

**UNIVERSIDAD NACIONAL AUTÓNOMA DE  
MÉXICO**

**FACULTAD DE INGENIERÍA**

**RÉGIMEN CRÍTICO EN CANALES DE SECCIÓN  
COMPUESTA**

Tesis que para obtener el título de Ingeniero Civil presenta:

Francisco Javier Lovera Salazar

Director de Tesis: Dr. Gilberto Sotelo Ávila

Ciudad de México, 2003



Universidad Nacional  
Autónoma de México

Dirección General de Bibliotecas de la UNAM

**Biblioteca Central**



**UNAM – Dirección General de Bibliotecas**  
**Tesis Digitales**  
**Restricciones de uso**

**DERECHOS RESERVADOS ©**  
**PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Esta tesis la dedico en especial a la memoria de mi madre *t*.

También a Amy por todo su amor y apoyo.

A mi padre y mis hermanos Héctor y Karla, mi familia y mi corazón.

A Edward y Kathy, mis incansables motivadores.

Agradezco infinitamente a:

Dr. Gilberto Sotelo por su paciencia y comprensión.

Mis profesores en la Facultad de Ingeniería por sus conocimientos y experiencias.

# ÍNDICE

INTRODUCCIÓN	1
I ANTECEDENTES	2
I.1 El agua en la vida humana	2
I.2 Los canales como infraestructura	2
I.3 Canales en México	2
I.4 Importancia del diseño de canales	4
II CANALES DE SECCIONES COMPUESTAS	6
II.1 Geometría	6
II.2 Casos prácticos	8
II.3 Flujo uniforme	9
II.4 Régimen crítico	11
III RÉGIMEN CRÍTICO EN SECCIONES COMPUESTAS	12
III.1 Criterio de energía específica mínima	12
III.2 Criterio de momentum mínimo	13
III.3 Método de Blalock y Sturm	14
III.3.1 Ecuaciones básicas	14
III.3.2 Variación de $n$ con el tirante	16
III.3.3 Tirantes críticos múltiples	18
III.3.4 Algoritmo	21
III.4 Método de Chaudrhy y Bhallamudi	23
III.3.1 Ecuaciones básicas	23
III.3.2 Tirantes críticos múltiples	27
III.3.3 Algoritmo	28
III.5 Comparación de ambos métodos	29
III.5.1 Comparación analítica	29
III.3.2 Comparación de resultados	30
IV PROGRAMA DE CÓMPUTO EN VISUAL BASIC	35
IV.1 Consideraciones	35
IV.2 Cálculo de las características geométricas	36
IV.3 Representación gráfica de puntos	39
V APLICACIONES NUMÉRICAS	41
V.1 Dos casos reales	41
V.1.1 Río Tijuana	41
V.1.2 Arroyo sin nombre en Cabo San Lucas	41
V.2 Revisión del diseño	42
V.3 Proceso constructivo	43
VI CONCLUSIONES	45

ANEXOS	
ANEXO I MANUAL DEL SISTEMA	46
ANEXO II PANTALLAS DEL PROGRAMA	64
ANEXO III CÓDIGO DE PROGRAMACIÓN	70
BIBLIOGRAFÍA	136

# INTRODUCCIÓN

El régimen crítico en canales es una condición muy importante para los problemas de diseño. Se utiliza como sección de control, ya que indica el punto de cambio entre los regímenes subcrítico y supercrítico. En la experimentación se ha observado que se presenta con características de flujo de mucha inestabilidad, debido a lo cual en el diseño de un canal se busca que, para su funcionamiento en condiciones normales, se encuentre lo más alejado de este punto

El régimen crítico para canales de sección sencilla se estudia a profundidad en cualquier curso de Hidráulica de Canales. Sin embargo, para los canales de sección compuesta, generalmente, sólo se presenta el estudio del flujo uniforme, dejando de lado la condición de flujo crítico. La definición del número de Froude para los canales de sección compuesta difiere de la empleada en los canales de sección sencilla, ya que se debe considerar la interacción entre el cauce principal y las llanuras o bermas de inundación laterales que se genera por las fuerzas tangenciales de transferencia de momentum entre las subsecciones. Así es que se puede considerar la ecuación de la energía o la de cantidad de movimiento, junto con la de continuidad; aparentemente cada definición presenta resultados diferentes. Este tema fue retomado recientemente por Aldama y Ocón (2002).

En el presente trabajo se realiza una comparación de los métodos de obtención de las condiciones críticas propuestos por Chaudhry (1988) y por Blalock (1981). Se propone un programa de cómputo que incluye ambos criterios, además de los resultados obtenidos comparados para tres canales diferentes. En el capítulo cinco se analizan dos casos reales en su diseño y se incluye una breve descripción del proceso constructivo de este tipo de canales.

## Referencias.

Aldama, Álvaro A. y Ocón Alfredo R., “¿Qué es el flujo crítico?”, XVII Congreso Nacional de Hidráulica, Monterrey, N.L., 2002.

Blalock, M. E. Y Sturm T. W., 1981, “Minimum specific energy in compound channel”, ASCE J Hydraulics Division, 107 (HY6):699-717

Chaudhry M. Hanif y Bhallamudi S. Murty, 1988, “Computation of critical depth in symmetrical compound channels” J. Hydraulic Research, IARH 26(4).

# **I ANTECEDENTES**

## **I.1 El agua en la vida humana**

El agua es la sustancia química más conocida y más abundante en la superficie terrestre, ya que cubre cerca de las tres cuartas partes de la misma. La podemos encontrar filtrándose a través de grietas y fracturas, corriendo en cauces naturales sobre la superficie o debajo de ella, en grandes cantidades en las regiones polares en forma de hielo y en la atmósfera en fase gaseosa. La vida en la Tierra, desde su origen, depende del agua. Es el principal ingrediente de las células vivas, compone el 90 por ciento del cuerpo humano. Prácticamente en cualquier lugar de la Tierra la humedad ayuda a la expansión de la vida.

Asimismo, la historia de la humanidad también se encuentra ligada a la disponibilidad de agua dulce que permita el desarrollo de civilizaciones. Los grupos humanos han buscado ubicarse cerca de lagos, lagunas, ríos, depósitos subterráneos para proveerse del vital líquido. Es necesaria para consumo propio y de sus animales, para sus sembradíos, para limpieza, para procesos artesanales, tecnológicos e industriales, etc. Desde siempre, el hombre se ha enfrentado a inundaciones, sequías, lluvias, etc. Ha sido necesario que, primero intuitivamente y después con teorías científicas, busque entender su comportamiento, sea para dirigir el agua hacia donde desea, sea para proteger las estructuras y ciudades que construye de los embates de la misma.

## **I.2 Los canales como infraestructura**

La disponibilidad, grado de control y ubicación del agua son en gran parte una medida del desarrollo económico y nivel de vida de una región. La infraestructura de un país se compone de todas aquellas obras de ingeniería que apoyan al desarrollo económico, industrial y humano. Entre ellas podemos considerar carreteras, puentes, vías de ferrocarril, puertos, aeropuertos, presas, plantas de generación de energía eléctrica, sistemas de riego, plantas de tratamiento de agua potable, sistemas de distribución de agua. Actualmente, las sociedades humanas necesitan de este tipo de obras para disponer, utilizar y desechar el agua de la manera más eficiente.

Los canales forman parte de estas estructuras y se diseñan para conducir el agua en las condiciones deseadas. Pueden ser parte de obras de protección de avenidas al corregir el cauce de un río, de un puerto al diseñarse como canales de navegación, de sistemas de riego, de sistemas de alcantarillado, de obras de urbanización al entubar un río.

## **I.3 Canales en México**

México es una República Federal formada por 31 Entidades Federativas y un Distrito Federal, 2 430 municipios y 16 delegaciones políticas. Existen 199 369 localidades en el país de las cuales 178 cuentan con 50 mil o más habitantes, 2 863 entre 2 500 y 49 999 hab. y 196 328 con menos de 2 500 hab. La población estimada en el 2001 fue de 100.1 millones de habitantes, el 75 por ciento vive en ciudades. La superficie es de casi 2 millones de km<sup>2</sup>, cuenta con 3 150 km de fronteras y con 11 122 km de litoral en ambos océanos.

La precipitación media histórica (1941-2001) en el país es de 772 mm, lo que equivale a 1 528 km<sup>3</sup>. Se estima que la evapotranspiración media es de 1 109 km<sup>3</sup> y que la recarga media de acuíferos de 75 km<sup>3</sup>, por lo que el escurrimiento virgen medio es de 394 km<sup>3</sup>. La precipitación ocurre, en la mayor parte del país, durante el verano, mientras que el resto del año es más bien escasa como se puede apreciar en la gráfica de la figura 1.1.

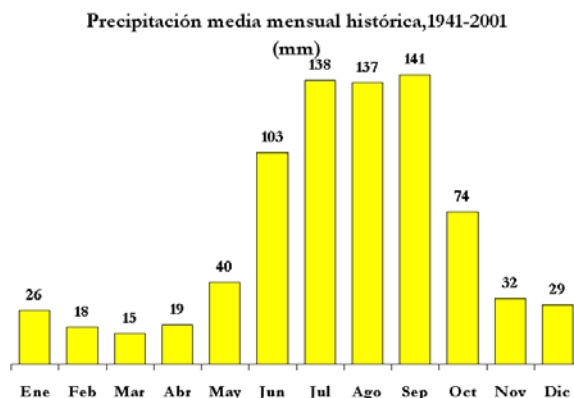


Fig. 1.1. Precipitación media histórica en México.  
(Datos de 1941 a 2001)

En México existe una gran cantidad de ríos, los cuales, de acuerdo con el lugar donde desembocan, se pueden clasificar en: la Vertiente Interior, la Vertiente del Golfo de México y la Vertiente del Pacífico. En la figura 1.2 y tabla 1.1 se muestran los principales. Se puede ver fácilmente que el sur del país cuenta con los ríos de mayor caudal, y hacia el norte los ríos son menores.



Fig. 1.2. Principales ríos en México



Tabla 1.1. Principales ríos en México

Vertiente	Río	Gasto medio anual (m <sup>3</sup> /s)	Longitud del río (km)
Interior	Nazas	6.6	255
Golfo de México	Grijalva-Usumacinta	2154.0	608*
	Papaloapan	583.6	354
	Pánuco	449.0	510
	Coatzacoalcos	440.8	325
	Bravo	37.4	2008*
Pacífico	Balsas	428.5	770
	Lerma-Santiago	226.8	1270
	Ometepec	185.3	115
	Colorado	7.2	30*

Un fenómeno común en nuestro país son los huracanes, debido a su ubicación geográfica, los cuales tienden a incrementar en cantidades importantes la precipitación y el escurrimiento en una zona específica. El caso más grave que se recuerda es el huracán Gilberto que atravesó la península de Yucatán, el Golfo de México y más adelante penetró en el continente hacia el estado de Nuevo León causando daños considerables en el año de 1988. Sin embargo, ambos litorales están expuestos a estos fenómenos sin distinción.

Como ejemplos de canalizaciones en nuestro país podemos mencionar los ríos Tijuana, ubicado en la ciudad homónima; Churubusco, Magdalena y el Rosario en la ciudad de México; Santa Catarina en Monterrey; Tuxpan en el puerto del mismo nombre.

#### I.4 Importancia del diseño de canales

En el diseño de canales se debe de considerar una sección hidráulica adecuada, las propiedades del material donde se construirá y factores económicos.

En el caso de la selección de la sección hidráulica se trabaja bajo las condiciones de flujo uniforme y de gasto máximo y se debe de considerar el trazo del canal; la condición crítica resulta en condiciones de alta inestabilidad, por lo que en el diseño se trata de alejar las condiciones de trabajo lo más posible de este estado. Las características del material que se deben tomar en cuenta van desde el tamaño medio de las partículas que lo conforman, densidad, capacidad de carga. El material formará parte del cuerpo del canal por lo que se deben de considerar sus propiedades para disminuir el arrastre de sedimentos y erosión, altura de taludes, pérdidas de caudal por infiltraciones, o bien será el soporte del recubrimiento. En este caso se debe de considerar la capacidad de carga del suelo. El factor económico ayuda a determinar la conveniencia o no de recubrir el canal, según las pérdidas de agua por filtración.

Los costos de construcción y mantenimiento de un canal son razones suficientes para destacar la importancia de un diseño adecuado del mismo, ya que si el nivel del agua sobrepasara la altura de taludes se podría ocasionar un deslave del material y la consecuente destrucción del

---

\* Considerando sólo la porción dentro de nuestro país.

canal en ese tramo. Aún más, si el canal forma parte de una obra de protección son vidas humanas las que se pondrían en riesgo, sin contar con los daños económicos a la misma población en sus bienes materiales.

## II CANALES DE SECCIONES COMPUESTAS

### II.1 Geometría

El concepto de pendiente del canal para una sección sencilla no varía cuando se analiza una sección compuesta. En cuanto a la sección transversal, la sección compuesta debe su nombre a que el canal se divide en dos o más subsecciones para su estudio, teniendo un canal más profundo y subsecciones laterales que suelen designarse como bermas y pueden ser simétricas o asimétricas. La sección trapezoidal es la de uso más común en el diseño, dada la facilidad de su construcción en campo. Las secciones rectangular y triangular quedan como casos particulares de la primera. En la figura 2.1 se muestra la geometría de una sección trapezoidal y en la tabla 2.1 se presentan las ecuaciones para obtener las características geométricas de la misma sección.

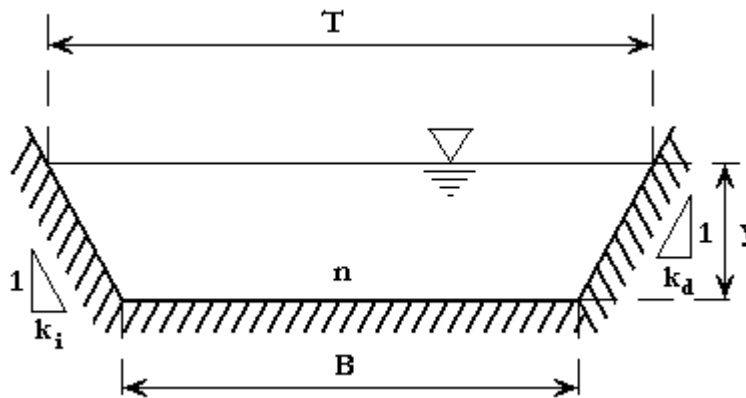


Fig. 2.1. Sección trapezoidal

Tabla. 2.1. Características geométricas de la sección trapezoidal

Características geométricas	Fórmula
Área	$A = By + \frac{(k_i + k_d)}{2} y^2$
Perímetro mojado	$P = B + \left( \sqrt{k_i^2 + 1} + \sqrt{k_d^2 + 1} \right) y$
Radio hidráulico	$R_h = \frac{A}{P}$
Ancho de superficie libre	$T = B + (k_i + k_d) y$
$\frac{dT}{dy}$	$\frac{dT}{dy} = k_i + k_d$
$\frac{dP}{dy}$	$\frac{dP}{dy} = \sqrt{k_i^2 + 1} + \sqrt{k_d^2 + 1}$

En el caso más general, los canales de sección compuesta son asimétricos geoméricamente y la rugosidad para cada subsección es diferente. En la figura 2.2 y en la tabla 2.2 se muestran la geometría y las características geométricas para una sección compuesta.

