;
if ( daily == 0 ) grid_date.day = back_day ;    /* Regresa valor.      */
printf ("salida get_drid  %d en dia  %d/n", nerr, grid_date.day);
if ( nerr )
 {
 if ( nerr == PTZG_BADFILE )
   printf ( "- No se puedo cargar el archivo.\n  Revise que CD_METEO esté instalado correctamente.\n"
) ;
 else if ( nerr == PTZG_EOF )
   printf ( "- Fin de archivo inesperado.\n  Revise que CD_METEO esté instalado correctamente.\n" ) ;
 else if ( nerr == PTZG_MISMATCH )
   printf ( "- Malla solicitada y leida no concuerdan.\n" ) ;
 /*
 else if ( nerr == PTZG_NOGRID )
   printf ( "- No se encontró la malla solicitada.\n" ) ;
 */
 else
   printf ( "- Los datos para ese día forman parte de las lagunas en la base de datos orig.,\n",
            " por lo que no están disponibles.\n" ) ;
 return 0 ;
 }
return 1 ;
/* ------------------------------------------------------------------ */
} /* get_mesh_char() */
```

```
int print_grid2 ( void )
/* ================================================================ */
/* Imprime la malla cargada de CD_METEO.                            */
/* ================================================================ */
{
int             i, j, jj, k, kimp ;     /* Contadores.              */
int             nx, ny ;                /* Contadores.              */
FILE            *fp ;                   /* Archivo.                 */
float           xl, xr, y, yt ;
float           x, dx, dy ;
/* ---------------------------------------------------------------- */
/* Se agregan tres arreglos  xlong, ylat, npuntos para dar las coord.  */
/* de los puntos de la malla irregular                              */
float xlong[] = {-100,-99,-102,-99,-101.5,-104,-98,-101,-104,-107,
                 -97, -101,-105,-109};
float ylat[] = { 18, 20, 20, 22, 22, 22, 25, 25, 25, 25, 29, 29, 29, 29};
int nump[] = { 1, 2, 3, 4, 4};

printf ( " dia   %d\n", grid_date.day );
/* ---------------------------------------------------------------- */
/* Carga la malla e interpola.                                      */
/* ---------------------------------------------------------------- */
  fp = fopen ( "PTZ.LIS", "a" ) ;
  if ( fp == NULL ) return 1 ;

fprintf ( fp, "%c %4d %d/ %02d/ %2d %2d:00\n", cfc, ipl, grid_date.day, grid_date.month,
grid_date.year, grid_date.hour ) ;
k = 0 ;
for ( i = 0 ; i < 5 ; i++ )
  {
   for (jj = 0 ; jj < nump[i] ; jj++ )
    {
     x = xlong [k+jj] ;
     y = ylat [k+jj] ;
     dx = 0 ;     dy = dx ;

     interp_grid ( r, 5, 1, 1, 1, x, y, dx, dy, grid ) ;
     fprintf ( fp, "%9.2f", r[0] ) ;
/*         for ( j = 0, kimp = 1 ; j < nump[i] ; j++ ) { /*
/*       if ( kimp < 7 ) kimp++ ;else{      /*
/*       kimp = 1 ;   } if ( kimpresion . ) /* for ( j .. */
    } /* for (jj... */
   fprintf ( fp, "\n" ) ;
   k = k + nump [i] ;
  } /* for ( i .. */
  fprintf ( fp, "\n" ) ;
  fclose ( fp ) ;
return 0 ;

/* ---------------------------------------------------------------- */
} /* print_grid2() */
/* **************************************************************** */
/* Implementación del módulo PTZ_GRID.                              */
/* Para manejo de mallas e interpolaciones.  Interfaz en PTZ_GRID.H.  */
/* **************************************************************** */
/* Dependencias de implementación.                                  */
/* ================================================================ */
/* ANSI C ---------------------------------------------- */
#include <stdio.h>      /* fopen(), fread(), fseek().               */
```

```
#include <math.h>      /* floor().                                          */
/* Específicos de implementación. ------------------------------------ */
#include "misc.h"      /* getnames(), monthind() nearest_dailyind().        */
#include "ptz_grid.h"  /* interfaz.                                         */
static unsigned ubuff[PTZ_N_PTS] ;


int get_grid ( int iopt, date_hr *sel_date, int ipl, char cfc,
               float *grid, char cdun )
/* ================================================================== */
/* "iopt"     = Load option:                                          */
/*              PTZG_FIND -> Read record specified by "*sel_date",    */
/*                           "ipl", "cfc"                             */
/*              PTZG_NEXT -> Read next record (sequential access)     */
/* "sel_date" = Selected date (Year without century, month, day, hour)*/
/*              (day = = 0 for monthly mean grid), hour in GMT (00/12)*/
/* "ipl"      = Pressure level in mb (surface = = 1013)               */
/* "cfc"      = Parameter type:                                       */
/*              'Z' = height;                                         */
/*              'T' = temperature;                                    */
/*              'P' = pressure;                                       */
/*              'U' = u wind component;                               */
/*              'V' = v wind component.                               */
/* "grid"     = Unpacked array of grid point values. [59*19]          */
/* "cdun"     = CDROM drive letter (e.g., cdun = 'L')                 */
/* Return value:                                                      */
/*              PTZG_OK          = okay, no error                     */
/*              PTZG_BADFILE = cannot find requested grid (iopt = 0)  */
/*              PTZG_EOF     = end of file read                       */
/*              PTZG_MISMATCH= mismatch between requested and returned*/
/*                             grid                                   */
/*              PTZG_NOGRID  = grid not available                     */
/* ================================================================== */
{
static long    itot ;
static char    dataname[50] ;
char           indname[50] ;
FILE           *fp ;
int            i, j, k ;
/* ------------------------------------------------------------------ */
/* Determina archivos e índice (o número de record).                  */
/* ------------------------------------------------------------------ */
if ( iopt = = PTZG_NEXT )
  itot + + ;
else
  {
  getnames ( dataname, indname, sel_date->day, ipl, cfc, cdun ) ;
  if ( sel_date->day < 1 )
    {
    if ( (cfc = = 'T') && (sel_date->month = = 4) && (sel_date->year = = 63) )
      return PTZG_NOGRID ;
    else
      itot = monthind ( sel_date->year, sel_date->month, ipl, cfc ) ;
    }
  else
    if ( sel_date->day < = 31 )
      itot = nearest_dailyind ( indname, sel_date, ipl, cfc ) ;
  if ( itot < 0L )
```

```
    return PTZG_NOGRID ;
  } /* else */
/* ------------------------------------------------------------ */
/* Carga la información.                                        */
/* ------------------------------------------------------------ */
fp = fopen ( dataname, "rb" ) ;
if ( fp == NULL ) return PTZG_EOF ;   /* Error de apertura.     */
if ( fseek ( fp, itot*PTZ_RECSIZE, SEEK_SET ) )
  {
  fclose ( fp ) ;
  return PTZG_NOGRID ;                /* Error de posicionamiento. */
  }
if ( fread ( ubuff, 2, PTZ_N_PTS, fp ) < PTZ_N_PTS )
  {
  fclose ( fp ) ;
  return PTZG_EOF ;                   /* Error de lectura.      */
  }
fclose ( fp ) ;                       /* Cierra archivo.        */
/* ------------------------------------------------------------ */
/* Desempaca la malla.                                          */
/* ------------------------------------------------------------ */
for ( k = j = 0 ; j < lat_n ; j++ )
  {
  for ( i = 0 ; i < lon_n ; i++, k++ )
    grid[k] = ubuff[k] * resol + base ;
  } /* for ( j .. */
return PTZG_OK ;
} /* get_grid() */

void interp_grid ( float e[], int lond, int latd, int nlon, int nlat,
                   float xmin, float ymin, float dlon, float dlat, float a[] )
/* ============================================================ */
/*....Interp a lat-lon array e from an NMC array a              */
/*....Written by Roy Jenne at NCAR, feb 1974                    */
/*....Modified by Jorge Sanchez-Sesma, june 1991                */
/*    Modified by Marco A. Sosa, june 1992.                     */
/* ------------------------------------------------------------ */
/*  e     = output array (dimension [lond,latd])                */
/*  lond  = number of longitude points                          */
/*  latd  = number of latitude points                           */
/*  nlat  = number of latitude points to actually return        */
/*  nlon  = number of longitude points to actually return       */
/*  ymin  = minimum latitude in degrees                         */
/*  xmin  = minimum longitude in degrees                        */
/*  dlat  = latitude interval in degrees                        */
/*  dlon  = longitude interval in degrees                       */
/*  a     = input array                                         */
/* ============================================================ */
{
int           i, j ;                /* Contadores.              */
int           k ;                   /* Offset al prim. elem e[j+i] */
float         xx, yy ;              /* Coordenadas relativas.   */
int           n, m ;                /* ^ a entero.              */
float         dx, dy ;              /* Exceso sobre ".          */
float         dxx, dyy ;            /* -1/4 de complemento de ". */
int           na, nb, nc, nd ;      /* Vecinos en altura.       */
int           ma,   mc, md ;        /* Vecinos en altura.       */
float         aa, ab, ac, ad ; /* Interpolaciones intermedias */
```