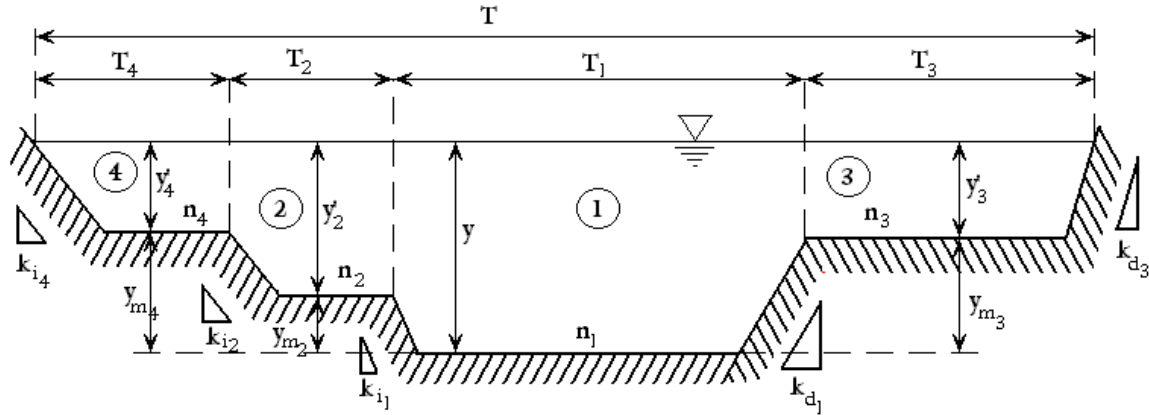


Fig. 2.2. Sección compuesta

Tabla. 2.2. Características geométricas de la sección compuesta

Características geométricas	Fórmula
Área	$A = \sum_{i=1}^n A_i$
Perímetro mojado	$P = \sum_{i=1}^n P_i$
Radio hidráulico	$R_b = \frac{A}{P}$
Ancho de superficie libre	$T = \sum_{i=1}^n T_i$
$\frac{d\Gamma}{dy}$	$\frac{d\Gamma}{dy} = \sum_{i=1}^n \frac{dT_i}{dy}$
$\frac{dP}{dy}$	$\frac{dP}{dy} = \sum_{i=1}^n \frac{dP_i}{dy}$

Para el cálculo de las características geométricas de la  $i$ -ésima sección se considera un tirante  $y'_i = y - y_{mi}$ , donde  $y$  es el tirante en la sección en metros y  $y_{mi}$  es la altura de la berma o llanura de inundación en la sección  $i$ .

En el caso del perímetro mojado de la subsección  $i$ , según el autor, se incluyen o no la longitud de las intercaras verticales agua-agua entre cada subsección para considerar el efecto de transferencia de momentum entre el canal principal y las llanuras de inundación. Lo más aconsejable es no incluirlas, para mantener el concepto general de dicho perímetro.

## II.2 Casos prácticos

Los canales de sección compuesta se diseñan para casos particulares en los que llega a existir una gran diferencia entre el gasto promedio de funcionamiento y el gasto esperado en el caso de una avenida. Al considerar un canal de dimensiones menores en la zona más profunda y con un caudal pequeño se respeta la velocidad mínima permisible; además, al tener llanuras de inundación, la capacidad de descarga del canal aumenta y se permite conducir el gasto de una avenida.

Como ejemplo podemos citar al río Tijuana, localizado aguas abajo de la presa Abelardo L. Rodríguez, cerca de la frontera con Estados Unidos. Este río fue canalizado recientemente, tiene una longitud desde la presa hasta la frontera de 17 km. La sección transversal tiene un canal más profundo que transporta las aguas residuales de la ciudad. Al considerar las llanuras de inundación, el canal fue diseñado para un gasto de  $2\,100\text{ m}^3/\text{s}$  en las primeras dos secciones de su trazo y  $3\,620\text{ m}^3/\text{s}$  en su sección final. Las dimensiones del canal se aprecian en la figura 2.3. Imágenes del río canalizado se presentan en las figuras 2.4 y 2.5.

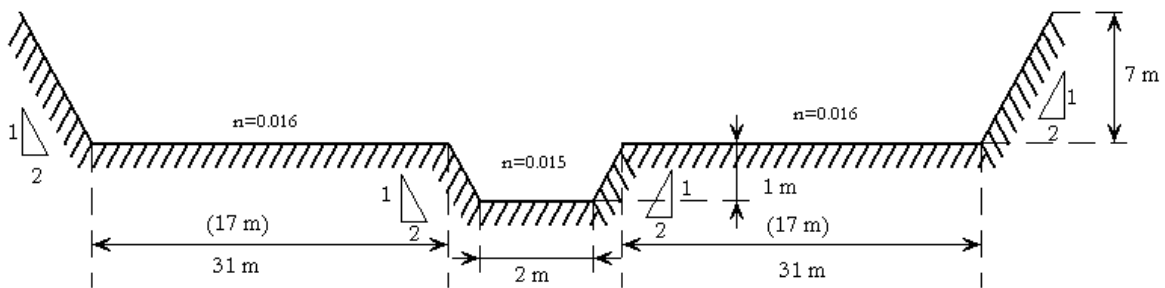


Fig. 2.3. Sección transversal del río Tijuana.  
El ancho de las bermas de las secciones II y III es de 32 m



Fig. 2.4. Vista transversal del río Tijuana



Fig. 2.5. Vista lateral del río Tijuana

Otro ejemplo de canal de sección compuesta lo podemos encontrar en el río Magdalena en la zona comprendida dentro del parque de los viveros en Coyoacán.

## II.3 Flujo uniforme

El flujo uniforme ocurre en canales de gran longitud, de sección transversal y pendiente constantes y sin curvas horizontales, suponiendo que el flujo se comporta unidimensionalmente. A pesar de que las condiciones idealizadas son difíciles de encontrar en la realidad, este tipo de flujo es básico en los problemas de diseño. La ecuación más aceptada para describir el flujo uniforme es la ecuación de Chezy, considerando el coeficiente de Manning, es

$$Q = \frac{AR_b^{2/3}S^{1/2}}{n} \quad (2.1)$$

donde  $Q$  es el gasto en  $m^3/s$ ,  $A$  el área de la sección transversal en  $m^2$ ,  $R_b$  el radio hidráulico en  $m$ ,  $S$  la pendiente del canal y es adimensional y  $n$  el coeficiente de Manning.

También se puede expresar como sigue

$$Q = KS^{1/2} \quad (2.2)$$

donde  $K$  es el factor de conducción expresado por la ecuación

$$K = \frac{AR_b^{2/3}}{n} \quad (2.3)$$

Sin embargo, para los canales de sección compuesta, investigaciones realizadas (Posey, 1967) han establecido que cuando  $y' \geq 0.5y$  el flujo uniforme se puede evaluar considerando los valores totales de  $A$ ,  $P$  y  $R_b$ , salvo el valor del coeficiente de Manning el cual corresponde a un coeficiente equivalente  $n_e$ . Horton, 1933 y Einstein, 1934 proponen obtener el valor de  $n_e$  con la ecuación 2.4, que Krishnamurthy y Christensen, en 1972, demostraron como la más precisa de las publicadas a la fecha.

$$n_e = \left[ \frac{\sum_{i=1}^n P_i n_i^{1.5}}{P} \right]^{2/3} = \left[ \frac{P_1 n_1^{1.5} + P_2 n_2^{1.5} + \Lambda + P_n n_n^{1.5}}{P} \right]^{2/3} \quad (2.4)$$

Para los casos en que  $y' < 0.5y$  se presentan fuerzas de fricción distintas entre el subcanal principal y los subcanales laterales. Existen estudios que demuestran que el gasto total es menor para el mismo nivel del agua que el que se esperaría si los gastos en cada subsección se calcularan por separado y después se sumaran, sin considerar interacción. Varios intentos se han hecho para cuantificar la transferencia de momentum usando criterios de intercaras imaginarias entre el canal principal y los canales laterales en los que se incluyen o se excluyen como perímetro mojado y que se definen en distintos sitios, con o sin la consideración de un esfuerzo tangencial aparente actuando sobre ellas. Además, el criterio empleado debe considerar que de la correcta distribución del gasto en cada subsección depende el valor del

coeficiente  $\alpha$  de coriolis que corrige la carga de velocidad en la ecuación de la energía cuando se idealiza el flujo unidimensional.

Cualquiera que sea el criterio que se considere para evaluar la interacción entre las subsecciones del canal compuesto, el flujo uniforme se obtiene aplicando por separado la ecuación de Manning para cada subsección y luego sumando los gastos, de la manera siguiente.

considerando que

$$V_i = \frac{K_i}{A_i} S^{1/2}$$

donde  $V$  es la velocidad media del flujo en la sección  $i$ . De la ecuación de continuidad, se tiene

$$Q = VA = \sum_{i=1}^n V_i A_i$$

Siendo la pendiente igual para todas las subsecciones, resulta

$$Q = \left( \sum_{i=1}^n K_i \right) S^{1/2} \quad (2.5)$$

por lo que

$$K = \sum_{i=1}^n K_i \quad (2.6)$$

Los coeficientes de coriolis ( $\alpha$ ) y de Boussinesq ( $\beta$ ) se obtienen a partir de la velocidad media en cada subsección y de acuerdo con su definición quedarían de la manera siguiente:

$$\alpha = \frac{1}{A} \iint_A \frac{v^3}{V^3} dA \cong \frac{1}{A} \frac{\sum_{i=1}^n (\alpha_i K_i^3 / A_i^2)}{\sum_{i=1}^n (K_i / A)^3}$$

$$\alpha = \frac{A^2}{K^3} \sum_{i=1}^n \frac{\alpha_i K_i^3}{A_i^2} \quad (2.7)$$

Para el coeficiente de Boussinesq queda

$$\beta = \frac{1}{A} \iint_A \frac{v^2}{V^2} dA$$

$$\beta = \frac{A}{K^2} \sum_{i=1}^n \frac{\beta_i K_i^2}{A_i} \quad (2.8)$$

## II.4 Régimen crítico

El régimen crítico en un canal se presenta cuando el número de Froude  $\mathbf{F}$  es igual a 1. En el caso de los canales de sección sencilla se define de la manera siguiente.

$$\mathbf{F} = \frac{V}{\sqrt{\frac{A}{T} \frac{g \cos \theta}{\alpha}}} \quad (2.9)$$

El régimen crítico obtiene importancia ya que indica las condiciones en que, a gasto constante, éste se conduce con energía específica mínima; al igual que con energía específica constante, se presenta el gasto máximo. Lo que nos permite usar esta condición como sección de control en el cálculo de perfiles de flujo. En la práctica se ha observado que ocurre en condiciones muy inestables, por lo que en la etapa de diseño se busca que el tirante normal del flujo uniforme, se encuentre lo más alejado de esta condición. Adicionalmente, cuando  $\alpha = 1$ ,  $\beta = 1$ , la condición de energía específica crítica corresponde también a la de momentum mínimo.

Sin embargo, la determinación de las condiciones de régimen crítico para los canales de sección compuesta se complica aún más, ya que la interacción entre las subsecciones del canal altera el flujo y permite que exista más de un tirante crítico en la sección. Además de lo anterior, no existe un criterio único para definir la condición de ocurrencia del flujo crítico; este fenómeno es tema del siguiente capítulo.

Referencias.

Einstein, H. A., “Der Hydraulische oder profil-radius”, Schweizerische Bauzeitung 108, número 103, páginas 89 a 91, Zurich, febrero 24 de 1934.

Horton, R. A., “Separate roughness coefficients for channel bottom and sides”, Engineering News Record III, número 22, páginas 652 a 653, noviembre 30 de 1933.

Posey C. J., “Computation of discharge including over-bank flow”, Civil Engineering American Society of Civil Engineers, páginas 62 a 63, abril 1967.



### III RÉGIMEN CRÍTICO EN SECCIONES COMPUESTAS

#### III.1 Criterio de energía específica mínima

En 1919, Böss demostró que la transición de flujo tranquilo (subcrítico) a un flujo rápido (supercrítico) pasa por un tirante crítico que minimiza la energía específica. Tomando en cuenta lo anterior y a partir de la ecuación de la energía se puede establecer que la energía específica vale.

$$E = y \cos \theta + \alpha \frac{V^2}{2g} = y \cos \theta + \alpha \frac{Q^2}{2gA^2} \quad (3.1)$$

que al derivar parcialmente respecto al tirante, conservando el gasto constante e igualando a cero, queda

$$\left( \frac{\partial E}{\partial y} \right)_Q = \cos \theta - \alpha \frac{Q^2}{gA^3} \frac{\partial A}{\partial y} + \frac{Q^2}{2gA^2} \frac{\partial \alpha}{\partial y} = 0 \quad (3.2)$$

la cual se expresa también como

$$\frac{Q^2 T}{gA^3} = \frac{1}{\alpha} \left( \cos \theta + \frac{Q^2}{2gA^2} \frac{\partial \alpha}{\partial y} \right) \quad (3.3)$$

La ecuación 3.3 representa la condición general de régimen crítico en canales de sección compuesta, según el criterio de energía específica mínima. La ecuación 3.2 también se puede expresar como se muestra en la ecuación siguiente:

$$\left( \frac{\partial E}{\partial y} \right)_Q = \left[ 1 - \left( \frac{\alpha Q^2 T}{g' A^3} \frac{\partial A}{\partial y} - \frac{Q^2}{2g' A^2} \frac{\partial \alpha}{\partial y} \right) \right] \cos \theta \quad (3.4)$$

donde  $g' = g \cos \theta$ . Esta ecuación al ser comparada con la obtenida para canales de sección sencilla, conduce a que

$$\mathbf{F_B} = \left( \frac{\alpha Q^2 T}{g' A^3} \frac{\partial A}{\partial y} - \frac{Q^2}{2g' A^2} \frac{\partial \alpha}{\partial y} \right)^{1/2} \quad (3.5)$$

sea el número de Froude de la sección compuesta, el cual debe tener el valor de uno para el tirante crítico con el que ocurre la energía específica mínima. Las ecuaciones obtenidas con este criterio son las mismas si se considera la energía específica constante y se maximiza el gasto.

### III.2 Criterio de momentum mínimo

Otro criterio para determinar las condiciones de flujo crítico es el propuesto por Boussinesq al definir el tirante crítico con base en la minimación de la fuerza específica o cantidad de movimiento. Por lo tanto, a partir de la ecuación de cantidad de movimiento, resulta

$$M = \frac{\beta Q^2}{gA} + y'_G A \cos \theta \quad (3.6)$$

donde  $M$  es la función momentum y  $y'_G$  es la profundidad del centroide del área de la sección transversal. En esta función el primer término corresponde a la cantidad de movimiento del flujo a través de una sección del canal y el segundo es el empuje debido a la presión sobre el área de la sección. Al derivar parcialmente respecto del tirante, conservando el gasto constante e igualando a cero, se obtiene

$$\left( \frac{\partial M}{\partial y} \right)_Q = -\frac{\beta Q^2}{gA^2} \frac{\partial A}{\partial y} + \frac{Q^2}{gA} \frac{\partial \beta}{\partial y} + \frac{\partial}{\partial y} (y'_G A \cos \theta) = 0 \quad (3.7)$$

se puede demostrar que  $\frac{\partial}{\partial y} (y'_G A \cos \theta) = A \cos \theta$ , por lo que

$$-\frac{\beta Q^2}{gA^2} \frac{\partial A}{\partial y} + \frac{Q^2}{gA} \frac{\partial \beta}{\partial y} + A \cos \theta = 0$$

la cual se expresa también de la manera siguiente

$$\frac{Q^2 T}{gA^3} = \frac{1}{\beta} \left( \cos \theta + \frac{Q^2}{gA} \frac{\partial \beta}{\partial y} \right) \quad (3.8)$$

La ecuación 3.8 representa la condición general de régimen crítico en canales de sección compuesta, según el criterio de momentum mínimo.

Por otro lado, al considerar las ecuaciones de continuidad (3.9) y momentum (3.10) para flujo unidimensional gradualmente variado utilizadas por Yen en 1973, se tiene que:

$$\frac{\partial A}{\partial t} + \frac{\partial (AV)}{\partial x} = 0 \quad (3.9)$$

$$\frac{1}{g} \frac{\partial V}{\partial t} + \frac{V^2}{g} \frac{\partial \beta}{\partial t} + (2\beta - 1)V \frac{\partial V}{\partial x} + (\beta - 1) \frac{V^2}{gA} \frac{\partial A}{\partial x} + \frac{\partial y}{\partial x} = S_o - S_f \quad (3.10)$$

donde  $S_o$  es la pendiente del fondo,  $S_f$  la pendiente de fricción,  $x$  la coordenada espacial y  $t$  el tiempo.

Empleando la teoría de sistemas hiperbólicos, se pueden transformar en un par de ecuaciones diferenciales ordinarias, válidas a lo largo de familias de curvas características determinadas por la ecuación

$$\left(\frac{\partial x}{\partial t}\right) = \beta V \pm \sqrt{\frac{gA}{T} + V^2 \left(\beta^2 - \beta + \frac{A}{T} \frac{\partial \beta}{\partial t}\right)} \quad (3.11)$$

Siendo  $x$  una dirección característica que representa la dirección a lo largo de la cual viaja una perturbación,  $\partial x / \partial t$  es la velocidad absoluta de la onda de traslación. El signo positivo separa la dirección aguas abajo y negativo para aguas arriba. Ahora bien,  $\partial x / \partial t$  tendrá valor positivo en ambos sentidos, si el término  $\beta V$  es mayor que el radical. Este caso representa flujo supercrítico ya que la onda viaja solamente hacia aguas abajo. De manera similar,  $\partial x / \partial t$  será positivo en la dirección hacia aguas abajo y negativo hacia aguas arriba cuando  $\beta V$  sea menor que el radical. Esto representa el flujo subcrítico, en el cual la onda de traslación viaja en ambos sentidos. Es así que, para el flujo crítico se debe de cumplir que ambos términos sean iguales. A partir de lo anterior se puede establecer una definición para el número de Froude  $F_C$ .

$$F_C = \frac{\beta V}{\sqrt{\frac{gA}{T} + V^2 \left(\beta^2 - \beta + \frac{A}{T} \frac{\partial \beta}{\partial t}\right)}} \quad (3.12)$$

Con esta ecuación se cumple que  $F_C$  es mayor que uno para flujo supercrítico, menor que uno para subcrítico e igual a uno para la condición crítica.

### III.3 Método de Blalock y Sturm

#### III.3.1 Ecuaciones básicas

En 1981, Blalock y Sturm establecieron con claridad las dificultades asociadas a varios métodos disponibles para el cálculo del tirante crítico y definieron un número de Froude para la sección compuesta del canal, ecuación 3.5, que corrige localmente los puntos de energía específica mínima cuando  $F = 1$  e identifica los tirantes, también llamados críticos, para los que esto ocurre. Para ello, obtuvieron  $dE/dy$  considerando  $\alpha$  como función del tirante, ecuación 3.4.

$$\frac{dE}{dy} = \left[ 1 - \left( \frac{\alpha Q^2 T}{g' A^3} \frac{\partial A}{\partial y} - \frac{Q^2}{2g' A^2} \frac{\partial \alpha}{\partial y} \right) \right] \cos \theta$$

Considerando que el coeficiente de coriolis se puede expresar con la ecuación 2.7, donde  $K = \Sigma K_i$  de la ecuación 2.6, se obtiene que vale

$$\alpha = \frac{A^2}{K^3} \sum_{i=1}^n \frac{\alpha_i K_i^3}{A_i^2}$$

Por tanto el término  $d\alpha/dy$  de la ecuación 3.4, considerando  $\alpha_i$  constante, resulta

$$\frac{d\alpha}{dy} = \frac{A^2}{K^3} \frac{d}{dy} \sum \left( \frac{\alpha_i K_i^3}{A_i^2} \right) + \sum \left( \frac{\alpha_i K_i^3}{A_i^2} \right) \frac{d}{dy} \frac{A^2}{(\sum K_i)^3}$$

desarrollando queda

$$\frac{d\alpha}{dy} = \frac{A^2}{K^3} \sum \left[ \alpha_i \left( \frac{K_i}{A_i} \right)^2 \frac{dK_i}{dy} - 2\alpha_i \left( \frac{K_i}{A_i} \right)^3 T_i \right] + \sum \left( \frac{\alpha_i K_i^3}{A_i^2} \right) \left( \frac{2AT}{K^3} - \frac{A^2}{K^4} 3 \sum \frac{dK_i}{dy} \right) \quad (3.13)$$

Pero de la ecuación 2.3,  $K_i = AR_{hi}^{2/3}/n_i$ , donde  $n_i$  también cambia con el tirante por lo cual

$$\frac{dK_i}{dy} = \frac{d}{dy} \left( \frac{A_i R_{hi}^{2/3}}{n_i} \right) = \frac{2}{3} \frac{A_i}{R_{hi}^{1/3} n_i} \frac{dR_{hi}}{dy} + \frac{R_{hi}^{2/3} T_i}{n_i} - \frac{A_i R_{hi}^{2/3}}{n_i^2} \frac{dn_i}{dy}$$

donde  $R_{hi} = A_i/P_i$

$$\frac{dR_{hi}}{dy} = \frac{d}{dy} \left( \frac{A_i}{P_i} \right) = \frac{P_i T_i - A_i (dP_i/dy)}{P_i^2} = \frac{T_i}{P_i} - \frac{R_{hi}}{P_i} \frac{dP_i}{dy}$$

y al sustituir

$$\frac{dK_i}{dy} = \frac{2}{3} \frac{A_i}{R_{hi}^{1/3} n_i} \left( \frac{T_i}{P_i} - \frac{R_{hi}}{P_i} \frac{dP_i}{dy} \right) + \frac{R_{hi}^{2/3} T_i}{n_i} - \frac{A_i R_{hi}^{2/3}}{n_i^2} \frac{dn_i}{dy}$$

considerando la ecuación 2.3

$$\frac{dK_i}{dy} = \frac{5}{3} \frac{R_{hi}^{2/3} T_i}{n_i} - \frac{2}{3} \frac{R_{hi}^{5/3}}{n_i} \frac{dP_i}{dy} - \frac{A_i R_{hi}^{2/3}}{n_i^2} \frac{dn_i}{dy}$$

$$\frac{dK_i}{dy} = \frac{5}{3} \frac{K_i T_i}{A_i} - \frac{2}{3} \frac{K_i R_{hi}}{A_i} \frac{dP_i}{dy} - \frac{K_i}{n_i} \frac{dn_i}{dy}$$

$$\frac{dK_i}{dy} = \frac{1}{3} \left( \frac{K_i}{A_i} \right) \left( 5T_i - 2R_{hi} \frac{dP_i}{dy} - 3 \frac{A_i}{n_i} \frac{dn_i}{dy} \right) \quad (3.14)$$

donde  $dn_i/dy = 0$  cuando  $n_i$  se considera constante.

Al sustituir la ecuación 3.14 en la 3.13 tenemos

$$\begin{aligned} \frac{d\alpha}{dy} = & \frac{A^2}{K^3} \sum \left[ \alpha_i \left( \frac{K_i}{A_i} \right)^3 \left( 3T_i - 2R_{hi} \frac{dP_i}{dy} - 3 \frac{A_i}{n_i} \frac{dn_i}{dy} \right) \right] \\ & + \sum \left( \frac{\alpha_i K_i^3}{A_i^2} \right) \left\{ \frac{2AT}{K^3} - \frac{A^3}{K^4} \sum \left[ \left( \frac{K_i}{A_i} \right) \left( 5T_i - 2R_{hi} \frac{dP_i}{dy} - 3 \frac{A_i}{n_i} \frac{dn_i}{dy} \right) \right] \right\} \end{aligned} \quad (3.15)$$

se designan los parámetros

$$\sigma_1 = \sum \left[ \alpha_i \left( \frac{K_i}{A_i} \right)^3 \left( 3T_i - 2R_{hi} \frac{dP_i}{dy} - 3 \frac{A_i}{n_i} \frac{dn_i}{dy} \right) \right] \quad (3.16)$$

$$\sigma_2 = \sum \left( \frac{\alpha_i K_i^3}{A_i^2} \right) \quad (3.17)$$

$$\sigma_3 = \sum \left[ \left( \frac{K_i}{A_i} \right) \left( 5T_i - 2R_{hi} \frac{dP_i}{dy} - 3 \frac{A_i}{n_i} \frac{dn_i}{dy} \right) \right] \quad (3.18)$$

y la ecuación 3.15 se puede simplificar como

$$\frac{d\alpha}{dy} = \frac{A^2}{K^3} \sigma_1 + \sigma_2 \left( \frac{2AT}{K^3} - \frac{A^2 \sigma_3}{K^4} \right) \quad (3.19)$$

Considerando que  $\alpha = A^2 \sigma_2 / K^3$ , al sustituir la ecuación 3.19 en la 3.5 se tiene que

$$\mathbf{F}_B = \left[ \frac{Q^2}{2g' K^3} \left( \frac{\sigma_2 \sigma_3}{K} - \sigma_1 \right) \right]^{\frac{1}{2}} \quad (3.20)$$

En las ecuaciones 3.16 y 3.18, los términos  $dP_i/dy$  y  $dn_i/dy$  representan la magnitud del cambio del perímetro mojado y de  $n$  en la subsección  $i$  respecto del tirante, siendo  $g = g'$ , cuando  $\theta$  es pequeño.

La ecuación 3.20 es una simplificación de la 3.5, y representa una nueva definición del número de Froude, que para  $\mathbf{F}_B = 1$  cumple con el criterio de energía específica mínima en un canal compuesto. Las ecuaciones 3.16 a 3.18 admiten valores de  $\alpha_i$  diferentes de 1, así como la variación del coeficiente de Manning en ellas.

### III.3.2 Variación de $n$ con el tirante

La variación del coeficiente de Manning es la única considerada en este método y es más importante en las subsecciones laterales con flujo de poca profundidad en ellas. Cuando la

pared se comporta como hidráulicamente rugosa, la variación de  $n$  se puede incluir con la ecuación de Nikuradse para el factor de fricción de Darcy-Weisbach en cada subsección

$$\frac{1}{\sqrt{f}} = \alpha_N \log \frac{cR_{hi}}{k_{si}}$$

donde  $k_{si}$  es la rugosidad equivalente de la pared de la subsección  $i$ ,  $c$  y  $\alpha_N$  son coeficientes que dependen de la geometría de la subsección (Sotelo, 2002). Considerando la ecuación de conversión de  $f$  a  $n$  resulta

$$\frac{k_{si}^{1/6}}{\sqrt{8gn_i}} = \left( \frac{k_{si}}{R_{hi}} \right)^{1/6} \frac{1}{\sqrt{f}}$$

por lo que el coeficiente de rugosidad de Manning queda definido como

$$n_i = \frac{R_{hi}^{1/6}}{\sqrt{8g\alpha_N} \log \frac{cR_{hi}}{k_{si}}} \quad (3.21)$$

donde  $n_i$  aumenta al disminuir el nivel del agua al tiempo que también decrece  $R_{hi}$ .

Al derivar la ecuación 3.21 se tiene

$$\frac{n_i}{dy} = \frac{1}{\sqrt{8g\alpha_N}} \left[ \frac{1}{6} \frac{1}{R_{hi}^{5/6} \log \frac{cR_{hi}}{k_{si}}} \frac{dR_{hi}}{dy} - \frac{R_{hi}^{1/6} (\log e)}{R_{hi} \left( \log \frac{cR_{hi}}{k_{si}} \right)^2} \frac{dR_{hi}}{dy} \right]$$

simplificando queda

$$\frac{n_i}{dy} = \frac{1}{R_{hi}} \frac{R_{hi}^{1/6}}{\sqrt{8g\alpha_N} \log \frac{cR_{hi}}{k_{si}}} \left[ \frac{1}{6} - \frac{(\log e) \sqrt{8g\alpha_N}}{R_{hi}^{1/6}} \frac{R_{hi}^{1/6}}{\sqrt{8g\alpha_N} \log \frac{cR_{hi}}{k_{si}}} \right] \frac{dR_{hi}}{dy}$$

con  $R_{hi} = A_i/P_i$ , al derivar se tiene

$$\frac{dR_{hi}}{dy} = \frac{T_i}{P_i} - \frac{A_i}{P_i^2} \frac{dP_i}{dy}$$

por lo que al sustituirla junto con la ecuación 3.21 en la anterior, se obtiene finalmente

$$\frac{A_i}{n_i} \frac{n_i}{dy} = \left[ \frac{1}{6} - (\log e) \sqrt{8g\alpha_N} \frac{n_i}{R_{hi}^{1/6}} \right] \left[ T_i - R_{hi} \frac{dP_i}{dy} \right] \quad (3.22)$$

La ecuación 3.22 se usa para el cálculo de los parámetros  $\sigma_1$  y  $\sigma_3$ .

Sturm y Sadiq en 1996 emplearon los coeficientes de Keulegan  $\alpha_N = 2$ ,  $c = 12.64$  en la ecuación 3.21, al aplicarla en el canal compuesto usado en sus experimentos, ver figura 3.1. Obtuvieron las rugosidades absolutas mediante pruebas de calibración tanto para el canal principal como para las dos laterales.

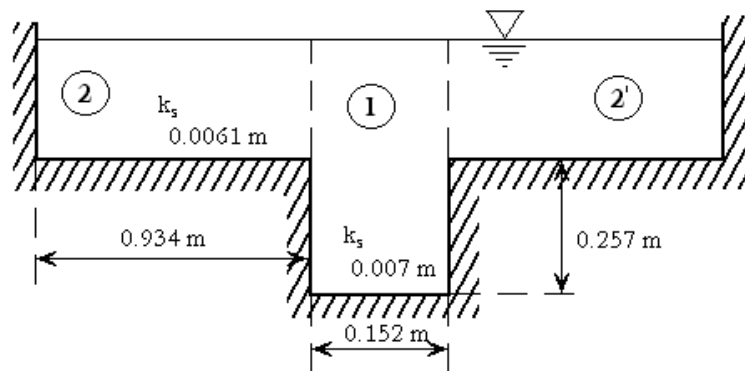


Fig. 3.1. Sección del canal

Los resultados obtenidos, en cuanto a la resistencia del flujo indicaron lo siguiente.

- La ecuación 3.21 predice muy bien el valor del coeficiente de Manning cuando el flujo no sobrepasa el nivel de desbordamiento hacia los laterales. Una vez sobrepasado el nivel, el valor de  $n$  en dicho canal resultó ser 1.19 veces el que se predice con la ecuación mencionada.
- En el caso de los canales laterales, el valor de  $n$  se ajustó al obtenido con la ecuación 3.21, una vez con flujo en ellos.

El factor de 1.19 observado en la predicción del coeficiente de Manning se atribuye a la interacción del flujo en las intercaras existentes con las de los laterales. El criterio de separación de las subsecciones es de líneas verticales que no aportan al perímetro mojado de ellas. A pesar de que los autores recomiendan utilizar el factor 1.19 para todos los canales simétricos, es necesario esperar más resultados al respecto para poder generalizar.

### III.3.3 Tirantes críticos múltiples

Para ilustrar la ocurrencia de tirantes críticos múltiples en canales de sección compuesta, se presenta el análisis de un canal cuya sección transversal se muestra en la figura 3.2. Por simplicidad y con fines de comparación entre los métodos presentados en este trabajo, se considera que  $\alpha_i = 1$  y que  $n_i$  es constante,  $dn_i/dy = 0$ . En la figura 3.3 se presenta una gráfica en la que se puede observar el comportamiento del número de Froude definido por la ecuación 3.20, para un gasto de  $Q_1 = 141.58 \text{ m}^3/\text{s}$ .

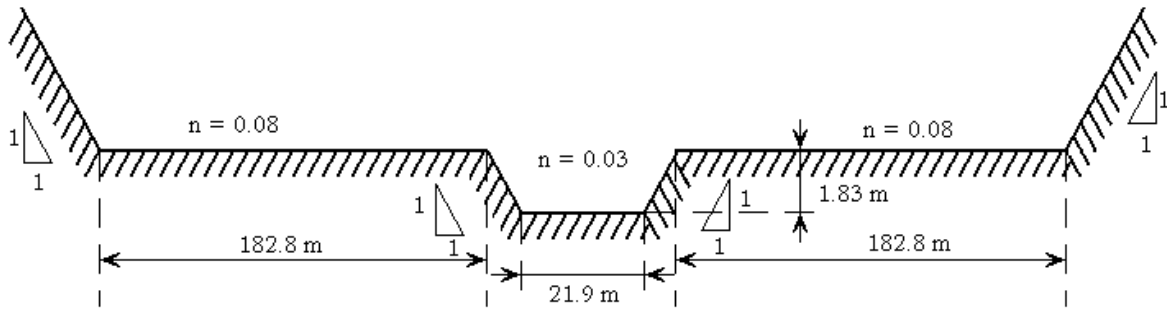


Fig. 3.2. Sección del canal

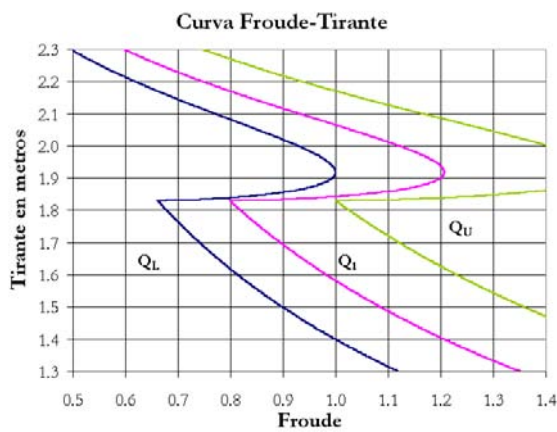


Fig. 3.3. Gráfica Froude-Tirante

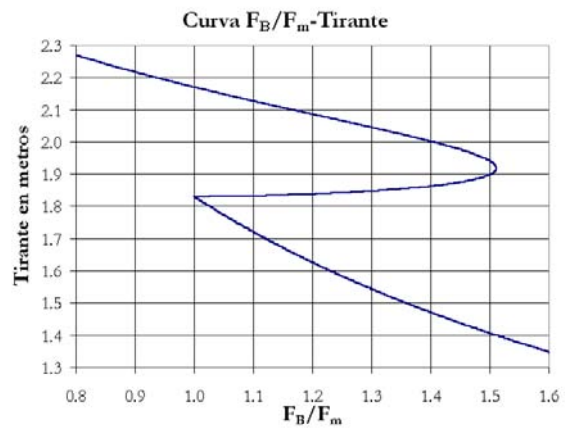


Fig. 3.4. Gráfica  $F_B/F_m$ -Tirante

En la gráfica de la figura 3.3 se puede apreciar que el número de Froude es igual a uno para tres puntos diferentes  $y_{c1} = 1.5815 \text{ m}$  y  $y_{c2} = 2.0644 \text{ m}$  que son los tirantes críticos obtenidos con este criterio, además de un tercero igual a  $1.8423 \text{ m}$ , muy cerca del nivel de berma y que no se considera como tirante crítico ya que representa más bien un máximo de energía específica.