```
unsigned        outside, outside_y ;      /* Codigo de puntos fuera.          */
/* ------------------------------------------------------------------------ */
/* Ahora si calcula para la malla lon-lat deseada.                          */
/* ------------------------------------------------------------------------ */
dlon /= lon_d ;
dlat /= lat_d ;
yy = ( ynlin - lat_i ) / lat_d ;
for ( j = k = 0 ; ( j < nlat ) && ( j < latd ) ; j++, k+=lond, yy+=dlat )
  {
  n = (int) floor ( yy ) ;
  outside_y  = ( n < -2 ) ? 0xffff : ( n < -1 ) ? 0xfff0 : ( n < 0 ) ? 0x00ff : ( n < 1 ) ? 0x000f :
0x0000 ;
  outside_y |= ( n >lat_n)? 0xffff : (n>lat_n-1)? 0xfff0 : (n>lat_n-2)?0xff00 : (n>lat_n-3)?0xf000 :
0x0000 ;
  xx = ( xmin - lon_i ) / lon_d ;
  for ( i = 0 ; ( i < nlon ) && ( i < lond ) ; i++ )
    {
    m = (int) floor ( xx ) ;
    dx = xx - m ;
    dy = yy - n ;
    dxx = 0.5 * ( 1.0 - dx ) ;
    dyy = 0.5 * ( 1.0 - dy ) ;
    ma = m - 1 ;
    mc = ( mc = m + 1 ) + 1 ;
    na = lon_n * ( n - 1 ) ;
    nc = ( nc = ( nb = na + lon_n ) + lon_n ) + lon_n ;
    outside = outside_y ;
    outside |= ( m < -2 ) ? 0xffff : ( m < -1 ) ? 0x7777 : ( m < 0 ) ? 0x3333 : ( m < 1 ) ? 0x1111
: 0x0000 ;
    outside |= ( m >lon_n)? 0xffff : (m>lon_n-1)? 0xeeee : (m>lon_n-2)?0xcccc :
(m>lon_n-3)?0x8888 : 0x0000 ;
    if ( ! outside )
      /* ---------------------------------------------------------------- */
      /* Do the 16 pt Bessel interp scheme.                               */
      /* ---------------------------------------------------------------- */
      {aa    =  a[m+na]   + dx*(a[mc+na]-a[m+na]    + dxx*(2*a[m+na]-a[ma+na]-a[mc+na]
+dx*(3*(a[mc+na]-a[m+na])+a[ma+na]-a[md+na])))  ;
      ab    =  a[m+nb]   + dx*(a[mc+nb]-a[m+nb]    + dxx*(2*a[m+nb]-a[ma+nb]-a[mc+nb]
+dx*(3*(a[mc+nb]-a[m+nb])+a[ma+nb]-a[md+nb])))  ;
      ac    =  a[m+nc]   + dx*(a[mc+nc]-a[m+nc]    + dxx*(2*a[m+nc]-a[ma+nc]-a[mc+nc]
+dx*(3*(a[mc+nc]-a[m+nc])+a[ma+nc]-a[md+nc])))  ;
      ad    =  a[m+nd]   + dx*(a[mc+nd]-a[m+nd]    + dxx*(2*a[m+nd]-a[ma+nd]-a[mc+nd]
+dx*(3*(a[mc+nd]-a[m+nd])+a[ma+nd]-a[md+nd])))  ;
      e[k+i] = ab + dy * ( ac-ab + dyy * ( 2*ab-aa-ac + dy * ( 3*(ac-ab) + aa-ad ) ) ) ;
      } /* if */
    else
      {
      unsigned    out_x, out_y = 0 ;
      out_x = ( outside & 0x000f ) ;
      if ( out_x == 0x000f )
          out_y |= 0x0001 ;
      else
          aa = interp ( out_x, dx, dxx, a[ma+na], a[m+na], a[mc+na], a[md+na] ) ;
      out_x = ( outside & 0x00f0 ) >> 4 ;
      if ( out_x == 0x000f )
          out_y |= 0x0002 ;
      else
          ab = interp ( out_x, dx, dxx, a[ma+nb], a[m+nb], a[mc+nb], a[md+nb] ) ;
```

```
    out_x = ( outside & 0x0f00 ) >> 8 ;
    if ( out_x == 0x000f )
        out_y |= 0x0004 ;
    else
        ac = interp ( out_x, dx, dxx, a[ma+nc], a[m+nc], a[mc+nc], a[md+nc] ) ;
    out_x = ( outside & 0xf000 ) >> 12 ;
    if ( out_x == 0x000f )
        out_y |= 0x0008 ;
    else
        ad = interp ( out_x, dx, dxx, a[ma+nd], a[m+nd], a[mc+nd], a[md+nd] ) ;
    e[k+i] = interp ( out_y, dy, dyy, aa, ab, ac, ad ) ;
    } /* else */
  } /* for ( i .. */
 } /* for ( ) .. */
} /* interp_grid() */

float interp ( unsigned out, float di, float dd2, float a, float b, float c, float d )
/* ========================================== */
/* Interpola entre los cuatro puntos igualmente espaciados, con valores */
/* "a", "b", "c", y "d", a una distancia "di" a la derecha del segundo  */
/* punto. La variable "dd2" debe valer "(1-di)/2". Dependiendo de los   */
/* puntos invalidos (que vienen en los bits de "out") se interpola con  */
/* dos parábolas (continuidad C1), con la izquierda o derecha, con rec- */
/* tas derecha o izquierda (extrapolando) o constante (extremos).       */
/* ========================================== */
{
float   intrp ;
switch ( out )
 {
 case  1: intrp = c - dd2 * ( d-b - (1.0-di) * (d+b-2*c) ) ;      break ;
 case  3: intrp = c - (1.0-di ) * ( d-c ) ;                      break ;
 case  7: intrp = d ;                                            break ;
 case 15: intrp = 0.0 ;                                          break ;
 case 14: intrp = a ;                                            break ;
 case 12: intrp = b + di * ( b-a ) ;                             break ;
 case  8: intrp = b + di * ( c-a + di * (c+a-2*b) ) / 2 ;        break ;
 case  0:
 default: intrp = b + di * ( c-b + dd2 * ( 2*b-a-c + di * ( 3*(c-b)+a-d ) ) ) ;
 } /* switch */
return intrp ;
} /* interp() */

/* ********************************************************************** */
/* Implementación del módulo MISC.                                        */
/* Rutinas varias para control de acceso a la información de NMC.         */
/* Interfaz en MISC.H.                                                    */
/* ********************************************************************** */
/* Tomado de MISC.FOR del paquete de programas del NMC.                   */

#include <stdio.h>      /* FILE, fopen(), fread(), NULL.                  */
#include <string.h>     /* strcpy(), strcat().                            */
#include "chdate.h"     /* date_hr, tojoul().                             */
#include "misc.h"       /* Interfaz.                                      */

/* ------------------------------------------------------------ */
/* Tamaño de los records de los archivos de indice              */
/* ------------------------------------------------------------ */
#define INDX_REC_SIZE           4030
```

```
#define         MAX_SRCH_HALFDAY    3

int     misc_err        = 0 ;   /* Ultimo error.                    */
/* ------------------------------------------------------------ */
/* Mensajes de error.                                           */
/* ------------------------------------------------------------ */
static char     *err_msg[]      = { "Success",
                                "Level, type combination is incorrect.",
                                "Not an index file.",
                                "Date is previous to beginning of data.",
                                "Date is after end of data.",
                                "Data for this particular date is missing.",
                                "      } ;
/* ------------------------------------------------------------ */
/* Para formar nombres de archivo.                              */
/* ------------------------------------------------------------ */
static char     dm_dir[]        = ":\\data\\monthly\\" ;        /* def. */
static char     dd_dir[]        = ":\\data\\daily\\" ;          /* def. */
static char     di_dir[]  = ":\\index\\daily\\" ;       /* def. */
static char     *m_dir          = dm_dir ;                      /* Act. */
static char     *d_dir          = dd_dir ;                      /* Act. */
static char     *i_dir          = di_dir ;                      /* Act. */
static char     *slp            = "slp" ;
static char     *uv             = "uv" ;
static char     *dat            = ".dat" ;
static char     *ind            = ".ind" ;
/* ------------------------------------------------------------ */
/* Para manejo de archivos, header y bitmap.                    */
/* ------------------------------------------------------------ */
static ubyte    btchr[INDX_REC_SIZE] ;
static ubyte    beg_mask[8]     = { 0xFF, 0x7F, 0x3F, 0x1F, 0x0F, 0x07, 0x03, 0x01 } ;
static ubyte    end_mask[8]     = { 0x80, 0xC0, 0xE0, 0xF0, 0xF8, 0xFC, 0xFE, 0xFF } ;
static char     inamesrv[50]    = "" ;
static date_hr  first_date      = { 1, 1, 1, 0 } ;      /* Primer fecha valid.  */

void nmc_directories ( char *nm_dir, char *nd_dir, char *ni_dir )
/* ============================================================ */
/* Cambia los directorios del NMC CD ROM:                       */
/*      "nm_dir"        Directorio para datos mensuales.        */
/*      "nd_dir"        Directorio para datos diarios.          */
/*      "ni_dir"        Directorio para indices diarios.        */
/* debe terminar en '\'. Si alguno de ellos es NULL cambia al default.  */
/
============================================================*/
{
m_dir = ( nm_dir != NULL ) ? nm_dir : dm_dir ;
d_dir = ( nd_dir != NULL ) ? nd_dir : dd_dir ;
i_dir = ( ni_dir != NULL ) ? ni_dir : di_dir ;
} /* nmc_directories() */

char * indx_error_msg ( void )
/* ============================================================*/
/* Regresa apuntador a cadena de caracteres con el error más reciente.  */
/
============================================================*/
{
return err_msg[misc_err] ;
} /* indx_error_msg() */
```

```
#pragma argsused
long dailyind ( char indname[], date_hr *sel_date, int ipl, char cfc )
/* ============================================================== */
/* Regresa el número del índice (o registro) asociado al día "sel_date"   */
/* buscado en la tabla de índices diarios cuyo nombre es "indname".       */
/* "cfc" debe contener la variable deseada:                               */
/*      P       :       Presión.                                          */
/*      T       :       Temperatura.                                      */
/*      Z       :       Elevación.                                        */
/*      U       :       Velocidad en dir X (malla octagonal).             */
/*      V       :       Velocidad en dir Y (malla octagonal).             */
/* "itot" regresa el número de bits encendidos hasta esa fecha, es de-    */
/* cir, el número de registro en el archivo de datos; o un valor nega-    */
/* tivo en caso de error.                                                 */
/* ============================================================== */
{
long    itot, start, ndday ;
FILE    *index_file ;           /* Archivo de índices.                    */
/* Estas que siguen no se para que se usen. .....                         */
/* int   inpl ;  */              /* pressure level code.                   */
/* int   ntyp ;  */              /* NCAR function code.                    */
/* int   nent ;  */              /* Number of entries.                     */
/* ..........                                                             */
uword   byte_no ;               /* Hasta ese byte hay que revisar.        */
uword   bit_mask ;              /* Máscara del último byte.               */
uword   i ;                     /* Contador.                              */
ubyte   test ;                  /* Compara de ese día en particular.      */
/* ---------------------------------------------------------------- */
/* Open bitmap file and get the pertinent info, i.e. starting data  */
/* record, starting year of index, pressure level, NCAR parameter code, */
/* # of entries in bitmap, and finally the bitmap itself.           */
/* ---------------------------------------------------------------- */
misc_err = 0 ;                  /* Limpia errores previos.                */
index_file = fopen ( indname, "rb" ) ;  /* Binario, 1 rec/4030 bytes.     */
if ( index_file == NULL )
  /* No se pudo abrir el archivo, tal vez inválido. ----------------  */
  {
  misc_err = 1 ;
  itot = INDX_ERRDURREA ;
  return itot ;
  }
start = fread ( bfchr, 1, INDX_REC_SIZE, index_file ) ;
if ( start < (long)(INDX_REC_SIZE) )
/*
if ( fread ( bfchr, INDX_REC_SIZE, 1, index_file ) < 1 )
*/
  /* No se pudo leer el archivo, tal vez mal nombre. ----------------  */
  {
  misc_err = 2 ;
  itot = INDX_ERRDURREA ;
  fclose ( index_file ) ;   /* FORTRAN lo cierra automat. al salir.  */
  return itot ;
  }
fclose ( index_file ) ;             /* FORTRAN lo cierra automat. al salir. */
strcpy ( inameprv, indname ) ;
/* ---------------------------------------------------------------- */
/* Traduce encabezado.                                                    */
/* ---------------------------------------------------------------- */
```

```
start = inv_long ( bfchr ) - 1L ;           /* Starting record (0).    */
frst_date.year = inv_int ( bfchr + 4 ) ;  /* First year w/data      */
if ( sel_date->year < frst_date.year )
  {
  /* Fecha anterior a la de datos. ---------------------------------    */
  misc_err = 3 ;
  itot = INDX_BEFOREDATA ;
  return itot ;
  }
/* Estas que siguen no se para que se usan. -------------------------          */
/* inpl = inv_int ( bfchr + 6 ) ;          */      /* pressure level code.  */
/* ntyp = inv_int ( bfchr + 8 ) ;          */      /* NCAR function code.        */
/* nent = inv_int ( bfchr + 10 ) ;         */      /* Number of entries.    */
/* -----------------------------------------------------------------          */
/* Determina el bit correspondiente al dia seleccionado "sel_date".          */
/* -----------------------------------------------------------------          */
ndday = ( tojul(sel_date) - tojul(&frst_date) ) << 1 ;
if ( sel_date->hour > 6 ) ndday++ ;
byte_no = (uword) ( ndday >> 3 ) ;       /* Hasta ese byte y ..             */
i = (uword) ( ndday & 7L ) ;               /* .. este bit hay que revisar.    */
bit_mask = end_mask[i] ;                            /* Máscara del último byte a        */
                                            /* revisar, iniciando con MSB     */
if ( (12+byte_no) >= INDX_REC_SIZE )
  {
  /* Fecha posterior a la de datos. ---------------------------------    */
  misc_err = 4 ;
  itot = INDX_AFTERDATA ;
  return itot ;
  }
test = 0x01 << ( 7 - i ) ;                  /* Inicia con MSB.                 */
if ( ! ( bfchr[12 +byte_no] & test ) )
  {
  /* No hay para ese día en particular. ------------------------------    */
  misc_err = 5 ;
  itot = INDX_MISSING ;
  return itot ;
  }
/* -----------------------------------------------------------------          */
/* Loop para contar bits encendidos.                                          */
/* -----------------------------------------------------------------          */
itot = 0 ;
for ( i = 0 ; i <= byte_no ; i++ )
  if ( i == byte_no )
    itot += ktab ( bfchr[12+i] & bit_mask ) ;
  else
    itot += ktab ( bfchr[12+i] ) ;
itot-- ;
if ( cfc == 'U' ) itot = 2*itot ;
if ( cfc == 'V' ) itot = 2*itot + 1 ;
itot += start ;
return itot ;                               /* Indice relativo.                */
/* -----------------------------------------------------------------          */
} /* dailyind() */

long nearest_dailyind ( char indname[], date_hr *sel_date, int ipl, char cfc )
/                                                                               *
==================================================================================*/
/* Regresa el número del índice (o registro) asociado al dia "sel_date"        */
```

```
/* buscado en la tabla de indices diarios cuyo nombre es "indname". Si */
/* no existe, busca el mas cercano.                                    */
/* "cfc" debe contener la variable deseada:                            */
/*       P       :       Presión.                                      */
/*       T       :       Temperatura.                                  */
/*       Z       :       Elevación.                                    */
/*       U       :       Velocidad en dir X (malla octagonal).         */
/*       V       :       Velocidad en dir Y (malla octagonal).         */
/* Regresa el número de bits encendidos hasta esa fecha, es decir, el  */
/* número de registro en el archivo de datos; o negativo si es error   */
/* ================================================================== */
{
long    indx ;
char    srch = 0x03 ;
int     del = 1 ;
date_hr         t_date ;
/* ------------------------------------------------------------------ */
indx = dailyind ( indname, sel_date, ipl, cfc ) ;
if ( indx >= 0 ) return indx ;                  /* Si lo encontró.    */
chdate ( sel_date, &t_date, del*12 ) ;          /* Busca adelante.    */
if ( INDX_MISSING != indx )     return indx ;   /* No se puede.       */
/* ------------------------------------------------------------------ */
do {
 indx = dailyind ( indname, &t_date, ipl, cfc ) ;
 if ( indx >= 0 ) break ;                       /* Ya lo encontró.    */
 if ( INDX_MISSING != indx )                    /* Incorrecto:        */
  srch &= ( srch & 0x10 ) ? ~0x02 : ~0x01 ;     /*  Evita esa direc.  */
 switch ( srch )
  {
  case 0x03:
  case 0x02:  /* Busca ahora atras. ------------------------------ */
              chdate ( sel_date, &t_date, -del*12 ) ;
              srch ^= 0x10 ;
              break ;
  case 0x13:
  case 0x11:  /* Busca ahora adelante. --------------------------- */
              del++ ;
              chdate ( sel_date, &t_date, del*12 ) ;
              srch ^= 0x10 ;
              break ;
  case 0x01:  /* Sigue buscando adelante. ------------------------ */
              chdate ( sel_date, &t_date, del*12 ) ;
              del++ ;
              break ;
  case 0x12:  /* Sigue buscando atras. --------------------------- */
              chdate ( sel_date, &t_date, -del*12 ) ;
              del++ ;
              break ;
  } /* switch */
 } while ( ( srch & 0x03 ) && ( del < MAX_SRCH_HALFDAY ) ) ;
/* ------------------------------------------------------------------ */
if ( indx >= 0 )                                /* Si es válido:      */
 chdate ( &t_date, sel_date, 0 ) ;              /* - Actualiza fecha. */
return indx ;
/* ------------------------------------------------------------------ */
} /* nearest_dailyind() */

long inv_long ( ubyte b[] )
```

```
/* ================================== */
/* Invierte los bytes de "b" para formar una cantidad long. */
/* ================================== */
{
unsigned long   rv ;
rv = ( ( ( ( b[0] << 8 ) | b[1] ) << 8 ) | b[2] ) << 8 ) | b[3] ;
return (long) rv ;
} /* inv_long() */

int inv_int ( ubyte b[] )
/* ================================== */
/* Invierte los bytes de "b" para formar una cantidad int. */
/* ================================== */
{
unsigned int    rv ;
rv = ( b[0] << 8 ) | b[1] ;
return (int) rv ;
} /* inv_int() */