En el caso particular analizado se presentan dos tirantes críticos; sin embargo, es posible que se presente un solo tirante crítico, ya sea en el canal central debajo del nivel de inundación de las bermas para un gasto pequeño o sobre el mencionado nivel para un caudal grande. De acuerdo con lo anterior y por razones de diseño, es conveniente determinar los gastos límite superior e inferior que encierran el rango de caudales para los que se presenta más de un solo tirante crítico. Sturm y Sadiq, en 1996, propusieron eliminar el efecto del gasto dividiendo la ecuación 3.20 entre el número de Froude  $F_m$  que corresponde al obtenido para la sección central en el nivel  $y = y_{m1}$ , es decir

$$F_m = \left( \frac{\alpha_m Q^2 T_m}{g' A_m^3} \right)^{1/2}$$



Así, se obtiene la ecuación 3.23

$$\frac{\mathbf{F}_B}{\mathbf{F}_m} = \frac{A_m^{3/2}}{\sqrt{2\alpha_m T_m}} \left[ \frac{1}{K^3} \left( \frac{\sigma_2 \sigma_3}{K} - \sigma_1 \right) \right]^{1/2} \quad (3.23)$$

donde no aparece el gasto y la función  $\mathbf{F}_B/\mathbf{F}_m$  depende sólo de la geometría del canal, la distribución de coeficientes de rugosidad y coeficientes  $\alpha_i$  en las subsecciones y del tirante  $y$ . Como se puede apreciar en la gráfica de la figura 3.4 la curva tiene un quiebre brusco en el punto donde  $y = y_{m1}$  y  $\mathbf{F}_B/\mathbf{F}_m = 1$ , después llega a un punto máximo.

Al establecer que el tirante crítico ocurre cuando  $\mathbf{F}_B = 1$ , existe un intervalo de valores de  $1/\mathbf{F}_m$  y por tanto un intervalo de gastos, dentro del cual hay dos tirantes críticos, uno inferior en la subsección más profunda  $y_{c1} < y_{m1}$  y uno superior  $y_{c2} > y_{m1}$  mayor que el valor  $y$  con el que se alcanza el máximo en la curva. El gasto límite superior  $Q_U$  del intervalo es el máximo para el cual ocurre el crítico  $y_{c1} = y_{m1}$ , es decir,  $\mathbf{F}_B = \mathbf{F}_m = 1$  y  $\mathbf{F}_B/\mathbf{F}_m = 1$  para  $Q = Q_U$ , lo que corresponde al punto de quiebre en la curva. Por tanto, de la definición  $\mathbf{F}_m = 1$ , se tiene

$$Q_U = \left( \frac{g' A_m^3}{\alpha_m T_m} \right)^{1/2}$$

El gasto límite inferior  $Q_L$  del intervalo es el último para el cual ocurre el crítico  $y_{c2} = y_{m1}$ , es decir  $\mathbf{F}_m < 1$ ,  $\mathbf{F}_B = 1$  y  $(\mathbf{F}_B/\mathbf{F}_m)_{\text{máx}}$  para  $Q = Q_L$ , de modo que

$$Q_L = \left( \frac{Q_U}{(\mathbf{F}_B/\mathbf{F}_m)_{\text{máx}}} \right)$$

En cuanto al ejemplo, en la gráfica de la figura 3.4 se puede ver que  $(\mathbf{F}_B/\mathbf{F}_m)_{\text{máx}} = 1.5116$ . Por otro lado, aplicando las ecuaciones anteriores, se tiene que el gasto límite superior es  $Q_U = 177.2871 \text{ m}^3/\text{s}$  y el caudal límite inferior es  $Q_L = 117.2794 \text{ m}^3/\text{s}$ . El comportamiento del número de Froude con respecto al tirante para estos dos gastos límite se puede apreciar en la gráfica de la figura 3.3.

Tomando en cuenta lo anterior y respecto a la multiplicidad de los tirantes críticos para un canal compuesto, se puede concluir lo siguiente:

- La curva  $\mathbf{F}_B/\mathbf{F}_m$  contra el tirante depende solamente de las propiedades geométricas e hidráulicas del canal.
- La curva presenta un punto de quiebre brusco cuando el nivel del agua llega a la altura de las bermas laterales y un punto máximo un nivel más arriba. Ambos puntos delimitan los gastos mínimo y máximo en que puede ocurrir más de un tirante crítico en la sección.
- Con los gastos límite  $Q_U$  y  $Q_L$ , se pueden presentar cualquiera de los siguientes casos para un caudal dado:

- a) Si  $Q < Q_L$ , existe un sólo tirante crítico y se ubica dentro de la subsección de mayor profundidad.
  - b) Si  $Q_L < Q < Q_U$ , existen dos tirantes críticos, uno se ubica dentro de la subsección de mayor profundidad y el segundo sobre el nivel de inundación.
  - c) Si  $Q > Q_U$ , existe un solo tirante crítico y se ubica sobre el nivel de inundación.
- Cuando hay una segunda o más ampliaciones de la sección a niveles superiores a la primera, se producen nuevos quiebres y máximos de la curva  $F_B/F_m$ , que acotan nuevos intervalos del gasto, más cortos o más amplios que el primero, y la posibilidad de más tirantes críticos.

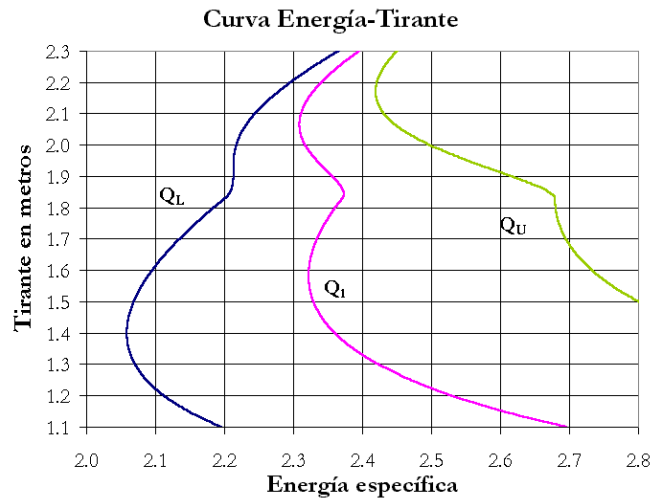


Fig. 3.5. Gráfica Energía-Tirante

En la gráfica de la figura 3.5 se presenta el comportamiento de la energía específica con respecto al tirante y se pueden ver los valores de energía mínimas que representan los puntos de régimen crítico para cada gasto.

### III.3.4 Algoritmo

El tirante crítico  $y_{c1} < y_{m1}$  se determina con el procedimiento convencional considerando el canal más profundo como un canal sencillo. Para determinar cualquier otro tirante crítico,  $y_{ci} > y_{m1}$ , se puede emplear el proceso iterativo propuesto por Sotelo en 1998.

De la ecuación 3.20 se debe cumplir que

$$F_B^2 = \frac{Q^2}{2g'K^3} \left( \frac{\sigma_2\sigma_3}{K} - \sigma_1 \right) = 1$$

Al multiplicar por  $\alpha K^3/A^2$ , resulta

$$\frac{\alpha Q^2}{2g'A^2} \left( \frac{\sigma_2 \sigma_3}{K} - \sigma_1 \right) = \frac{\alpha K^3}{A^2}$$

Pero  $\frac{\alpha Q^2}{2g'A^2} = E - y$ , y  $\frac{\alpha K^3}{A^2} = \sigma_2$ , por tanto

$$E - y = \frac{\sigma_2}{\left( \frac{\sigma_2 \sigma_3}{K} - \sigma_1 \right)}$$

de manera que el tirante en la iteración  $i+1$  resulta

$$y_{i+1} = E_i - \left( \frac{\sigma_2}{\left( \frac{\sigma_2 \sigma_3}{K} - \sigma_1 \right)} \right)_i \quad (3.24)$$

donde

$$E_i = y_{i+1} + \frac{\alpha_i Q^2}{2gA_i^2}$$

Por tanto, el algoritmo consiste de los siguientes pasos.

1. Se determina un valor  $y_{cl}$  del tirante crítico con las dimensiones de la subsección más profunda, utilizando el método convencional para canales de sección sencilla.
2. Si  $y_{cl}$  es mayor que  $y_{m1}$ , el tirante crítico  $y_{cl}$  no existe y queda cancelado. Si  $y_{cl}$  es menor o igual que  $y_{m1}$ , el tirante crítico  $y_{cl}$  queda con el valor calculado. En cualquier caso, se continúa con el paso 3.
3. Se elige un valor inicial del tirante  $y_0$  que sea ligeramente mayor que  $y_{m1}$ .
4. Se calculan los valores de  $K$ ,  $\alpha$ ,  $E$ ,  $\sigma_1$ ,  $\sigma_2$  y  $\sigma_3$  para el tirante elegido  $y_i$  en el paso 3 y con ellos se determina un nuevo valor  $y_{i+1}$  con la ecuación 3.24.
5. Si  $y_{i+1}$  es menor que  $y_{m1}$ , el tirante crítico  $y_{cl}$  queda cancelado y se sigue con el paso 7. Si  $y_{i+1}$  es mayor o igual que  $y_{m1}$  se continúa con el paso 6.
6. Se obtiene el error  $e = \left| \frac{y_{i+1} - y_i}{y_i} \right|$ . Si  $e$  es menor que una tolerancia establecida, el tirante crítico  $y_{cl}$  existe con el valor calculado y se sigue con el paso 7. Si  $e$  es mayor que la tolerancia, se repite el proceso desde el paso 3.
7. Si no existen más ampliaciones el proceso concluye, de lo contrario se continúa con el paso 8.
8. El número de veces que se repite el proceso es igual al número de alturas de bermas diferentes que existen en la sección del canal, es decir, para todos los  $y_{mj}$  distintos en el canal.

8. Se repite el proceso desde el paso 2 considerando  $y_{i2}$  en lugar de  $y_{i1}$  y  $y_{m2}$  en vez de  $y_{m1}$ , y así sucesivamente hasta concluir con todos los procesos, de tal manera que se cambiaran los  $y_{i+1}$  por  $y_{ij}$  y  $y_{mj+1}$  por  $y_{mj}$ .

### III.4 Método de Chaudhry y Bhallamudi

#### III.4.1 Ecuaciones básicas

Con base en el principio de momentum mínimo Chaudhry y Bhallamudi (1988) presentaron un método de cálculo de tirantes críticos. A partir de la ecuación 3.8 que establece la condición general de régimen crítico en canales compuestos se puede establecer

$$\frac{gA \cos \theta}{Q^2} = \frac{T\beta}{A} - \frac{\beta'}{A} \quad (3.25)$$

donde  $\cos \theta = 1$  para pendientes pequeñas y  $\beta' = d\beta/dy$ . Si se considera que  $\beta_i = 1$  en la ecuación 2.8, se tiene que

$$\beta = \frac{A}{K^2} \sum \frac{K_i^2}{A_i} = A \sum \left( \frac{K_i}{K} \right)^2 \frac{1}{A_i} \quad (3.26)$$

al derivar con respecto al tirante queda

$$\beta' = \frac{dA}{dy} \sum \left( \frac{K_i}{K} \right)^2 \frac{1}{A_i} - A \sum \left[ \left( \frac{K_i}{KA_i} \right)^2 \frac{dA_i}{dy} \right] + 2A \sum \left[ \frac{K_i}{KA_i} \frac{d}{dy} \left( \frac{K_i}{K} \right) \right]$$

Considerando que  $T = dA/dy$  y la inclusión de la ecuación 3.26, se tiene

$$\beta' = \frac{T}{A} \beta - A \sum \left[ \left( \frac{K_i}{KA_i} \right)^2 \frac{dA_i}{dy} \right] + 2A \sum \left[ \frac{K_i}{KA_i} \frac{d}{dy} \left( \frac{K_i}{K} \right) \right]$$

y al sustituir en la ecuación 3.25, resulta

$$\frac{gA}{Q^2} = \sum \left[ \left( \frac{K_i}{KA_i} \right)^2 \frac{dA_i}{dy} \right] - 2 \sum \left[ \frac{K_i}{KA_i} \frac{d}{dy} \left( \frac{K_i}{K} \right) \right] \quad (3.27)$$

que es válida para cualquier sección compuesta de cualquier geometría, donde

$$\frac{d}{dy} \left( \frac{K_i}{K} \right) = \frac{1}{K} \frac{dK_i}{dy} - \frac{K_i}{K^2} \frac{dK}{dy} \quad (3.28)$$

y el término  $dK_i/dy$  se obtiene de la ecuación 3.14, siendo  $dK/dy = \Sigma(dK_i/dy)$ .

Si se establece a  $A_m$  como el área y  $T_1$  como el ancho de superficie libre, cuando el nivel del agua se encuentra a la altura de las bermas de inundación, es decir  $y = y_{m1}$ , se puede definir un parámetro adimensional  $C$  al multiplicar el primer término de la ecuación 3.27 por  $A_m^3/AT_1$ , quedando

$$\frac{gA_m^3}{Q^2T_1} = C \quad (3.29)$$

lo que lleva a

$$C = \frac{A_m^3}{AT_1} \left\{ \sum \left[ \left( \frac{K_i}{KA_i} \right)^2 \frac{dA_i}{dy} \right] - 2 \sum \left[ \frac{K_i}{KA_i} \frac{d}{dy} \left( \frac{K_i}{K} \right) \right] \right\}$$

y al multiplicar por  $A A_i^2/A_m^3$ , resulta

$$\frac{AA_1^2}{A_m^3} C = \sum \left[ \left( \frac{K_i}{K} \right)^2 \left( \frac{A_1}{A_i} \right)^2 \frac{1}{T_1} \frac{dA_i}{dy} \right] - 2 \sum \left[ \frac{K_i}{K} \frac{A_1}{T_1} \frac{A_1}{A_i} \frac{d}{dy} \left( \frac{K_i}{K} \right) \right] \quad (3.30)$$

El factor  $C$  es función de la geometría del canal y del caudal, así es que al conocer estos datos, de pueden resolver las ecuaciones 3.29 y 3.30 en forma simultánea mediante un procedimiento iterativo para obtener los valores de tirantes críticos cuando el nivel del agua es mayor a la altura de las bermas laterales.

Chaudhry particularizó las ecuaciones mencionadas para canales como el mostrado en las figuras 3.1 y 3.2 para facilitar el método. La sección compuesta se puede dividir en tres subsecciones: la central (1) y dos laterales iguales (2) y (2'). De esta manera se pueden definir las características geométricas como:

$$A = A_1 + 2A_2$$

$$P = P_1 + 2P_2$$

$$K = K_1 + 2K_2$$

$$T_1 = \frac{dA_1}{dy}$$

$$T_2 = \frac{dA_2}{dy}$$

Además, estableciendo el parámetro

$$m = \frac{K_1}{K} = \frac{K_1}{K_1 + 2K_2} \quad (3.31a)$$

que también se expresa como

$$\frac{K_2}{K} = \frac{1-m}{2} \quad (3.31b)$$

Usando también los siguientes parámetros

$$c_1 = \frac{T_2}{T_1} \quad (3.32)$$

$$c_2 = \frac{A_1}{A_2} \quad (3.33)$$

$$c_3 = \frac{A_1}{T_1} \frac{1}{P_2} \frac{dP_2}{dy} \quad (3.34)$$

Con  $T_i = dA_i/dy$ , la ecuación 3.30 se desarrolla como sigue:

$$\frac{AA_1^2}{A_m^3} C = \left(\frac{K_1}{K}\right)^2 + 2\left(\frac{K_2}{K}\right)^2 \left(\frac{A_1}{A_2}\right)^2 \frac{T_2}{T_1} - 2 \sum \left[ \frac{K_1}{K} \frac{A_1}{T_1} \frac{d}{dy} \left(\frac{K_1}{K}\right) + 2 \frac{K_2}{K} \frac{A_1}{A_2} \frac{A_1}{T_1} \frac{d}{dy} \left(\frac{K_2}{K}\right) \right]$$

considerando todos los parámetros anteriores queda

$$\frac{AA_1^2}{A_m^3} C = m^2 + \frac{(1-m)^2}{2} c_1 c_2^2 - 2 \frac{A_1}{T_1} \left[ m \frac{dm}{dy} + (1-m) c_2 \frac{d}{dy} \left(\frac{1-m}{2}\right) \right]$$

pero

$$\frac{d}{dy} \left(\frac{1-m}{2}\right) = -\frac{1}{2} \frac{dm}{dy}$$

por lo que

$$\frac{AA_1^2}{A_m^3} C = m^2 + \frac{(1-m)^2}{2} c_1 c_2^2 - 2 \left( m \frac{1-m}{2} c_2 \right) \frac{A_1}{T_1} \frac{dm}{dy} \quad (3.35)$$

Por otro lado, se tiene

$$\frac{dm}{dy} = \frac{d}{dy} \left( \frac{K_1}{K} \right) = \frac{1}{K} \frac{dK_1}{dy} - \frac{K_1}{K^2} \frac{dK}{dy} = \frac{1}{K} \frac{dK_1}{dy} - \frac{K_1}{K^2} \sum \frac{dK_i}{dy} \quad (3.36)$$

Con  $dP_i/dy = 0$ , de las ecuaciones 3.14 y 3.31a resulta

$$\frac{1}{K} \frac{dK_1}{dy} = \frac{1}{K} \left( \frac{5}{3} \frac{K_1 T_1}{A_1} \right) = \frac{m}{3} \frac{T_1}{A_1}$$

y para toda la sección compuesta es

$$\frac{K_1}{K^2} \sum \frac{dK_i}{dy} = \frac{K_1}{K^2} \left[ \frac{5}{3} \frac{K_1 T_1}{A_1} + 2 \left( \frac{5}{3} \frac{K_2 T_2}{A_2} - \frac{2}{3} K_2 \frac{R_{b2}}{A_2} \frac{dP_2}{dy} \right) \right]$$

Al sustituir las ecuaciones 3.32 y 3.34 se puede expresar también como

$$\frac{K_1}{K^2} \sum \frac{dK_i}{dy} = \frac{m}{3} [5m + 5(1-m)c_1c_2 - 2(1-m)c_3] \frac{T_1}{A_1}$$

por tanto la ecuación 3.36 se transforma en

$$\frac{dm}{dy} = \frac{m}{3} \frac{T_1}{A_1} + \frac{m}{3} [-5m - 5(1-m)c_1c_2 + 2(1-m)c_3] \frac{T_1}{A_1}$$

la que, al factorizar, también es

$$\frac{dm}{dy} = \frac{m}{3} (1-m) [5(1-c_1c_2) + 2c_3] \frac{T_1}{A_1}$$

Con lo anterior, la ecuación 3.35 se convierte en

$$\frac{AA_1^2}{A_m^3} C = \left[ m^2 + \frac{(1-m)^2}{2} c_1c_2 \right] - \frac{2}{3} m(1-m) [5(1-c_1c_2) + 2c_3] \left( m - \frac{1-m}{2} c_2 \right)$$

la que, al multiplicar por  $c_2/(2+c_2)$ , queda como

$$\frac{AA_1^2}{A_m^3} C \frac{c_2}{2+c_2} = \frac{c_2}{2+c_2} \left[ m^2 + \frac{(1-m)^2}{2} c_1c_2 \right] - \frac{2m(1-m)c_2}{3(2+c_2)} [5(1-c_1c_2) + 2c_3] \left( m - \frac{1-m}{2} c_2 \right) \quad (3.37)$$

Del primer término de la ecuación 3.37 se tiene

$$\frac{c_2}{2+c_2} = \frac{A_2/A_1}{2+A_2/A_1} = \frac{A_1}{A_1+2A_2} = \frac{A_1}{A} \quad (3.38)$$

quedando entonces

$$C \left( \frac{A_1}{A_m} \right)^3 = \frac{c_2}{2+c_2} \left[ m^2 + \frac{(1-m)^2}{2} c_1 c_2^2 \right] - \frac{2m(1-m)c_2}{3(2+c_2)} [5(1-c_1 c_2) + 2c_3] \left( m - \frac{1-m}{2} c_2 \right) \quad (3.39)$$

Además

$$\left( \frac{A_1}{A_m} \right)^3 = \left[ \frac{A_m + T_1(y - y_m)}{A_m} \right]^3 = \left[ 1 + \frac{T_1 y_m}{A_m} \left( \frac{y}{y_m} - 1 \right) \right]^3$$

de tal manera que la ecuación 3.39 se expresa finalmente como

$$C \left[ 1 + \frac{T_1 y_m}{A_m} \left( \frac{y}{y_m} - 1 \right) \right]^3 = \frac{c_2}{2+c_2} \left[ m^2 + \frac{(1-m)^2}{2} c_1 c_2^2 \right] - \frac{2m(1-m)c_2}{3(2+c_2)} [5(1-c_1 c_2) + 2c_3] \left( m - \frac{1-m}{2} c_2 \right) \quad (3.40)$$

Así es que la ecuación 3.40 expresa el factor  $C$  para un canal simétrico con sólo dos llanuras de inundación. La obtención de los tirantes críticos depende de la resolución de las ecuaciones 3.29 y 3.40 mediante un procedimiento iterativo. El algoritmo desarrollado por Chaudhry se explica más adelante, en el apartado III.4.4.

Por otro lado, aunque Chaudhry, a diferencia de Blalock, no propone ningún factor para el coeficiente de Manning del canal principal que tome en cuenta la interacción entre la subsección más profunda y las llanuras de inundación, los resultados presentados se realizaron considerando el mencionado factor con un valor de 1.19.

### III.4.2 Tirantes críticos múltiples

En forma similar a la empleada para ilustrar la ocurrencia de tirantes críticos con el método de Blalock, se analizará el caso del canal presentado en la figura 3.2 con el método de Chaudhry. Se considera que en cada subsección se tiene que  $\alpha_i = 1$ ,  $\beta_i = 1$  y que  $n_i$  es constante  $dn_i/dy = 0$ . En la figura 3.6 se puede observar el comportamiento del número de Froude, definido por la ecuación 3.12, para un gasto de 141.58 m<sup>3</sup>/s.

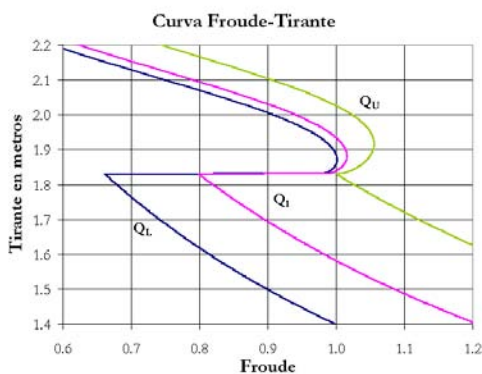


Fig. 3.6. Gráfica Froude-Tirante

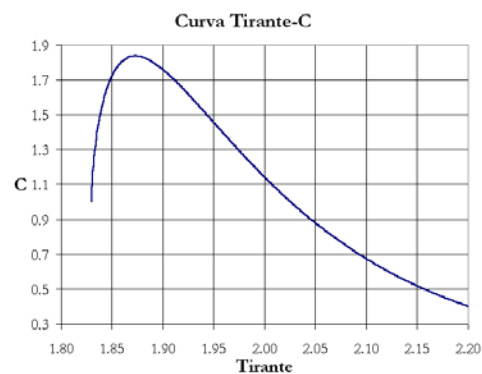


Fig. 3.7. Gráfica Tirante-C



En la gráfica de la figura 3.6 se puede apreciar que el número de Froude es igual a uno para tres puntos diferentes  $y_{c1} = 1.5815 \text{ m}$ ,  $y_{c2} = 1.8410 \text{ m}$  y  $y_{c3} = 1.9330 \text{ m}$ , que son los tirantes críticos obtenidos con este criterio. Chaudhry propone la existencia de los tres a diferencia de Blalock que no toma en cuenta el valor intermedio.

A partir de la ecuación 3.40 se puede establecer que

$$C = \left( \frac{A_m}{A_m + T_1(y - y_m)} \right)^3 \frac{c_2}{2 + c_2} \left[ m^2 + \frac{(1-m)^2}{2} c_1 c_2^2 \right] - \frac{2m(1-m)c_2}{3(2+c_2)} [5(1-c_1c_2) + 2c_3] \left( m - \frac{1-m}{2} c_2 \right) \quad (3.41)$$

Con la ecuación 3.41 se obtiene la gráfica tirante-C de la figura 3.7 para valores de tirante por encima del nivel de inundación. Con la geometría del canal y el gasto  $Q$ , se obtiene el valor de  $Ka = 1.5680$  utilizando la ecuación 3.29, que corresponde a los valores de  $y_{c2}$  y  $y_{c3}$ .

Por otro lado, en la gráfica 3.7 se puede ver que cuando el tirante se acerca al nivel de inundación, el valor de  $C$  tiende a uno; este caso es general para cualquier canal. Existe, además, un punto máximo  $C = 1.8371$  que depende exclusivamente de la geometría del canal. Estos dos puntos ayudan a determinar los caudales límite en los que se presentan tres tirantes críticos. Para el canal analizado son  $Q_L = 130.8010 \text{ m}^3/\text{s}$  y  $Q_U = 177.2871 \text{ m}^3/\text{s}$ , que se determinan despejando el gasto de la ecuación 3.29. Chaudhry demostró que el valor de  $C_{min}$  siempre es igual a uno.

De esta manera pueden presentarse cualquiera de los tres siguientes casos:

- a) Si  $k_a < 1$ , existe un sólo tirante crítico y se ubica sobre el nivel de inundación.
- b) Si  $1 < k_a < C_{m\acute{a}x}$  existen tres tirantes críticos, uno se ubica dentro de la subsección de mayor profundidad y los otros dos sobre el nivel de inundación.
- c) Si  $k_a > C_{m\acute{a}x}$ , existe un sólo tirante crítico y se ubica dentro de la subsección de mayor profundidad.

### III.4.3 Algoritmo

Chaudhry y Bhallamudi propusieron también un algoritmo para determinar primero el número de tirantes críticos en una sección para un gasto dado, para después calcular uno a uno todos los valores posibles.

1. Calcular  $k_a$  de la ecuación 3.29.
2. Si  $k_a$  es menor que uno, se determina  $C$  de la ecuación 3.41 para distintos valores de  $y$  empezando con uno mayor que  $y_m$  y continuando hasta que el calculado sea igual a  $k_a$  dentro de la tolerancia establecida.
3. Si  $k_a$  es mayor o igual a uno, continuar con los pasos del 4 al 9.

4. Calcular  $C_{m\acute{a}x}$  determinando la magnitud de  $C$  mediante la ecuaci3n 3.41 para distintos valores de  $y$ , a partir de uno inicial ligeramente mayor que  $y_m$  y terminarlo hasta que se alcance el valor mximo de  $C$ .
5. Si  $k_a$  es mayor que  $C_{m\acute{a}x}$ , entonces se obtiene  $y_{c1}$  al resolver la ecuaci3n  $Q^2/g = A^3/T$ .
6. Si  $k_a$  es menor que  $C_{m\acute{a}x}$ , se sigue con los pasos 7 a 9.
7. Calcular  $y_{c1}$  a partir de resolver:  $Q^2/g = A^3/T$ .
8. Obtener  $y_{c2}$  determinando diferentes valores de  $C$  a partir de la ecuaci3n 3.41 para distintos valores de  $y$  empezando con uno mayor que  $y_m$  y continuando hasta que el calculado sea igual a  $k_a$  dentro de la tolerancia establecida.
9. Se repite el proceso del paso 8 para obtener el valor de  $y_{c3}$  con un valor de  $y$  mayor que  $y_{c2}$ .

Este es el algoritmo que se emplea en el programa de cculo de tirantes crticos para canales simtricos con dos llanuras de inundaci3n, siguiendo el criterio de momentum mnimo propuesto por Chaudhry. Los detalles de la programaci3n se detallan en el captulo siguiente.

### III.5 Comparaci3n de ambos mtodos

#### III.5.1 Comparaci3n analtica

Una vez presentados los dos criterios ms aceptados para determinar el rgimen crtico en canales de secci3n compuesta, se presenta una comparaci3n de ellos. A partir de las ecuaciones 3.3 y 3.8 se puede establecer lo siguiente:

$$\frac{1}{\alpha} \left( \cos \theta + \frac{Q^2}{2gA^2} \frac{d\alpha}{dy} \right) = \frac{1}{\beta} \left( \cos \theta + \frac{Q^2}{gA^2} \frac{d\beta}{dy} \right)$$

o bien

$$\beta \left( \cos \theta + \frac{Q^2}{2gA^2} \frac{d\alpha}{dy} \right) = \alpha \left( \cos \theta + \frac{Q^2}{gA^2} \frac{d\beta}{dy} \right) \quad (3.42)$$

recordando que se puede establecer la siguiente relaci3n

$$\beta = 1 + \frac{\alpha - 1}{3} = \frac{\alpha + 2}{3} \quad (3.43)$$

adems

$$\beta' = \frac{1}{3} \alpha'$$

donde  $\beta' = d\beta/dy$  y  $\alpha' = d\alpha/dy$ . Por otro lado, al sustituir en la ecuación 3.42

$$(\alpha + 2) \left( \cos \theta + \frac{Q^2}{2gA^2} \alpha' \right) = 3\alpha \left( \cos \theta + \frac{Q^2}{gA^2} \alpha' \right)$$

y desarrollando

$$2 \cos \theta - 2\alpha \cos \theta + \frac{Q^2}{gA^2} \alpha' - \frac{\alpha}{2} \frac{Q^2}{gA^2} \alpha' = 0$$

ecuación que se puede expresar como

$$2 \cos \theta (1 - \alpha) + \frac{Q^2}{gA^2} \alpha' \left( 1 - \frac{\alpha}{2} \right) = 0 \quad (3.44)$$

La ecuación 3.44 es válida solamente cuando  $\alpha = 1$  y constante para toda la sección; lo que implica que con la expresión 3.43 se tenga  $\beta = 1$  y constante.

Los estudios realizados muestran que para los canales de sección compuesta los valores de  $\alpha$  y  $\beta$  son distintos de uno y cambian con la profundidad del flujo, por lo que no es factible que ambos criterios coincidan en la práctica.

### III.5.2 Comparación de resultados

Dado que la comparación analítica no presenta evidencia suficiente, se realizó una comparación de los resultados obtenidos para tres canales diferentes empleando ambos métodos. El primer canal analizado es el que se muestra en la figura 3.2. Los resultados obtenidos se muestran en la tabla 3.1. Este canal es el segundo que presenta Chaudhry en su artículo de 1988.

Tabla. 3.1. Resultados para el canal de la figura 3.2

Gasto (m <sup>3</sup> /s)	y <sub>c1</sub> (m)	y <sub>c2</sub> (m)		Error (%)
		Blalock	Chaudhry	
100.00	1.2607	-	-	-
117.2794	1.3990	1.9176	-	-
130.8010	1.5021	2.0217	1.8733	7.34
141.58	1.5815	2.0645	1.9330	6.37
177.2871	1.8300	2.1700	2.0258	6.65
200.00	-	2.2237	2.0714	6.85

El canal de la figura 3.8 corresponde al analizado exhaustivamente por Chaudhry en su mencionado trabajo de 1988. Los resultados se presentan en la tabla 3.2.