#define MO_01_46      0
#define MO_10_62      201
#define MO_08_63      211

long monthind ( int iyr, int imo, int ipl, char cfc )
/* ================================== */
{
long    offset,
        max,
        month,
        ktot ;
/* Determina el "offset" del inicio de los datos, a partir de Jan/46.  */
/* (Oct/62 ->201, Aug/63 ->211).                                       */
/* ------------------------------------------------------------------ */
misc_err = 0 ;
if ( cfc == 'P' )
  {
  offset = ( ipl == 1013 ) ? MO_01_46 : INDX_WRONGCOMB ;
  max    = 522 ;
  }
else
  switch ( ipl )
    {
    case 850:
      switch ( cfc )
        {
        case 'Z':       offset = MO_10_62 ;   max = 321 ;     break ;
        case 'T':       offset = MO_10_62;    max = 321 ;
                if ( ( iyr == 63 ) && ( imo == 4 ) )
                  offset = INDX_MISSING ;             break ;
        case 'U':
        case 'V':       offset = MO_08_63 ;   max = 578 ;     break ;
        default:        offset = INDX_WRONGCOMB ;             break ;
        } /* switch */              break ;
    case 700:
      switch ( cfc )
        {
        case 'Z':       offset = MO_10_62;    max = 321 ;     break ;
        default:        offset = INDX_WRONGCOMB ;             break ;
```

```
        } /* switch */              break ;
  case 500:
    switch ( cfc )
        {
        case 'Z':       offset = MO_01_46 ;   max = 522 ;      break ;
        case 'T':       offset = MO_10_62 ;   max = 321 ;
                if ( ( iyr = = 63 ) && ( imo = = 4 ) )
                offset = INDX_MISSING ;              break ,
        default:        offset = INDX_WRONGCOMB ;            break :
        } /* switch */              break ;
  default:              offset = INDX_WRONGCOMB ;  .
  } /* switch */
/* ---------------------------------------------------------------        */
/* Ahora calcula el número de registros a avanzar.                        */
/* ---------------------------------------------------------------        */
if ( offset < 0 )
  {
  misc_err = ( offset = = INDX_WRONGCOMB ) ? 1 : 5 ;
  return offset ;         /* Código de error, regresa negativo.    */
  }
month = ( iyr - 46 ) * 12 + imo - 1 ;
itot = month - offset ;
if ( itot < 0 )
  {
  misc_err = 3 ;
  return INDX_BEFOREDATA ;   /* Es antes de datos, regresa negativo.  */
  }
if ( itot > max )
  {
  misc_err = 4 ;
  return INDX_AFTERDATA ;    /* Es despues de datos, regresa neg.     */
  }
if ( cfc = = 'U' ) itot = 2*itot ;
if ( cfc = = 'V' ) itot = 2*itot + 1 ;
return itot ;                     /* Regresa número de registro.          */
/* ---------------------------------------------------------------        */
} /* monthind() */

void getnames ( char dataname[], char indname[], int idy, int ipl, char cfc, char cdun )
/* = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =      */
/* Forma el nombre del archivo de datos "dataname" (y de índice, si es   */
/* diario "indname", con "idy" entre 1 y 31), para el nivel "ipl", para  */
/* la variable "cfc" ('P', 'T', 'U', 'V', 'Z'), estando los datos en el  */
/* driver "cdun" ('D', 'E', etc).                 */
/                                                                          *
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
= = = = = = = = = = = = = = = = = = = = = =    */
{
if ( ( 1 > idy ) || ( idy > 31 ) )
  { /* Menusal -------------------------------------------------        */
  switch ( cfc )
     {
     case 'P': sprintf(dataname,"%c%s%s%s",  cdun,m_dir,slp,   dat);  break ;
     case 'U':
     case 'V': sprintf(dataname,"%c%s%s%d%s",cdun,m_dir,uv,ipl,  dat);  break ;
     default:  sprintf(dataname,"%c%s%c%d%s",cdun,m_dir,cfc,ipl,dat);
     } /* switch */
  } /* if ( ( 1 > idy ) || ( idy > 31 ) ) */
else
```

```
  { /* Diario. ------------------------------------------------------ */
  switch ( cfc )
    {
    case 'P': sprintf(dataname,"%c%s%s%s",  cdun,d_dir,sip,   dat);
              sprintf(indname, "%c%s%s%s",  cdun,i_dir,sip,   ind); break ;
    case 'U':
    case 'V': sprintf(dataname,"%c%s%s%d%s",cdun,d_dir,uv,ipl,  dat);
              sprintf(indname, "%c%s%s%d%s",cdun,i_dir,uv,ipl,  ind); break ;
    default:  sprintf(dataname,"%c%s%c%d%s",cdun,d_dir,cfc,ipl,dat);
              sprintf(indname, "%c%s%c%d%s",cdun,i_dir,cfc,ipl,ind);
    } /* switch */
  } /* else */
/* ------------------------------------------------------------------ */
} /* getnames() */

unsigned unpack_bits ( ubyte *buffer, int offset, int n_bits )
/*=================================================================*/
/* Desempaca bits, según el formato del NMC, tomándolos del arreglo  */
/* "buffer", iniciando con un desplazamiento "offset", y tomando en  */
/* cuenta "n_bits". Ejemplo:                                         */
/*  | buffer[0]          | buffer[1]          | buffer[2]          */
/*  80 40 20 10 08 04 02 01 80 40 20 10 08 04 02 01 80 40 20 10 08 04 */
/*  |--offset--------->                                            */
/*                |--n_bits---------------------->                  */
/* Regresa la palabra de 2 bytes con el valor desempacado.          */
/*=================================================================*/
{
unsigned        rv ;            /* Return value.                   */
unsigned        fltr ;          /* Bits filtrados.                 */
int             beg,            /* Número de inicio de bits a filtrar. */
                end,            /* Número de fin de bits a filtrar. */
                n,              /* Número de fin de bits que faltan. */
                shift ;         /* Corrimiento de bits < <(+) > >(-). */
/* ------------------------------------------------------------------ */
rv = 0 ;                        /* Inicializa valor desemp.        */
beg = offset ;                  /* Inicia con "offset".            */
n = n_bits ;                    /* Faltan todos los bits.          */
/* ------------------------------------------------------------------ */
/* Loop para revisión, termina cuando no faltan bits.               */
/* ------------------------------------------------------------------ */
while ( n > 0 )
  {
  if ( (end = beg + n-1) > 7 ) end = 7 ;  /* Ultimo bit (en ese byte). */
  fltr = *buffer & beg_mask[beg] & end_mask[end] ;   /* Filtrado.    */
  n -= end - beg + 1 ;          /* Actualiza los que faltan.       */
  shift  = n - ( 7 - end ) ;    /* Corrimiento.                    */
  if ( shift > 0 )      rv |= fltr << shift ;   /* Recorre a la izq. */
  else if ( shift < 0 ) rv |= fltr >> -shift ;  /* Recorre a la derecha */
  else                  rv |= fltr ;            /* Añade así.      */
  if ( n > 0 )                  /* Si todavía faltan bits:         */
    { buffer++ ; beg = 0 ; }    /* - Recorre buffer e inicia.      */
  } /* while */
return rv ;
} /* unpack_bits() */

int ktab ( ubyte test_byte )
/*=================================================================*/
/* KTAB returns the number of bits in "test_byte".                  */
```

```
/* KTAB is the table of bit counts per byte. Using this should be      */
/* faster than calling gbits and summing each bit individually.        */
/*=====================================================================*/
{
static ubyte      kntab[256] = {
        0,1,1,2,1,2,2,3,1,2,2,3,2,3,3,4,1,2,2,3,2,3,3,
          4,2,3,3,4,3,4,4,5,1,2,2,3,2,3,3,4,2,3,3,4,3,
          4,4,5,2,3,3,4,3,4,4,5,3,4,4,5,4,5,5,6,1,2,2,
          3,2,3,3,4,2,3,3,4,3,4,4,5,2,3,3,4,3,4,4,5,3,
          4,4,5,4,5,5,6,2,3,3,4,3,4,4,5,3,4,4,5,4,5,5,
          6,3,4,4,5,4,5,5,6,4,5,5,6,5,6,6,7,1,2,2,3,2,
          3,3,4,2,3,3,4,3,4,4,5,2,3,3,4,3,4,4,5,3,4,4,
          5,4,5,5,6,2,3,3,4,3,4,4,5,3,4,4,5,4,5,5,6,3,
          4,4,5,4,5,5,6,4,5,5,6,5,6,6,7,2,3,3,4,3,4,4,
          5,3,4,4,5,4,5,5,6,3,4,4,5,4,5,5,6,4,5,5,6,5,
          6,6,7,3,4,4,5,4,5,5,6,4,5,5,6,5,6,6,7,4,5,5,
          6,5,6,6,7,5,6,6,7,6,7,7,8 } ;
return (int) (kntab[test_byte]) ;
} /* ktab */