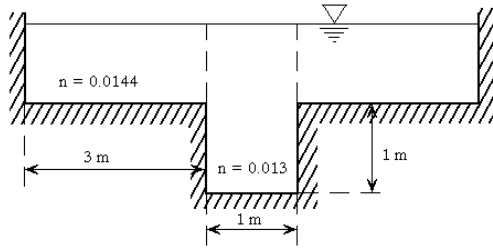


Fig. 3.8. Canal analizado

Tabla. 3.2. Resultados para el canal de la figura 3.8

Gasto (m <sup>3</sup> /s)	y <sub>e1</sub> (m)	y <sub>e2</sub> (m)		Error (%)
		Blalock	Chaudhry	
1.7	0.6654	-	-	-
2.0	0.7415	1.0850	1.0710	1.29
2.5	0.8605	1.1271	1.1128	1.27
3.0	0.9717	1.1593	1.1453	1.21
3.5	-	1.1871	1.1739	1.11

Adicionalmente se analizó el canal que presenta Sotelo en su trabajo publicado en 1998, modificando las rugosidades para cada subsección, de manera que se pudiese establecer la comparación de los métodos. En la figura 3.9 se tiene la sección transversal del canal mencionado, cuyos resultados se incluyen en la tabla 3.3.

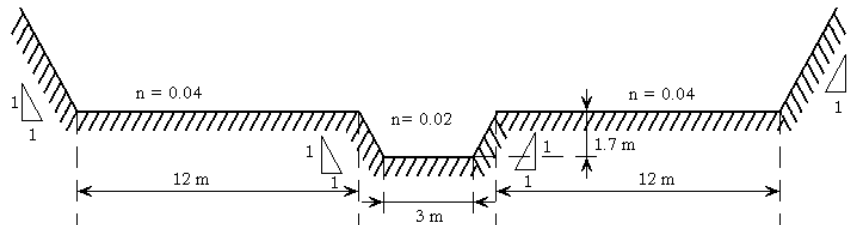


Fig. 3.9. Canal analizado

Tabla. 3.3. Resultados para el canal de la figura 3.9

Gasto (m <sup>3</sup> /s)	y <sub>e1</sub> (m)	y <sub>e2</sub> (m)		Error (%)
		Blalock	Chaudhry	
23.0	1.5219	-	-	-
24.0	1.5592	1.8325	1.7825	2.73
25.0	1.5958	1.8659	1.8127	2.85
26.0	1.6316	1.8933	1.8364	3.01
27.0	1.6668	1.9175	1.8573	3.14
28.0	-	1.9397	1.8765	3.26

A pesar de que a partir de los resultados de la tabla 3.2 se podría establecer que la diferencia entre ambos métodos es mínima y despreciable, los datos de las tablas 3.1 y 3.3 contradicen dicha afirmación. En la tabla 3.4 se incluyen las relaciones entre rugosidades y anchos de plantilla del canal principal y las llanuras de inundación; además del cociente de la altura de la berma y el nivel de las bermas.

Tabla. 3.4. Comparación entre los canales estudiados

Canal	$n_1/n_2$	$b_1/b_2$	$y_{m2}/b_2$	Error (%)
3.2	0.3750	0.1198	0.0100	6.80
3.8	0.9028	0.3333	0.3333	1.22
3.9	0.5000	0.2500	0.1467	3.00

Partiendo de los resultados presentados en la tabla 3.4, se analizaron varios canales para determinar la influencia de las relaciones  $n_1/n_2$ ,  $b_1/b_2$  y  $y_{m2}/b_2$ . Para cada sección se obtuvieron los tirantes críticos con cinco caudales diferentes ubicados dentro de los límites superior e inferior de acuerdo con el criterio de Chaudhry, dado que el criterio de Blalock arroja un intervalo más amplio. Se calculó el promedio de los errores para cada canal analizado. Los resultados se presentan en las tablas 3.5 y en las gráficas de la figura 3.10.

Tabla 3.5a. Porcentaje de error para  $b_1/b_2=1.00$

	$n_1/n_2$				
$y_{m2}/b_2$	1.00	0.50	0.33	0.25	0.10
1.00	0.88	1.58	1.71	2.07	1.19
0.50	1.19	1.87	1.89	2.12	0.84
0.25	1.38	2.15	2.11	2.01	0.59
0.10	0.88	1.58	1.71	2.07	1.24

Tabla 3.5b. Porcentaje de error para  $b_1/b_2=0.50$

	$n_1/n_2$				
$y_{m2}/b_2$	1.00	0.50	0.33	0.25	0.10
1.00	0.54	1.66	2.59	3.27	4.80
0.50	1.02	2.44	3.50	4.20	4.98
0.25	1.49	3.93	4.26	4.96	4.49
0.10	0.55	1.71	2.59	3.27	4.80

Tabla 3.5c. Porcentaje de error para  $b_1/b_2=0.33$

	$n_1/n_2$				
$y_{m2}/b_2$	1.00	0.50	0.33	0.25	0.10
1.00	0.28	1.37	2.31	3.11	5.92
0.50	0.75	2.17	3.35	4.31	7.24
0.25	1.27	3.04	4.44	5.49	7.47
0.10	0.24	2.26	2.31	3.11	5.92

Tabla 3.5d. Porcentaje de error para  $b_1/b_2=0.10$

	$n_1/n_2$				
$y_{m2}/b_2$	1.00	0.50	0.33	0.25	0.10
1.00	0.17	1.08	1.95	2.73	5.97
0.50	0.55	1.85	2.99	3.99	7.58
0.25	1.07	2.76	4.17	5.35	8.81
0.10	0.17	1.08	1.95	2.73	13.09

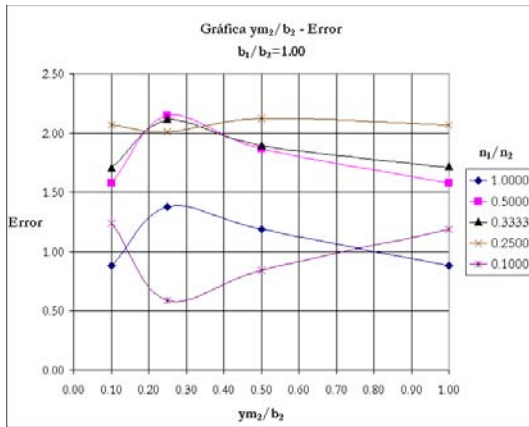


Figura 3.10.a. Gráfica para  $b_1/b_2=1.00$

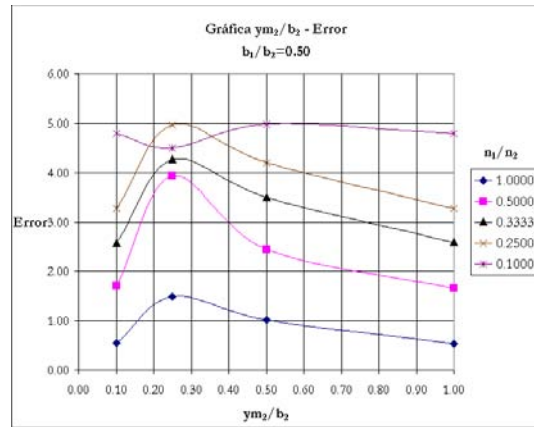


Figura 3.10.b. Gráfica para  $b_1/b_2=0.50$

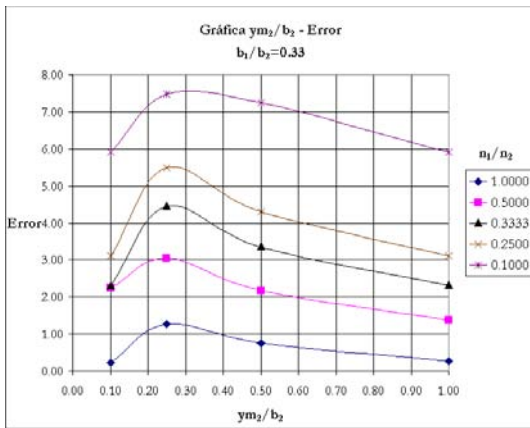


Figura 3.10.c. Gráfica para  $b_1/b_2=0.33$

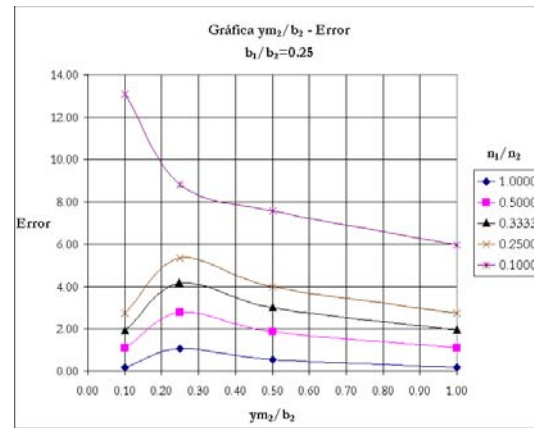


Figura 3.10.d. Gráfica para  $b_1/b_2=0.25$

De las tablas 3.5 y figuras 3.10 se puede deducir lo siguiente.

- El error encontrado para los canales de las figuras 3.2, 3.8 y 3.9 coincide con los obtenidos en este análisis extensivo.
- La diferencia entre los tirantes críticos calculados con ambos criterios es aceptable cuando la relación entre las rugosidades del canal central y las bermas de inundación,  $n_1/n_2$ , es cercana a la unidad y tiende a crecer si dicha relación tiende a cero.
- La relación entre los anchos de plantilla del canal más profundo y de las bermas,  $b_1/b_2$ , afecta la diferencia observada entre ambos criterios de tal manera que cuando la relación se acerca a la unidad el error tiende a cero y que al disminuir la relación el error tiende a crecer.
- El error es pequeño cuando el coeficiente  $y_{m2}/b_2$  se acerca a la unidad, crece cuando el valor del coeficiente se acerca a 0.25, y finalmente disminuye nuevamente si  $y_{m2}/b_2$  tiende a cero.

- e) Para que los resultados obtenidos con ambos criterios no difieran considerablemente basta con conservar la relación  $n_1/n_2$  entre 0.5 y 1.0, independientemente de las dimensiones del canal.
- f) Cuando los anchos de plantilla del canal principal y las bermas son muy similares, no importa el valor de los otros dos coeficientes considerados, el error es despreciable.
- g) No se puede establecer un rango específico en el que la relación  $y_{m2}/b_2$  asegure resultados similares para ambos métodos.

#### Referencias.

Blalock, M. E. Y Sturm T. W., 1981, "Minimum specific energy in compound channel", ASCE J Hydraulics Division, 107 (HY6):699-717

Boussinesq, J. V., "Essai sur la théorie des eaux courantes" Mém. Acad. Sciences, 23(2), 1-680, París, 1877

Böss, P., "Berechnung der Wasserspiegellage beim Wechsel des Fliesszustandes", Springer, Berlín, 1919.

Chaudhry M. Hanif y Bhallamudi S. Murty, 1988, "Computation of critical depth in symmetrical compound channels" J. Hydraulic Research, IARH 26(4).

Sotelo Ávila, Gilberto, "Hidráulica de canales", Facultad de Ingeniería, U.N.A.M., 2002.

Sturm, T. W. y A. Sadiq, 1996, "Water surface profiles in compound channel with multiple critical depths", ASCE, J. Hydraulic Engineering 122(12):703-709.

Yen Ben Chie "Open channel flow equations revisited", ASCE, Journal of the Engineering Mechanics Division, vol. 99, número EM5, octubre, 1973.

## IV PROGRAMA DE CÓMPUTO EN VISUAL BASIC

### IV.1 Consideraciones

Uno de los objetivos del presente trabajo de tesis es obtener un programa de cómputo que permita la obtención de las condiciones de régimen crítico para canales de sección compuesta empleando cualquiera de los métodos descritos en el capítulo anterior.

En la elección del lenguaje de programación se consideró que el programa fuera compatible con la mayoría de los equipos de cómputo y que fuera accesible a los estudiantes de la Facultad de Ingeniería de la UNAM. *Visual Basic* fue seleccionado porque proporciona un ambiente similar a *Windows* y puede ser instalado en cualquier sistema operativo *Windows 95* o superior, además de ser un lenguaje de fácil comprensión.

Para el manejo de los datos de cada canal se decidió que fueran almacenados en archivos con formato *Database V*, dado que pueden ser abiertos con cualquier versión de *Excel* en caso necesario y no tienen problemas de conflicto de versiones. Las características geométricas e hidráulicas del canal por analizar se almacenan en archivos independientes que pueden ser creados y abiertos en una ventana separada del proceso de cálculo.

El sistema se instala por descompactado en la carpeta de *Archivos de programa\Régimen crítico* dentro del disco duro, además crea dos subcarpetas, *... \sistema*, donde se guardan los archivos base, *canales.dbf* y *campos.dbf*, además del archivo de configuración del sistema, *tiraners.ini*. La segunda carpeta creada es *... \datos*, donde se almacenan los archivos de datos para cada canal.

La estructura de los archivos se copia de la que existe en *canales.dbf* y puede ser modificada si se desea. En la tabla 4.1 se muestra dicha estructura.

Tabla 4.1. Estructura de los archivos de datos

Campo	Nombre	Tipo	Ancho	Descripción
1	nseccion	Numérico	2.0	Número de subsección
2	nplantilla	Numérico	8.4*	Ancho de plantilla, m
3	ntaludi	Numérico	5.2	Talud izquierdo, si existe
4	ntaduldd	Numérico	5.2	Talud derecho, si existe
5	nberma	Numérico	20.5	Altura de la berma, m
6	ctipo	Carácter	1	Tipo de sección, 1 para central, 2 para izquierda y 3 para derecha
7	nalfa	Numérico	5.2	Coefficiente de coriolis
8	nc	Numérico	5.2	Coefficiente $c$ de Keulegan
9	nalfan	Numérico	5.2	Coefficiente $\alpha_n$ de Keulegan
10	nks	Numérico	10.5	Rugosidad equivalente, m
11	nn	Numérico	10.5	Coefficiente de Manning

\* El número decimal en el ancho de los campos de tipo numérico indica la cantidad de decimales que se pueden incluir en dicho campo.



El archivo *campos.dbf* contiene el nombre de los campos listados y son usados para el sistema, de igual manera, los registros en el mismo pueden ser modificados si así se desea. El archivo *tirantes.ini* no se incluye en el sistema, pero es creado al configurar el sistema y contiene el directorio donde se encuentran los archivos base.

Los datos del canal se capturan en una ventana en la que se verifica que no existan inconsistencias que generen errores en el momento del cálculo. Se incluye además una opción que muestra la sección del canal.

Para el proceso de cálculo se abre otra ventana en la que se proporcionan los valores de gasto y gravedad con los que se obtendrán las condiciones críticas. En esta ventana también se pueden proporcionar los factores que afectan al coeficiente de Manning para considerar el efecto en las intercaras de las subsecciones. Los resultados se pueden obtener los tirantes críticos con el método de Blalock considerando o no la variación del coeficiente  $n$  respecto del tirante para este último. Para usar el método de Chaudhry es necesario que se cumplan las características del canal compuesto que él propone. Además de encontrar todos los tirantes críticos posibles, se presentan los caudales límite en los que se presentan tirantes críticos múltiples sólo para secciones con dos llanuras de inundación al mismo nivel.

Para complementar los resultados y con fines didácticos, se incluyen también las iteraciones del proceso y una gráfica Energía específica-tirante.

En los siguientes incisos del presente capítulo se describe la solución planteada para dos de los problemas más importantes en el proceso de cálculo.

El Manual de Instalación y Operación del sistema se incluye en el anexo I, las pantallas en el anexo II y finalmente en el anexo III se presenta el código completo.

## **IV.2 Cálculo de las características geométricas**

Al analizar el problema de adaptar los procesos de cálculo a un programa de cómputo que resolviera cualquier canal de sección compuesta, el principal obstáculo era el de determinar las características geométricas para cada subsección sin importar el nivel del agua supuesto en cada parte del proceso.

Para determinar las subsecciones se consideraron fronteras verticales, a partir del punto donde el nivel del agua empieza a inundar la llanura de inundación. Ellas delimitan el área transversal pero no contribuyen a incrementar la longitud del perímetro mojado. Al tomar en cuenta lo anterior, las subsecciones se pueden dividir en otras subsecciones más, con lo que se definen los tipos de sección transversal indicados en las figuras 4.1.

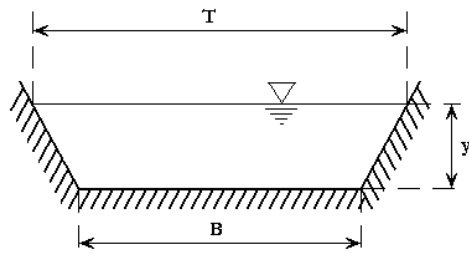


Figura 4.1.a. Sección tipo 0

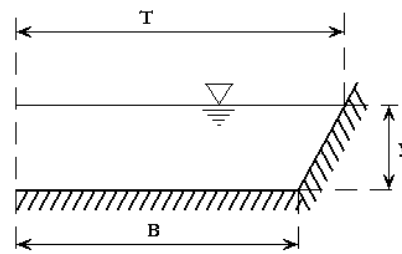


Figura 4.1.b. Sección tipo 1

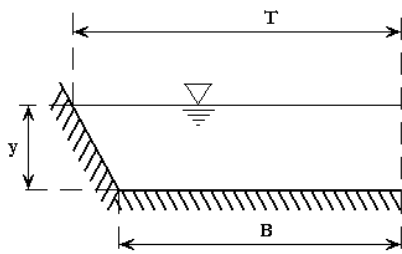


Figura 4.1.c. Sección tipo 2

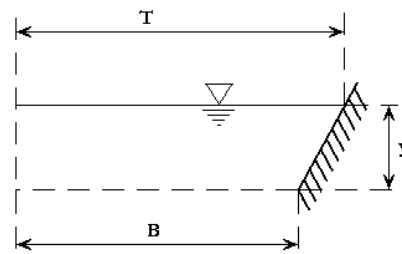


Figura 4.1.d. Sección tipo 3

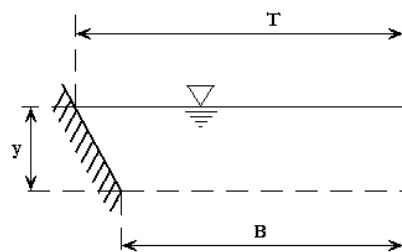


Figura 4.1.e. Sección tipo 4

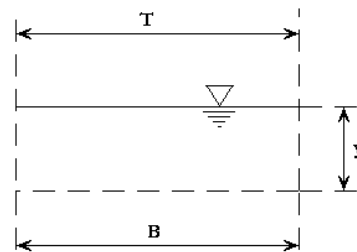


Figura 4.1.f. Sección tipo 5

Los seis tipos de sección representan lo que se describe a continuación.

- a) Tipo 0. Es la misma que para un canal de sección sencilla y corresponde a la subsección más profunda cuando el nivel del agua no inunda aún ninguna berma lateral.
- b) Tipo 1. Es la que ocurre en una llanura de inundación del lado derecho mientras el nivel del flujo no llega a la altura de la siguiente berma del mismo lado.
- c) Tipo 2. Es la misma que la anterior, pero del lado izquierdo.
- d) Tipo 3. Se presenta en el canal más profundo cuando la primera berma del lado izquierdo está a una altura menor que la primera del lado derecho y el flujo de agua inunda solamente a la del lado izquierdo.
- e) Tipo 4. Es igual a la anterior, pero la altura de la primera berma derecha es menor que la izquierda.
- f) Tipo 5. Es la que se presenta en cualquiera de las subsecciones cuando el nivel del agua ha rebasado la altura de las bermas.

Es así que para obtener las características geométricas de una subsección, primero se calcula si el nivel del agua rebasa o no la altura de las bermas, y después se determinan las subsecciones tipo en que se puede dividir para, finalmente, sumar los valores. De esta manera, para el canal central se puede presentar la suma de las subsecciones tipo 0, 3 y 5; 0, 4 y 5 ó 0 y 5. Para una subsección de berma derecha, la adición de las 1 y 5, ó sólo la 1. Si se trata de una llanura del lado izquierdo, 2 y 5, ó exclusivamente la 2. Las ecuaciones de cálculo se muestran en las tablas 4.2 y 4.3.

Tabla 4.2. Cálculo del área, ancho de superficie libre y  $dT/dy$  para cada tipo de subsección

Subsección	Área	Ancho de superficie libre	$\frac{dT}{dy}$
0	$A = By + \frac{k_d + k_i}{2} y^2$	$T = B + (k_d + k_i) y$	$\frac{dT}{dy} = k_d + k_i$
1	$A = By + \frac{k_d}{2} y^2$	$T = B + k_d y$	$\frac{dT}{dy} = k_d$
2	$A = By + \frac{k_i}{2} y^2$	$T = B + k_i y$	$\frac{dT}{dy} = k_i$
3	$A = By + \frac{k_d}{2} y^2$	$T = B + k_d y$	$\frac{dT}{dy} = k_d$
4	$A = By + \frac{k_i}{2} y^2$	$T = B + k_i y$	$\frac{dT}{dy} = k_i$
5	$A = By$	$T = B$	$\frac{dT}{dy} = 0$

Tabla 4.3. Cálculo del perímetro mojado y  $dP/dy$  para cada tipo de subsección

Subsección	Perímetro	$\frac{dP}{dy}$
0	$P = B + (\sqrt{k_d^2 + 1} + \sqrt{k_i^2 + 1}) y$	$\frac{dP}{dy} = \sqrt{k_d^2 + 1} + \sqrt{k_i^2 + 1}$
1	$P = B + \sqrt{k_d^2 + 1} y$	$\frac{dP}{dy} = \sqrt{k_d^2 + 1}$
2	$P = B + \sqrt{k_i^2 + 1} y$	$\frac{dP}{dy} = \sqrt{k_i^2 + 1}$
3	$P = \sqrt{k_d^2 + 1} y$	$\frac{dP}{dy} = \sqrt{k_d^2 + 1}$
4	$P = \sqrt{k_i^2 + 1} y$	$\frac{dP}{dy} = \sqrt{k_i^2 + 1}$
5	$P = 0$	$\frac{dP}{dy} = 0$

Por otro lado, es conveniente mencionar que para el proceso de cálculo en el programa el número de tirantes críticos posibles es igual a la cantidad de inundaciones que suceden conforme el nivel del agua va aumentando. Esto es, para una sección compuesta con dos llanuras de inundación a la misma altura, el número de tirantes críticos posibles es de dos; sin embargo si la berma derecha tiene una altura diferente a la de la berma izquierda, la cantidad de tirantes críticos posibles es de tres. Con fines explicativos, al número de tirantes críticos

posibles en un canal se le denominará número de procesos; y a la distancia entre llanuras de inundación, rango del proceso.

### IV.3 Representación gráfica de puntos

Una vez obtenidos los tirantes críticos posibles en el canal, el sistema incluye una opción para ver la gráfica Energía específica vs. Tirante ( $E-y$ ). En el proceso de graficación se debe considerar que para un canal, la función  $E = f(y)$  es una familia de curvas que se desplaza a la derecha o a la izquierda de acuerdo con el valor del gasto. Además, se ha demostrado que para una sección sencilla la función tiene al menos dos asíntotas; una es una recta horizontal, ya que cuando el tirante tiende a cero, la energía tiende a infinito ( $y \rightarrow 0, E \rightarrow \infty$ ); y la otra es una recta a  $45^\circ$ , es decir, que si el tirante tiende a infinito, la energía tiende a ser igual al tirante ( $y \rightarrow \infty, E \rightarrow y$ ). Como se mencionó en el capítulo III, para una sección compuesta, la gráfica presenta un punto de quiebre cuando el tirante es igual al nivel de la llanura de inundación, y a partir de ese punto, la curva tiende nuevamente a la asíntota de  $45^\circ$ .

Al considerar lo anterior, no se pueden establecer el dominio ni el contradominio para la función  $E = f(y)$  tomando en cuenta exclusivamente la geometría del canal. Por esta razón se decidió, antes de graficar, definir un dominio inicial que abarca desde el valor de tirante cero,  $y = 0$  hasta la altura de la última berma de inundación y para esa subsección el valor más grande entre el valor mayor de todos los rangos de proceso o dos veces la diferencia entre el tirante crítico de esa subsección y la altura de su berma. Además, el número de puntos dentro de cada rango se determina al dividir dicha distancia en 10 segmentos iguales o en secciones de 5 cm, lo que resulte en la mayor cantidad de puntos. A continuación, se calcula el valor de Energía específica para cada tirante. Con esto se asegura abarcar toda la sección del canal independientemente del valor del gasto.

Sin embargo, si se quisiera graficar todos estos puntos, la escala no permitiría observar el fenómeno, debido a los valores que alcanza la Energía específica. Para determinar los puntos que se graficarán hay que eliminar los que se alejan demasiado del promedio de valores de energía específica para poder apreciar el fenómeno adecuadamente. Es por esto que se determinó el proceso que se describe a continuación.

1. Se obtiene el promedio y la desviación estándar de la Energía específica para todos los puntos determinados.
2. Se eliminan todos los puntos cuyos valores de Energía específica son mayores al promedio y se compara la desviación con el promedio, si el promedio es mayor a la desviación, se procede al paso 5; en caso contrario se continúa con el paso 3.
3. Se obtiene el promedio y la desviación estándar para los puntos que quedan. Si la desviación es menor de uno, se continúa con el paso 5.
4. Se eliminan todos los puntos de valor de energía mayores al promedio. Si el promedio es menor a la desviación, se regresa al paso 3.
5. Se obtiene el valor menor entre el tirante crítico en el canal central, si existe, y la altura de la primera berma de inundación.

6. Si la altura encontrada en el punto 5 es menor que el límite inferior obtenido en la depuración de los puntos, se modifica dicho límite por valor obtenido en el punto 5 menos tres puntos hacia abajo. Con esto aseguramos que se visualizarán todos los tirantes críticos y las bermas para valores pequeños del tirante.
7. Se obtiene el valor mayor entre el tirante crítico en la berma más alta el canal central, si existe, y la altura de dicha berma de inundación.
8. Si el valor encontrado en el punto 7 es mayor que el límite superior, se modifica dicho límite por el valor obtenido más tres puntos hacia arriba. Con esto aseguramos que se visualizarán todos los tirantes críticos y las bermas para valores grandes del tirante.
9. Con el grupo de puntos que resulta de este proceso, se continúa con la graficación de los puntos.

Este procedimiento se propone como una solución racional para determinar el dominio y contradominio de una función que se desea graficar.

## V APLICACIONES NUMÉRICAS

### V.1 Dos casos reales

Con el objetivo de presentar los resultados obtenidos con ambos métodos, se revisó el diseño para dos canales reales, limitándose a las condiciones de flujo normal y de las condiciones críticas, para la sección tipo. Los datos fueron proporcionados por la Comisión Nacional del Agua y el Instituto de Ingeniería de la UNAM. El proceso constructivo que se describe corresponde a un canal recubierto de concreto.

#### V.1.1 Río Tijuana

El canal construido en el río Tijuana se localiza aguas abajo de la presa Abelardo L. Rodríguez, se encuentra cerca de la ciudad del mismo nombre y tiene el objetivo de protegerla en el caso de una avenida. En condiciones normales el canal se usa para transportar aguas residuales de la ciudad y el tirante observado no llega a desbordar las llanuras.

El proyecto de la canalización consiste en tres etapas, la I consta de 4.6 km y termina en la frontera con Estados Unidos; la II es de 5.6 km de largo y recibe un afluente del río Álamos; y la III tiene una longitud de 6.8 km y que inicia en la base de la cortina de la presa. Las etapas II y III tienen un gasto de diseño de  $2\ 100\ \text{m}^3/\text{s}$  y una pendiente de 0.0035 y 0.00204 respectivamente, mientras que la etapa III, cuya sección transversal se muestra en la figura 5.1, un caudal de  $3\ 620\ \text{m}^3/\text{s}$  y una pendiente de 0.00204. Todo el canal está recubierto de concreto, por lo que el coeficiente de Manning se considera de 0.015 para la subsección más profunda y de 0.016 para las bermas.

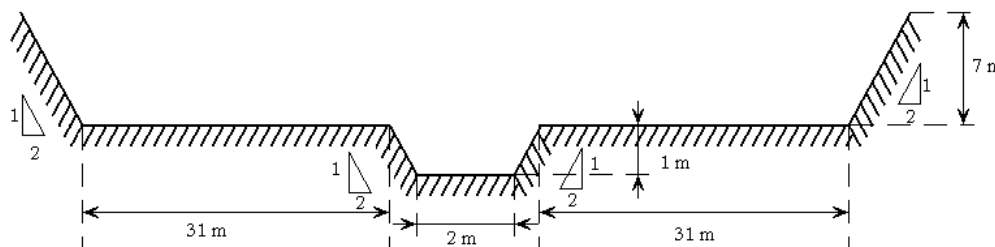


Figura 5.1 Sección transversal del río Tijuana en su etapa I. En las etapas II y III el ancho de plantilla de las bermas es de 24 m

#### V.1.2 Arroyo sin nombre en Cabo San Lucas

El segundo canal se ubica en el Rancho Paraíso, a 500 m al norte del km 7 de la carretera Cabo San Lucas – San José del Cabo y es una alcantarilla recubierto de concreto que se diseñó para transportar el caudal de una avenida de  $8.3\ \text{m}^3/\text{s}$ , con una pendiente de 0.00045 y un bordo libre de 0.70 m. La sección transversal se muestra en la figura 5.2. Los coeficientes de Manning considerados para esta sección son los mismos que para el río Tijuana.

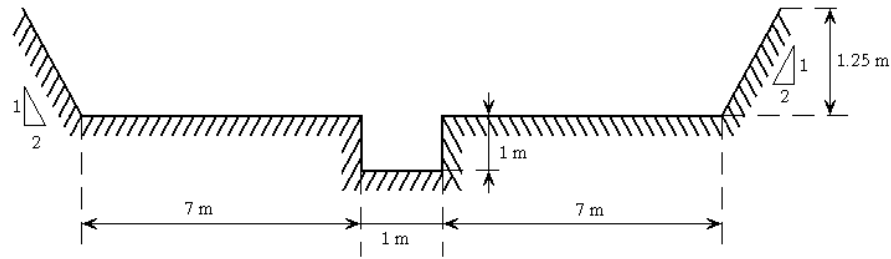


Figura 5.2. Sección transversal del arroyo sin nombre en Cabo San Lucas

## V.2 Revisión del diseño

El primer paso de la revisión es verificar que para las condiciones de flujo normal las dimensiones del canal tengan la capacidad de transportar el caudal de diseño, además, que los tirantes críticos posibles para ese gasto se encuentren alejados del tirante normal. El coeficiente de Manning para el canal principal se consideró de 0.015 y para las bermas de 0.016; además, se utilizó el factor de 1.19 para el coeficiente de Manning del canal principal cuando el canal trabaja como sección compuesta. Con fines de comparación de resultados, para régimen crítico se supusieron los coeficientes  $\alpha$  y  $\beta$  iguales a la unidad. Finalmente, se determinó la velocidad media en cada caso para confirmar que se respeten las velocidades mínima y máxima para el tipo de recubrimiento. Los resultados se presentan en las tablas 5.1 y 5.2.

Tabla 5.1 Resultados de la revisión del flujo normal.

Canal	Tirante normal, m	Bordo libre*, m	Altura de taludes, m	Velocidad media, m/s
Río Tijuana Etapa I	6.656	1.964	8.620	7.998
Río Tijuana Etapas II y III	5.665	1.716	7.381	7.014
Arroyo sin nombre	1.556	0.689	2.245	0.834

Tabla 5.2 Resultados de la revisión de las condiciones críticas.

Canal	Chaudhry	Blalock	Chaudhry	Blalock	Gasto máximo m <sup>3</sup> /s
	Tirante crítico, m		Gasto mínimo m <sup>3</sup> /s		
Río Tijuana Etapa I	7.155	7.155	6.061	5.773	10.229
Río Tijuana Etapas II y III	5.965	5.966	6.425	6.195	10.229
Arroyo sin nombre	1.256	1.261	1.388	1.341	3.132

Para el río Tijuana en sus etapas II y III se observa que la altura de los taludes es satisfactoria para conducir los gastos de diseño en condiciones de flujo uniforme sin ningún problema; por el contrario, en la etapa I se tiene que la altura necesaria sobrepasa a la existente, lo que implica que cuando el canal se encuentre trabajando con el gasto de diseño, el oleaje del flujo puede desbordar los taludes. Por otro lado, para el arroyo sin nombre esta altura apenas se cumple.

\* Nota: El bordo libre se obtiene de la ecuación empírica  $L_b = 0.30 + 0.25 y$ .