/* ••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••  */
/* Implementación del módulo CHDATE.                                    */
/* Para cambio de Fecha-Hora.                                           */
/* Interfaz en CHDATE.H.                                                */
/* ••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••  */
#include "chdate.h"      /* Interfaz: date_hr.                          */
/*=====================================================================*/
/* Variables locales.                                                   */
/*=====================================================================*/
char     m_days[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 } ;
int      month_j_day[12] = {      /* days in year before this month.    */
                0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334 } ;

void chdate ( date_hr *source, date_hr *dest, int del_hr )
/*=====================================================================  */
/* Procedimiento para cambiar una fecha, incrementando a la fecha-hora  */
/* fuente (apuntada por "source") el número de horas "del_hr", y asig-  */
/* nando este valor a la fecha destino (apuntada por "dest").           */
/* Se supone que "source" contiene un valor válido.                     */
/*=====================================================================*/
{
int              t_year, t_month, t_day, t_hour ;
/* ------------------------------------------------------------------- */
/* Inicializa. Se emplean variables temporales de 16 bits "t_*", pues   */
/* durante las operaciones se pueden saturar los valores de los campos  */
/* (recuérdese que son unsigned de 4 o 5 bits).                         */
/* ------------------------------------------------------------------- */
if ( ( source->year & 3 ) == 0 )          /* Cada cuatro años:          */
  m_days[1] = 29 ;              /*  es bisiesto.                        */
t_year  = source->year ;
t_month = source->month ;
t_day   = source->day ;
t_hour  = source->hour ;
t_hour += del_hr ;
/* ------------------------------------------------------------------- */
/* Loop hasta encontrar la fecha hora válida.                           */
/* ------------------------------------------------------------------- */
while ( ( t_hour < 0 ) || ( t_hour >= 24 ) )
```

```
{
/* ------------------------------------------------------------ */
/* Checa hora negativa (regresa tiempo).                        */
/* ------------------------------------------------------------ */
if ( t_hour < 0 )
  { /* Hora negativa, pasa al día anterior. ------------------- */
  t_hour += 24 ;
  if ( --t_day == 0 )
    { /* Día negativo, pasa al mes anterior. ------------------ */
    if ( --t_month == 0 )
        { /* Mes negativo, pasa al año anterior. ------------- */
        t_month = 12 ;
        t_year-- ;
        m_days[1] = ( ( t_year & 3 ) == 0 ) ? 29 : 28 ;
        } /* if */
    t_day = m_days[t_month-1] ;
    } /* if */
  } /* if ( t_hour < 0 ) */
/* ------------------------------------------------------------ */
/* Checa hora mayor a las 11:00 pm (adelantar tiempo).          */
/* ------------------------------------------------------------ */
if ( t_hour > 23 )
  { /* Pasa al siguiente día. --------------------------------- */
  t_hour -= 24 ;
  if ( ++t_day > m_days[t_month-1] )
    { /* Pasa al siguiente mes. ------------------------------- */
    if ( ++t_month == 13 )
        { /* Pasa al siguiente año. ------------------------- */
        t_month = 1 ;
        t_year++ ;
        m_days[1] = ( ( t_year & 3 ) == 0 ) ? 29 : 28 ;
        } /* if */
    t_day = 1 ;
    } /* if */
  } /* if ( t_hour > 23 )*/
 /* ------------------------------------------------------------ */
 } /* while */
/* ------------------------------------------------------------ */
/* Finalmente copia valores a la estructura destino.            */
/* ------------------------------------------------------------ */
dest->year  = t_year ;
dest->month = t_month ;
dest->day   = t_day  ;
dest->hour  = t_hour ;
} /* chdate() */

int tojul ( date_hr *d )
/*==============================================================*/
/* Convierte la fecha a días transcurridos desde el 31 de Diciembre de  */
/* 1944 (1 de enero de 1945 es 1).                              */
/*==============================================================*/
/* Antes estaba en el módulo MISC, pero está más relacionado con fechas */
/* por lo que se cambió aquí. En el original tenía el siguiente coment. */
/*   This subroutine converts month, day, year to days after    */
/*   January 0, 1945                                            */
/*==============================================================*/
{  int  j_day ;
/* ------------------------------------ */
```

```
j_day = (d->year-45)*365 +          /* Días de los años transcurr.      */
        (d->year-45)/4 +            /* Días de los años bis. trans.     */
        month_j_day[d->month-1] +   /* Días de los meses transcurr.     */
        d->day ;                    /* Días del mes.                    */
if ( ( d->month > 2 ) && ( ( d->year & 3 ) == 0 ) )
  j_day++ ;                         /* Bisiesto.                        */
return j_day ;
} /* tojul */

void change_year ( date_hr *source, date_hr *dest, int del_yr )
/*=================================================================*/
/* Procedimiento para cambiar una fecha, incrementando a la fecha-hora      */
/* fuente (apuntada por "source") el número de años "del_yr", y asig-       */
/* nando este valor a la fecha destino (apuntada por "dest").               */
/* Se supone que "source" contiene un valor válido.                         */
/*=================================================================*/
{
dest->year  = source->year + del_yr ;
m_days[1] = ( ( dest->year & 3 ) == 0 ) ? 29 : 28 ;
dest->month = (source->month<1) ? 1 : (source->month>12) ? 12 : source->month ;
dest->day   = source->day ;
if ( dest->day > m_days[dest->month-1])     dest->day = m_days[dest->month-1] ;
else if ( dest->day < 1 )                   dest->day = 1 ;
dest->hour  = (source->hour<0) ? 0 : (source->hour>23) ? 23 : source->hour ;
/* ------------------------------------------------------------------ */
} /* change_year() */

void change_month ( date_hr *source, date_hr *dest, int del_mo )
/*=================================================================*/
/* Procedimiento para cambiar una fecha, incrementando a la fecha-hora      */
/* fuente (apuntada por "source") el número de meses "del_mo", y asig-      */
/* nando este valor a la fecha destino (apuntada por "dest").               */
/* Se supone que "source" contiene un valor válido.                         */
/*=================================================================*/
{ int   mo ;
/* ------------------------------------------------------------------ */
dest->year  = source->year ;
mo = source->month + del_mo ;
while ( 1 > mo )        { mo += 12 ; dest->year-- ; }
while ( mo > 12 )       { mo -= 12 ; dest->year++ ; }
dest->month = mo ;
dest->day   = source->day ;
m_days[1] = ( ( dest->year & 3 ) == 0 ) ? 29 : 28 ;
if ( dest->day > m_days[dest->month-1])     dest->day = m_days[dest->month-1] ;
else if ( dest->day < 1 )                   dest->day = 1 ;
dest->hour = (source->hour<0) ? 0 : (source->hour>23) ? 23 : source->hour ;
} /* change_month() */

int week_day ( date_hr *source )
/*=================================================================*/
/* Regresa el día de la semana correspondiente a la fecha-hora (apunta-     */
/* da por "source") DHW_SU=Domingo, DHW_MO=Lunes, .. DHW_SA=Sábado.         */
/* Se supone que "source" contiene un valor válido.                         */
/*=================================================================*/
{
/* El 31 de Diciembre de 1944 (la referencia) es Domingo.                   */
/* ------------------------------------------------------------------ */
return ( tojul ( source ) % 7 ) ;
```

```
} /* week_day() */

int month_days ( date_hr *source )
/*==================================================================*/
/* Regresa el número de días que tiene el mes correspondiente a la fe-   */
/* cha-hora (apuntada por "source").                                     */
/* Se supone que "source" contiene un valor válido.                      */
/*==================================================================*/
{
m_days[1] = ( ( source->year & 3 ) == 0 ) ? 29 : 28 ;
return m_days[source->month-1] ;
} /* week_day() */
```

Apéndice D

```
type
 Arreg30 =   Array [1..30] of Real;
 Arreg16 =   Array [1..16] of Real;
 Rfecha =  Record
           dia : 1..31;
           mes : 1..12;
           ano : integer;
           hora : integer;
          End;
 cadena2     = string[2];

var
i, j, k    : integer;
e1, e2, e3  : text;

cadena            :string[20];
cad19             :string[19];
cad7              :string[7];
y, x, y2, x2, DyA, DxA, DyD, DxD : Real;
prom              : Real;
fecha1, fecha2 : Rfecha;
Dia1, Dia2     : Arreg30;


   Function  Media (p1, p2: Arreg30; dim : Integer): Real;
   var
      medio : Real;
      i     : integer;

   Begin
      medio := 0;
      For i:= 1 to dim do
        medio := medio + p1[i] + p2[i];
      Media := Medio/ (2*dim);
   End;


 Procedure  LeeDatosDia ( var Dia: arreg30 );
  var
   i,k  : integer;
   cadena   : String[20];
 Begin
     readln(e1);
     For k := 1 to 6 do
       begin
         read (e1, cadena);
         for i := 1 to 5 do
          read (e1,dia[(k-1)*5  + i]);
         readln(e1);
       end;
 End;
```

```
Function conv (caract : cadena2 ) : longint;

 type
   digito = set of '0'..'9' ;
 var
   digitos : set of '0'..'9';
   valor : longint;
 begin
   valor := 0;
   if caract[1] in digitos then
     begin writeln ('entro');valor :=  ( (Ord(caract[1]) - 48)* 10 ) ;end;
   writeln (Ord(caract[1]):3, valor:3);
   valor := valor + (Ord (caract[2])-48);
   conv := valor;
 end;

(*  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  *)
Procedure Leefecha ( var fecha : Rfecha );
  { Lee la fecha y la convierte de caracteres a enteros. La regresa
    en un registro fecha  }

var  a7    : string[7] ;
    d2, m2, e2, h2   : cadena2;
    a2     : integer;
    a      : char;

   Begin
     readln (e1, a7, d2, e2, m2,e2, a2, a, h2);
     with fecha do
       begin
         dia := conv (d2);
         mes := conv (m2);
         ano := a2 ;
         hora := conv (h2);
         writeln (d2, m2, a2, h2,'  ', dia:3, mes:3,ano:3, hora:3);
       end;
   End;

(*  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  *)

Begin
  assign (e1,'invcaso2.dat');        reset (e1 );
  assign (e2,'\pas\tp6\caso2\fechas.dat');      rewrite (e2 );
  assign (e3,'\pas\tp6\caso2\diasdato.dat');        rewrite (e3 );
  Leefecha ( fecha1 );
  LeeDatosDia ( Dia1 );
  k := 0 ;
  While NOT EOF(e1) do
  Begin
    Leefecha ( fecha2 );
    LeeDatosDia ( Dia2 );
    if (fecha1.hora = 0) and (fecha2.hora = 0 ) then
      if (fecha2.dia =fecha1.dia +1) Or ((fecha2.dia = 1) and (fecha1.dia = 31)) then
```

```
      Begin
        k := k + 1 ;
        Prom := Media (Dia1, Dia2, 30 );
        For i := 1 to 30 do  write (e3, Dia1[i]-Prom:7:2);
        For i := 1 to 30 do  write (e3, Dia2[i]-Prom:7:2);
        writeln ( e3, prom:9:2 );
        with fecha1 do
          writeln (e2, dia:3, mes:3, ano:3, hora:3);
      End;
    fecha1 := fecha2 ;
    Dia1 := Dia2;
  End;
  close (e1);  close (e2);   close (e3);
End.
```

```
Program Diasleidos;

{programa que busca los datos de temperatura y vientos de los días
correspondientes, y el valor de la temperatura mínima del último día
y del día a pronosticar, y arma el vector de datos}

type
 Rfecha =  Record
           dia : 1..31;
           mes : 1..12;
           ano : Integer;
           hora : longint;
         End;

var
i, j, k    : integer;
f_datos, f_min1, f_min2 : Rfecha;
e1, e2, e3, fe, fs  : text;
tem_1, tem_2   : Real;
misma : boolean;
datobase :Array [1..121] of real;

Procedure Leefecha ( var fecha : Rfecha );
var  i : Integer;
   Begin
     with fecha do
       begin
       readln (e1, dia, mes, ano, hora );
       for i := 1 to 121 do read (fe, datobase [i] );
       end;
   End;

(*  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  +  *)

Procedure Leefechaymin ( var fecha : Rfecha; var tmin : real );
 var
  aux : real;
 Begin
   with fecha do
     readln (e2, ano, mes, dia, aux, tmin, aux, aux);
 End;
 (*  •  •  •     •     •     •     •     •)

Function mismafecha ( f1, f2: Rfecha): Boolean;

 begin
  if ((f1.dia=f2.dia) AND (f1.mes=f2.mes) AND (f1.ano=f2.ano)) Then
    mismafecha := TRUE
  else
    mismafecha :=FALSE
 end;
```

```
Begin
  assign (e1,'fechas2.dat');              reset (e1 );
  assign (e2,'78a91.dat');                reset (e2 );
  assign (e3,'tempmin.dat');                      rewrite (e3 );
  assign (fe,'vector61.dat');             reset (fe );
  assign (fs,'vector120.dat');            rewrite (fs);
  Leefecha ( f_datos );
  Leefechaymin ( f_min1,tem_1 );
  k := 0 ;
  While NOT EOF(e1) do
  Begin
    misma := mismafecha ( f_datos, f_min1);
    if misma then
    begin
      Leefechaymin (f_min2, tem_2 );
      write ( f_datos.dia:3,f_datos.mes:3,f_datos.ano:3,'   ');
      writeln ( f_min1.dia:3, f_min1.mes:3 ,f_min1.ano:3, tem_1:6:2);
      writeln (e3, tem_1:7:2, tem_2:7:2, f_min1.dia:3, f_min1.mes:3, f_min1.ano:3 );
      for i := 1 to 121 do  write (fs, datobase[i]:7:2 ); writeln(fs);
      k:= k+1;  writeln (k);
      Leefecha ( f_datos );
      f_min1 := f_min2;  tem_1 := tem_2;
    end
    else
    begin
     writeln ('no hay dia ',  f_min1.dia:3, f_min1.mes:3, f_min1.ano:3);
      if (f_datos.ano < f_min1.ano) then
         Leefecha (f_datos )
       else
        if (f_datos.ano > f_min1.ano) then
          Leefechaymin ( f_min1,tem_1 )
          else  if (f_datos.mes < f_min1.mes) then
                Leefecha (f_datos )
               else if (f_datos.mes > f_min1.mes) then
                Leefechaymin ( f_min1,tem_1 )
                   else if (f_datos.dia<f_min1.dia) then
                        Leefecha (f_datos )
                       else
                        Leefechaymin ( f_min1,tem_1 );
    end;
  End;
  close (e1);   close (e2);   close (e3);   close (fe); close (fs);
End.
```

Apendice E

```
C*****************************************************************
C* PROGRAMA para calcular los vectores o FUNCIONES ORTOGONALES ( EOF) *
C* EMPIRICAS de un campo de datos, vector f(i,n) del que se tienen nmu *
C* muestras.....                                                 *
C*****************************************************************
C* Se utilizó para predecir las temperaturas minimas del DF como      *
C* función de la temperatura sobre 14 puntos del territorio, los dos  *
C* dias anteriores.                                                   *
C* Cada vector muestra contiene: la temperatura del dia 1 y dia 2     *
C*                    en 14 puntos de una malla                       *
C*                                                                    *
C*****************************************************************
C    Nel  número de elementos del vector muestra  f
C    Nmu  número de muestras
C    Ndim tamaño del vector que contiene elementos de la matriz simétrica

     PARAMETER( Ndim=  5000, Nmu =400, Nvar =  28 )
     REAL  E , AA, Lamda, X
     DOUBLE PRECISION   A, R
     DIMENSION E (Nmu,nvar), X(Nvar),  R(Nvar,Nvar), AA(nvar,nvar)
     DIMENSION A (Ndim)

     open(10,file = 'DatosTem.dat')
     open(15, file = 'matcov.var')
     open(30,file = 'salidaok.dat')
     write(*,'(A \)') ' Numero de elementos del vector ? '
     read (*,*)  Nel
     write(*,'(A \)') ' Numero de casos ? '
     read (*,*)  Ncasos
     DO 2 K1 = 1,  Ndim
   2   A(K1) = 0.
     DO 3 K1 = 1,Nvar
      DO 3 K2 = 1,Nvar
   3    R(K1,K2) = 0.
     NN = 0
C                    Lee archivo de datos, muestra por vez
     DO 19 i = 1,  Ncasos
       write(*,*)  'lee caso ..',i
        READ(10,123) (E(i, jj), jj = 1,  28)
   19  CONTINUE
c .....     normalizar los vectores columna
c      do i = 1, nel
c       call  normaliza ( E(1,i), Ncasos)
c      end do

     L  = 0
     DO 60 id = 1,Nel
       DO 71  j = 1,  id
         L  = L + 1
         DO 70 k = 1,  Ncasos
```

```
 70        A(L)=A(L)+E(k,id)*E(k,j)
           A(L) = A(L) / Ncasos
           AA(id,j) = A(l)
           AA(j,id) = A(l)
 71    CONTINUE
 60  CONTINUE
     WRITE(6,*) ' dimension de A  ',L
c ..... imprime la matriz de covarianza
     idiag = 0
     do i=1, Nel
       write(15,*) '...renglon..', i
       write(*,*) i
       write(15,'(10f8.2)') ( A(idiag+j), j=1,i)
       idiag = idiag + i
     end do
c  .....  calcula valores y vectores
     CALL EIGEN(A, R, Nel,0)

     write(30,*) '     VALORES EN LA MATRIZ A.....'
     idiag = 0
     do i=1, Nel
      idiag = idiag + i
      write (30,'(f12.3)')  A(idiag)
     end do
     write (30,*) '     VECTORES  .....'
     do i=1, Nel
       write (30,*) '  vector  ' , i
       do j=0, Nel,10
         jr = Nel - j
         if (jr .GE. 10) jr = 10
         write (30,125) (R(jj,i), jj=j+1, j+jr)
       end do
     end do
c ...  verifica ortogonalidad
     do k = 1, nvar
      do j = 1, nvar
       PP = 0
       do i = 1, Nel
         PP = PP + R(i,k)*R(i,j)
       end do
       write (*,*) 'columna',j, 'y ', k, '->', PP
      end do
     end do
     idiag = 0
     do  k = 1, nel
       idiag = idiag + k
       lamda = A (idiag)
       do i = 1, nel
         X(i) = R (i,k)
       end do
       CALL Comprueba (AA,lamda, X, nel)
     end do
     write (*,*) ' ya termino de imprimir '
     CALL coeficientes ( R, nel )
```

```
123  format(28f6.2)
124  format(16F8.2)
125  format(10f7.4)
     STOP
     END

c    ***************************************************************
     subroutine Comprueba ( A, lamda, X, n )
c    **********
c    esta rutina verifica  que  A x = L x
c    ***********
     real A, lamda, X, Aux
     dimension  A (n,n), X(n)

     do i = 1, n
       aux = 0
       do 100 j = 1, n
100      aux = aux + A(i,j)*X(j)
c      write (30,*) aux, lamda*X(i)
     end do
     return
     end


c    ***************************************************************
     subroutine coeficientes ( Vect,  n )
c    ************************
c    rutina que obtiene los coeficientes de la combinacion lineal
c    de los v.p. para formar los dias a verificar
c    ***************************************************************
     PARAMETER  (nucasos = 260  )
     double precision Vect
     real     datodia, alfa, modvector, mod1, mod2, mod5, mod10, mod11
     real     c5, c2, c10, c11, s11
     real     e1, e2, e5, e10, m1, m2, m5, m10,m11, s1,s2,s5,s10, tmin
     dimension  Vect (n,n), datodia (28), alfa (28)
     integer    i, j, k, jj, jr
     DATA  m1, m2, m5, m10, s1, s2, s5, s10 /8*0/

     open (20, file='datospru.man')
     open (21, file='dataymin.dat')
     open (22, file='coefymin.dat')
     do k = 1, nucasos
       read (20, *)  (datodia(j), j= 1, n ), tmin
       write(21,205) (datodia(j), j= 1, n ), tmin
       write (30,*) 'datos caso..', k
       do i = 1, n
         alfa (i) = 0
         do 10 j = 1, n
10         alfa(i) = alfa(i) + Vect(j,i)*datodia(j)
       end do
       do j=0, N,10
         jr = N - j
         if (jr .GE. 10) jr = 10
         write (30,225) (alfa(jj), jj=j+1, j+jr)
```

```
      end do
      write (30,*) tmin
      write (22,210) (alfa(j), j=1,10), tmin
c...    verifica el error con 1, 5 y 10 vect propios
      modvector = 0
      mod1 = 0
      mod2 = 0
      mod5 = 0
      mod10 = 0
      mod11 = 0
      do 20 i = 1, 28
20      modvector = modvector + datodia(i)**2
      do i = 1, 28
        mod1 = mod1 + (alfa(1)*vect(i,1))**2
        mod2 = mod2 + (alfa(1)*vect(i,1) + alfa(2)*vect(i,2))**2
        c5 = 0
        c10 = 0
        do 30 j = 1, 5
          c5 = c5 + alfa(j)*vect(i,j)
30        c10 = c10 + alfa(j+5)*vect(i,j+5)
        c11 = c11 + alfa(11)*vect(i,11)
        mod5 = mod5 + c5 **2
        mod10 = mod10 + (c5+ c10) **2
        mod11 = mod11 + (c5+c10+c11)**2
      end do
      e1 = (modvector - mod1)/modvector *100
      e2 = (modvector - mod2)/modvector *100
      e5 = (modvector - mod5) / modvector*100
      e10 = (modvector - mod10)/modvector*100
      e11 = (modvector - mod11)/modvector*100
      write (*,*) 'llega a impresion ', k
      write(30,'(A15,f10.2)') 'modulo vector. ', modvector
      write(30,230) 'comp principal.',mod1,' error ', e1
      write(30,230) '2 componentes  ',mod2,' error ', e2
      write(30,230) '5 componentes  ',mod5,' error ', e5
      write(30,230) '10 componentes ',mod10,' error ',e10
      write(30,230) '11 componentes ',mod11,' error ',e11
      m1 = m1 + e1
      m2 = m2 + e2
      m5 = m5 + e5
      m10 = m10 +e10
      m11 = m11 +e11
      s1 = s1 + e1**2
      s2 = s2 + e2**2
      s5 = s5 + e5**2
      s10 = s10 + e10**2
      s11 = s11 + e11**2
    end do
    s1 = (s1 - (m1**2)/nucasos)/nucasos
    s2 = (s2 - (m2**2)/nucasos)/nucasos
    s5 = (s5 - (m5**2)/nucasos)/nucasos
    s10 = (s10 - (m10**2)/nucasos)/nucasos
    s11 = (s11 - (m11**2)/nucasos)/nucasos
    write (30,*) '                '
    write (30,*) '...parametros estadisticos de los errores ..'
```

```
      write (30,*)  '                media    varianza  desviacion '
      write (30,235) 'primer compon', m1/nucasos,  s1, SQRT(s1)
      write (30,235) '2 componentes', m2/nucasos,  s2, SQRT(s2)
      write (30,235) '5 componentes',m5/nucasos,  s5, SQRT(s5)
      write (30,235) '10 componentes',m10/nucasos, s10, SQRT(s10)
      write (30,235) '11 componentes',m11/nucasos, s11, SQRT(s11)
200   format(28f6.2)
225   format(10f7.2)
230   format (A15,f10.2,A8,f6.2,'%')
235   format (A15,f10.2,f10.2,f10.2)
205   format (28f7.2, f9.2)
210   format (10f7.2, f9.2)
      return
      end

c     ****************************************************************
      subroutine  normaliza ( V, ndim)
c     ************************
c  esta rutina normaliza los valores de un vector, restandole la media
c  y luego dividiendo entre la desviacion estandar
c     ****************************************************************
      real  v, media, sumaq, sigma
      dimension  v(200)

      media = 0
      do i= 1,ndim
        media = media + v(i)
      end do
      media = media / ndim
      sumaq = 0
      do i = 1, ndim
        sumaq = sumaq + v(i)*v(i)
      end do
      sigma = SQRT((sumaq - ndim * media*media) / ndim)
      do i= 1, ndim
        v(i) = (v(i) - media)/ sigma
      end do
      return
      end


C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE EIGEN(A,R,N,MV)
C     ..................................................................
C
C        PURPOSE
C           COMPUTE EIGENVALUES AND EIGENVECTORS OF A REAL SYMMETRIC MATRIX
C
C        USAGE
C           CALL EIGEN(A,R,N,MV)
C
C        DESCRIPTION OF PARAMETERS
C           A - ORIGINAL MATRIX (SYMMETRIC), DESTROYED IN COMPUTATION.
C               RESULTANT EIGENVALUES ARE DEVELOPED IN DIAGONAL OF
```

```
C          MATRIX A IN DESCENDING ORDER.
C        R - RESULTANT MATRIX OF EIGENVECTORS (STORED COLUMNWISE,
C          IN SAME SEQUENCE AS EIGENVALUES)
C        N - ORDER OF MATRICES A AND R
C        MV- INPUT CODE
C              0  COMPUTE EIGENVALUES AND EIGENVECTORS
C              1  COMPUTE EIGENVALUES ONLY (R NEED NOT BE
C                 DIMENSIONED BUT MUST STILL APPEAR IN CALLING SEQUENCE)
C
C     REMARKS
C        ORIGINAL MATRIX A MUST BE REAL SYMMETRIC (STORAGE MODE=1)
C        MATRIX A CANNOT BE IN THE SAME LOCATION AS MATRIX R
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        DIAGONALIZATION METHOD ORIGINATED BY JACOBI AND ADAPTED
C        BY VON NEUMANN FOR LARGE COMPUTERS AS FOUND IN 'MATHEMATICA
C        METHODS FOR DIGITAL COMPUTERS', EDITED BY A. RALSTON AND
C        H.S. WILF, JOHN WILEY AND SONS, NEW YORK, 1962, CHAPTER 7
C
C     .....................................................................
      DIMENSION A(1),R(1)
C     .....................................................................
C
C     IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C     C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C     STATEMENT WHICH FOLLOWS.
C
      DOUBLE PRECISION A,R,ANORM,ANRMX,THR,X,Y,SINX,SINX2,COSX,
     1                 COSX2,SINCS,RANGE
C
C     THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C     APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C     ROUTINE.
C
C     THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C     CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENT
C     40, 68, 75, AND 78 MUST BE CHANGED TO DSQRT.  ABS IN STATEMENT
C     62 MUST BE CHANGED TO DABS. THE CONSTANT IN STATEMENT 5 SHOULD
C     BE CHANGED TO 1.0D-12.
C
C     .....................................................................
C
C     GENERATE IDENTITY MATRIX
C
      write (*,*) ' Genera matriz identidad...'
    5 RANGE=1.0D-18
      IF(MV-1) 10,25,10
   10 IQ=-N
      DO 20 J=1,N                    !
      IQ=IQ+N
      DO 20 I=1,N
      IJ=IQ+I
```

```
      R(IJ)=0.0
      IF(I-J) 20,15,20
15    R(IJ)=1.0
20 CONTINUE
C
C     COMPUTE INITIAL AND FINAL NORMS (ANORM AND ANORMX)
C
   write (*,*) ' Calcula normas...'
25 ANORM=0.0
   DO 35 I=1,N
     DO 35 J=I,N
     IF(I-J) 30,35,30
30       IA=I+(J*J-J)/2
         ANORM=ANORM+A(IA)*A(IA)
35 CONTINUE
   IF(ANORM) 165,165,40
40  ANORM=1.414* DSQRT(ANORM)
    ANRMX=ANORM*RANGE/FLOAT(N)
   write (*,*) '... Normas = > ', anorm, anrmx
C
C     INITIALIZE INDICATORS AND COMPUTE THRESHOLD, THR
C
   iii =0
   IND=0
   THR=ANORM
45 THR=THR/FLOAT(N)                    !
50 L=1
55 M=L+1

C
C     COMPUTE SIN AND COS
C
60 MQ=(M*M-M)/2
   LQ=(L*L-L)/2
   LM=L+MQ
62 IF( ABS(A(LM))-THR) 130,65,65
65  IND=1
    LL=L+LQ
    MM=M+MQ
    X=0.5*(A(LL)-A(MM))
68  Y=-A(LM)/ DSQRT(A(LM)*A(LM)+X*X)
    IF(X) 70,75,75
70     Y=-Y
75     SINX=Y/ DSQRT(2.0*(1.0+( DSQRT(1.0-Y*Y))))
       SINX2=SINX*SINX
78     COSX= DSQRT(1.0-SINX2)
       COSX2=COSX*COSX
       SINCS =SINX*COSX
C
C     ROTATE L AND M COLUMNS
C
       iii = iii + 1
       write (*,*) ' Rotacion  ',iii
       ILQ=N*(L-1)
       IMQ=N*(M-1)
```

```
      DO 125 I=1,N
        IQ=(I*I-I)/2
        IF(I-L) 80,115,80
 80     IF(I-M) 85,115,90
 85     IM=I+MQ
        GO TO 95
 90     IM=M+IQ
 95     IF(I-L) 100,105,105
 100    IL=I+LQ
        GO TO 110
 105    IL=L+IQ
 110    X=A(IL)*COSX-A(IM)*SINX
        A(IM)=A(IL)*SINX+A(IM)*COSX
        A(IL)=X
 115    IF(MV-1) 120,125,120
 120    ILR=ILQ+I
        IMR=IMQ+I
        X=R(ILR)*COSX-R(IMR)*SINX
        R(IMR)=R(ILR)*SINX+R(IMR)*COSX
        R(ILR)=X
125     CONTINUE
   X=2.0*A(LM)*SINCS
   Y=A(LL)*COSX2+A(MM)*SINX2-X
   X=A(LL)*SINX2+A(MM)*COSX2+X
   A(LM)=(A(LL)-A(MM))*SINCS+A(LM)*(COSX2-SINX2)
   A(LL)=Y
   A(MM)=X
C
C      TESTS FOR COMPLETION
C      TEST FOR M = LAST COLUMN
C
 130 IF(M-N) 135,140,135
 135 M=M+1
     GO TO 60
C
C      TEST FOR L = SECOND FROM LAST COLUMN
C
 140 IF(L-(N-1)) 145,150,145
 145 L=L+1
     GO TO 55
 150 IF(IND-1) 160,155,160
 155 IND=0
     GO TO 50
C
C      COMPARE THRESHOLD WITH FINAL NORM
C
 160 IF(THR-ANRMX) 165,165,45
C
C      SORT EIGENVALUES AND EIGENVECTORS
C
 165 IQ=-N
     DO 185 I=1,N
     IQ=IQ+N
     LL=I+(I*I-I)/2
     JQ=N*(I-2)
```

```
      DO 185 J=I,N
      JQ=JQ+N
      MM=J+(J*J-J)/2
      IF(A(LL)-A(MM)) 170,185,185
  170 X=A(LL)
      A(LL)=A(MM)
      A(MM)=X
      IF(MV-1) 175,185,175
  175 DO 180 K=1,N
      ILR=IQ+K
      IMR=JQ+K
      X=R(ILR)
      R(ILR)=R(IMR)
  180 R(IMR)=X
  185 CONTINUE
c ...  verifica ortogonalidad
      do k = 1, 10
        PP = 0
        do i = 1, N
          PP = PP + R(i)*R(i+N*(k-1))
        end do
        write (*,*) ' eigen ..columna 1 y ', k, '-> ', PP
      end do
      RETURN
      END
C*****************************************************************
```

Apéndice F

```
program valida;
  type
     registro =record
        n      : integer;
        S_error: real;
        error_cua :real;
        mini, maxi : real;
        errorabs : real;
        ncasos : integer;
      end;

  var  i,j, k,n : integer;
       t1, t2 :  real;
       f, f2, g1 : text;
       Coe : Array [ 1..120] of real;
       Clust : Array [1..10] of registro;
       fun1, fun2 :  real;
       aux :real;

  Procedure  Evalua_errores (f:real; k, i: integer );
   var   delta : real;
   begin
     delta := f - t2;
     with clust[k] do
       begin
         n := n+1 ;
         S_error := S_error + delta ;
         error_cua := error_cua + Sqr ( delta);
         Errorabs := errorabs + abs (delta);
         if mini > abs( delta) then mini := abs(delta);
         if maxi < abs( delta) then maxi := abs(delta);
         if delta <= 1 then ncasos := ncasos +1;
       end;
   end;


   Begin
     assign (F, 'vector12.dat'); reset (f);
     assign (f2, 'tempmin.dat');reset (f2);
     assign (g1, 'salida.500'); rewrite (g1);

     For j := 1 to 10 do
       with clust[j] do
          begin
           n := 0;
           S_error := 0;
           Error_cua := 0;
```

```
        errorabs := 0;
        mini := 100;
        maxi := 0;   ncasos := 0;
      end;
  For i := 1 to 516 do
   begin
     For j := 1 to 120 do
      read(f, Coe[j]);
     readln(f, aux );
     readln(f2, t1, t2);
   end;


  For i := 517 to 611 do
   begin
     For j := 1 to 120 do
      read(f, Coe[j]);
     readln(f, aux );
     readln(f2, t1, t2);

     fun1 :=  2.203 +0.486*T1 + 0.574*Coe[11] -0.119*Coe[56] - 0.135*Coe[33] - 0.086*Coe[15]
              + 0.076*Coe[4] - 0.309*Coe[16] + 0.126*Coe[58] -0.214*Coe[6];
     Evalua_Errores (fun1, 1, i);
     fun2 :=  2.02 +0.50*T1 + 1.2*Coe[83] - 0.087*Coe[10] + 0.07*Coe[87] - 2.34*Coe[102]
            - 1.73*Coe[72] + 1.48*Coe[113] - 0.1259*Coe[56] + 0.079*Coe[81] - 0.04*Coe[50];
     Evalua_Errores (fun2, 2, i);
     writeln (g1, fun1:9:2, fun2:9:2, T2:9:2, fun1-t2:10:2 );
   end;
  n := 95 ;
  writeln (g1);
  writeln( g1, '#grupo n   suma_e  error_cua min  max');
  For k:= 1 to 2 do
    with clust[k] do
      writeln( g1, K:4, n:4, S_error/n:8:3, Sqrt(error_cua/n):9:3,
               mini:7:3,maxi:7:3,  errorabs/n:9:3, ncasos:4);
  close (f); close(g1); readln;
End.
```

Bibliografia

[Coole71] Cooley W. y Lohnes P., 1971: Multivariate Data Analysis, editorial John Wiley Sons

[Drapp66] Drapper N. & Smith H., Applied Regression Analysis, editorial Wiley, 1966

[Glahn72] Glahn H. y Lowry Dale, 1972: The Use of Model Output Statistics (MOS) in Objetive Weather Forecasting. *Journal of Applied Meteorology*, Vol 11. 1203-1211

[Green76] Green Paul, Mathematical Tools for Applied Multivariate Analysis, editorial Academic Press, 1976

[Jolli86] Jollife I.T., Principal Components Analysis, Springer Verlang, 1986

[Kalna91] Kalnay E., Petersen R., Kanamitsu M. y Baker E., 1991: U.S. Operational Numerical Weather Prediction. Reviews of Geophisics, Suplement, U.S. National Report to International Union of Geodesy and Geophisics 1987-1990. 104-114

[Kalna90] Kalnay E., Kanamitsu M. y Baker E., 1990: Global Numerical Weather Prediction at tht National Meteorological Center. *Bulletin of teh Meteorological Society*, Vol 71, nº 10. 1410-1427

[Kanam89] Kanamitsu Masao, 1989 : Description of the NMC Global Data Asimilation and Forecast System. *Weather and Forecasting*, vol 4, nº 3, 335-342

[Klein59] Klein, Lewis & Enger, 1959, Objective prediction of five-day mean temperature during winter, Journal of Metereology, Vol 16

[Klein70] Klein & Lewis, 1970, Computer forecasts of maximun and minimun temperature, Journal of Applied Metereology

[Kruiz80] Kruizinga S. 1980? Statistical Interpretation of ECMWF Products in Dutch Weather Service - Royal Netherlands Meteorological Institute De Bilt, The Netherlands

[Kung80] Kung E. & Sharif T., Multi-regression Forecasting of the Indian summer monsoon with antecedent patterns oOf large scale circulation, WMO Symposium on Probabilistic and Stadistical Met in Weather Forecasing, 1980

[Mozer93] Mozer Joel & Zehnder Joseph, Cluster Analysis of Eastern North Pacific Tropical Cyclogenesis Precursors, enviado a Journal of Geophysical Research, 1993

[Neter79] Neter John & Wasserman William, Applied Statistics, editorial Allyn and Bacon, Inc, 1979

[Overl82] Overland J. & Preisendorfer R., A significance test for principal components applied to a cyclone climatology, Monthy Weather Review, 1982

[Reyme93] Reyment Richard & Joreskog K., Applied Factor Analysis in the Natural Sciences, editorial Cambridge University Press, 1993

[Ralst62] Ralston & Wilf, Mathematical Methods for Digital Computers, Wiley, 1962

[Rosco87] Roscoe John, Fundamental research Statistics, editorial Holt, Rinehart and Winston, Inc., 1987

[Rouss82] Rousseau D. 1982? Work on Statistical Adaptation for Local Forecasts in France - Direction de la Météorologie, EERM/GMD, Paris Francia. 395-415

[Wilki78] Wilkinson J. H., The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1978

Compact Disc of National Meteorological Center Grid Point Data Set, versión II, General Information and User's Guide, Universidad de Washington, Data Support Section, National Center for Atmospheric Research, 1990