En conclusión, se recomienda aumentar la altura de los taludes del río Tijuana en su etapa I para evitar desbordamientos.

Con relación a la velocidad mínima, es de 0.60 m/s para canales pequeños y de 0.90 m/s para canales grandes, por lo que para los tres casos la velocidad media es mayor a este valor y se logra evitar el crecimiento de vegetación y la acumulación de sedimentos. Por otro lado, la velocidad máxima permisible para concreto bien acabado se recomienda no sea mayor de 25 m/s; de igual manera se cumple este requisito.

En el río Tijuana se puede observar que el flujo normal está en el rango de régimen supercrítico ya que el tirante normal es menor que el crítico; además, el gasto de diseño es mucho mayor que el límite superior para ocurrencia de tirantes críticos múltiples. En el arroyo sin nombre, el flujo normal se desarrolla en régimen subcrítico pero el tirante normal se encuentra muy cerca del crítico. Por otro lado, la diferencia entre los criterios de Blalock y Chaudhry es despreciable y se confirma lo mencionado en el capítulo IV referente a la relación  $n_1/n_2$ , que para este caso es de 1.

Para finalizar la revisión del diseño, sería necesario obtener los perfiles de flujo a lo largo de todo el canal, lo cual queda fuera del objetivo del presente trabajo.

### **V.3 Proceso constructivo**

Para la construcción del canal hay que tomar en cuenta que si se trata de un río, la mejor época del año para efectuar los trabajos es la temporada de sequía. Además, se considera el trazo del canal para determinar los movimientos de tierra, así como los bancos de material, tanto para el cuerpo del canal como para los agregados que se usarán en la fabricación del concreto. Es importante también localizar bancos de nivel para el trazo en campo del canal.

La primera fase consiste en efectuar el despalme para remover el material fino que recubre las capas de suelo más resistente, sean arenas o gravas, incluyendo hasta 15 cm de dicha capa. Previo al tendido de la primera capa de terracerías se pasa el equipo de compactación para mejorar la compacidad del terreno de desplante, el equipo se selecciona conforme con las características del material. Se realizan los movimientos de tierra necesarios sean de corte o de relleno de acuerdo con el trazo y la nivelación del canal para lograr la pendiente de diseño. El material de relleno debe colocarse en capas, del espesor determinado en los estudios de mecánica de suelos, para lograr la compactación y resistencias deseada. La maquinaria necesaria para llevar a cabo este proceso son tractores, motoniveladoras, motoescrepas, pipas, compactadores vibratorios lisos o de pata de cabra.

Una vez tendido y compactado el material de soporte, se procede a colocar la cimbra para el colado de las losas de concreto de recubrimiento que puede ser o no reforzado, según las dimensiones del canal y las solicitaciones a las que se someta. De acuerdo con la importancia del canal se puede colar además una losa en la corona de los terraplenes para proteger el suelo que compone los taludes laterales en el caso de un desbordamiento. El colado se hace en tramos pequeños alternos de tal forma que las losas ya coladas sirven de cimbra a la siguiente.



Hay que recordar que la reacción química entre el cemento y el agua es exotérmica lo que puede provocar evaporación del agua y la consecuente reducción de la relación agua-cemento que es función directa de la resistencia del concreto, además de grietas. Para un canal, no se puede permitir grietas en las losas de concreto, pues constituirían áreas de fácil erosión que reducirían la vida útil de la estructura. Debido a lo anterior, se debe de tomar en cuenta el clima, ya que en zonas con temperaturas muy altas se puede, además de regar con agua, colocar mangueras ahogadas en el colado y hacer circular agua a través de ellas para disminuir la temperatura del concreto. Dependiendo del tiempo disponible para la construcción de la obra, se pueden utilizar también catalizadores para acelerar el fraguado del concreto.

Finalmente, se hace una limpieza del canal antes de ponerlo en funcionamiento.

## VI CONCLUSIONES

Las pruebas experimentales nos permiten reconocer que el régimen crítico para canales de sección compuesta tiene un comportamiento muy distinto al presente en los canales de sección sencilla. Esto se debe a que no se cumple con la hipótesis de flujo unidimensional en el momento que el flujo del agua inunda las bermas laterales.

Debido a lo anterior, el flujo uniforme en un canal de sección compuesta se presenta muy inestable cuando el nivel del flujo apenas rebasa la altura de las llanuras de inundación. Al establecerse el flujo existe una transferencia de momentum en las intercaras idealizadas, la cual no puede ser despreciable. Sturm propuso considerar que el coeficiente de Manning se incrementa en 19 % para la subsección más profunda cuando el nivel del agua se encuentra sobre la altura de las bermas de inundación; sin embargo, se recomienda profundizar en las investigaciones al respecto para describir mejor este fenómeno.

El método propuesto por Blalock y Sturm es general para cualquier sección compuesta, a diferencia del propuesto por Chaudhry que sólo contempla secciones simétricas con coeficientes  $\alpha = 1$  y  $\beta = 1$  para cada subsección; sin embargo los resultados obtenidos con ambos métodos son muy similares cuando los coeficientes de Manning en el canal central y las subsecciones son muy similares.

Los resultados obtenidos en la comparación de los criterios de energía específica mínima y momentum mínimo permiten establecer que, mientras los cocientes  $n_1/n_2$  y  $b_1/b_2$  se encuentren en el rango entre 0.5 y 1, los tirantes críticos determinados con cualquiera de estos criterios son muy similares.

El rango de caudales en los que se presentan tirantes críticos múltiples en una sección compuesta es mayor para el criterio de energía específica mínima, al ser necesario ubicar en la práctica los límites superior e inferior de dicho rango, se recomienda contemplar el caudal mínimo obtenido por el método de Blalock-Sturm con fines de diseño.

## **ANEXO I MANUAL DEL SISTEMA**

En este anexo se incluye el manual de instalación y operación del sistema desarrollado para obtener las condiciones críticas para cualquier canal de sección compuesta.

**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**



**FACULTAD DE INGENIERÍA**

**PROGRAMA DE CÁLCULO DE TIRANTES CRÍTICOS EN  
CANALES CON SECCIÓN COMPUESTA**

Francisco Javier Lovera Salazar

## ÍNDICE

1	Requerimientos mínimos del sistema	3
2	Instalación	3
3	Iniciar el programa	5
4	Configuración	6
5	Crear una sección	8
6	Abrir una sección	8
7	Agregar o modificar subsecciones	9
8	Guardar una sección	11
9	Ver una sección	12
10	Cerrar una sección	12
11	Iniciar el cálculo	13
12	Efecto de interacción en las intercaras	14
13	Obtención de resultados	15

# 1 REQUERIMIENTOS MÍNIMOS DEL SISTEMA

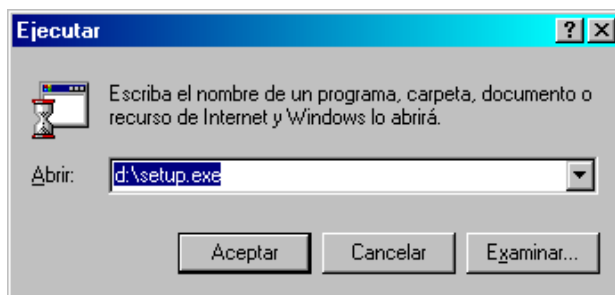
Para funcionar correctamente este sistema necesita un equipo PC con las siguientes características mínimas:

- Pentium 300 Mhz.
- Windows 98.
- 10 MB de espacio en disco duro.
- 64 MB de memoria RAM.
- Monitor VGA con resolución 800 x 600 píxeles.

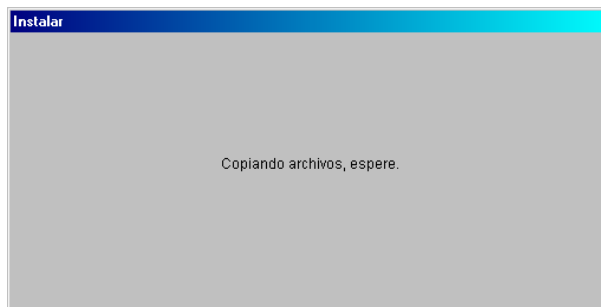
# 2 INSTALACIÓN

Antes de iniciar la instalación del sistema, cierre todos los programas en ejecución para evitar errores. Esta etapa tiene una duración de 5 a 15 minutos, dependiendo del equipo en que se instale.

Introduzca el CD-ROM o el primer disco flexible de 3 1/2 “ en la unidad correspondiente. Haga clic en *Inicio*, seleccione la opción *Ejecutar...*, y en la ventana que aparece escriba *D:\setup.exe*. Haga clic en *Aceptar*.\*



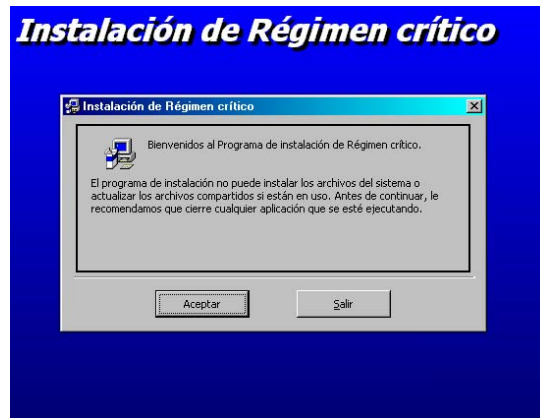
Se inicia el programa de instalación desplegando una ventana en la que se informa de la copia de algunos archivos necesarios.



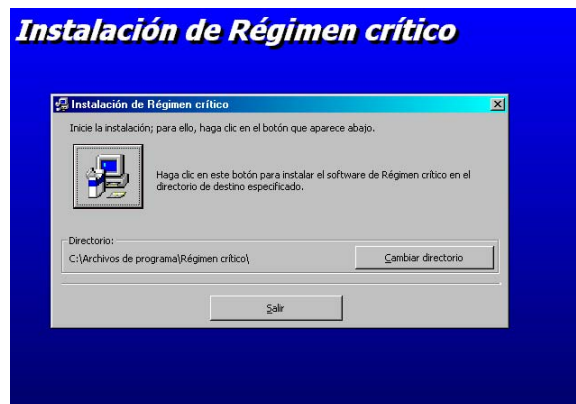
---

\* Nota: Los archivos y el programa de instalación han sido examinados para verificar que no contengan ningún virus. Sin embargo, es recomendable que antes de continuar con el proceso de instalación, utilice su software de protección antivirus para evitar una infección.

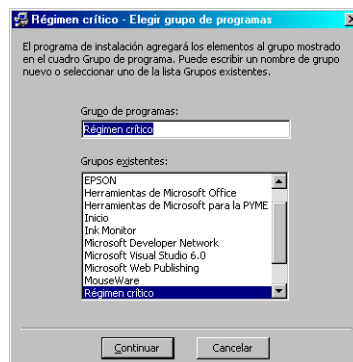
A continuación su pantalla presentará un cuadro de dialogo similar al presentado en la página siguiente, en el que se vuelve a pedir cerrar todos los programas en ejecución:



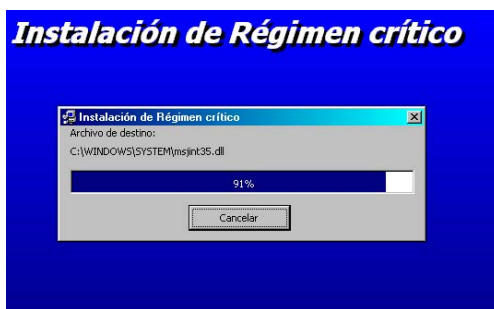
Haga clic en el botón *Aceptar* para continuar con la instalación. La ventana que aparece a continuación le indica la carpeta en la que se instalará el sistema. Se recomienda no modificar esta información. Para proceder, haga clic en el botón con el icono de instalación que se encuentra en la parte izquierda de la ventana.



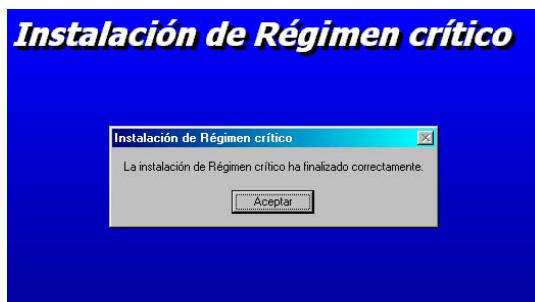
El sistema crea una carpeta de grupo de programas donde se instalarán los accesos directos. En la ventana activa se indica el grupo *Régimen crítico*, se recomienda no modificarlo. Haga clic en el botón *Continuar*.



Se inicia la descompresión y copia de los archivos del CD-ROM o discos flexibles hacia el disco duro. Se visualiza una pantalla similar a esta\*:





Al finalizar el proceso se verá el mensaje siguiente, indicando la instalación exitosa.



### 3 INICIAR EL PROGRAMA

Al instalar el sistema en el equipo, se crea un acceso directo en la carpeta de programas de su PC. Para iniciar el sistema siga los pasos que se describen a continuación:

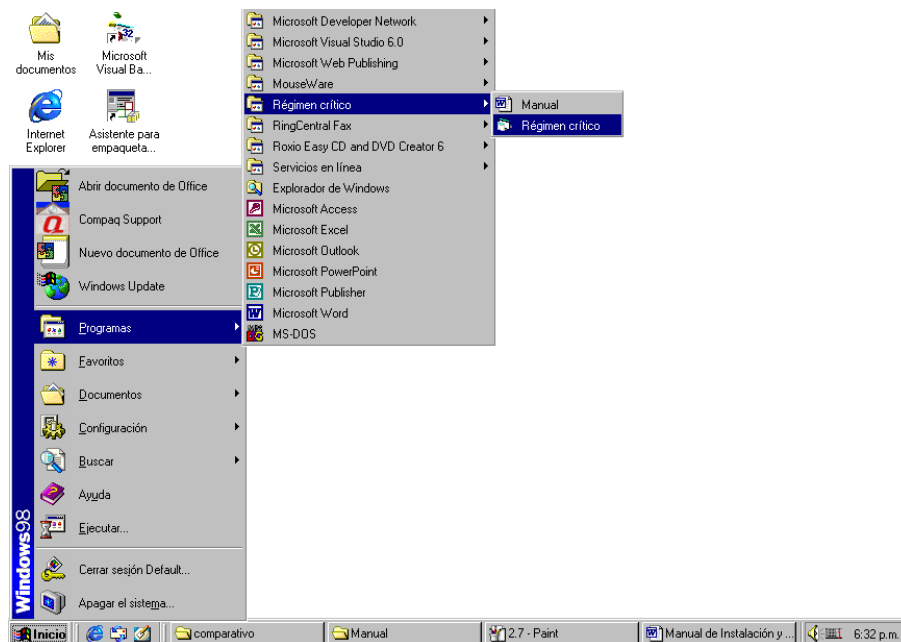
- 3.1 Haga clic en el botón *Inicio*, seleccione el icono de *Programas*, después el icono *Régimen crítico*.
- 3.2 Se despliega un menú en el que aparecen dos accesos directos. El primero, *Manual*,  corresponde al presente manual en formato Word 2000\*\*; el segundo, *Régimen crítico*,  al programa. Para iniciar el sistema haga clic en el segundo icono.

---

\*Nota: Al copiar los controladores de este sistema, el proceso de instalación puede detectar que los archivos existentes en la PC son más recientes, por lo que se recomienda conservar los archivos y no sobrescribir. También es posible que se requiera reiniciar el equipo antes de usar el sistema.

\*\*Nota: Para poder abrir el Manual, necesita tener Office 2000 instalado en su PC.



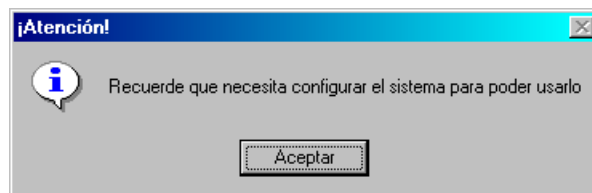


3.3 En este momento aparece la ventana del programa.\*

## 4 CONFIGURACIÓN

El proceso de configuración debe de realizarse antes de empezar a usar el programa. Este sistema usa un archivo en formato DBase III, como modelo de los archivos que se generan para cada ejemplo. Estos datos representan las características geométricas e hidráulicas propias de cada sección compuesta. Para iniciar la configuración se tienen que seguir los siguientes pasos:

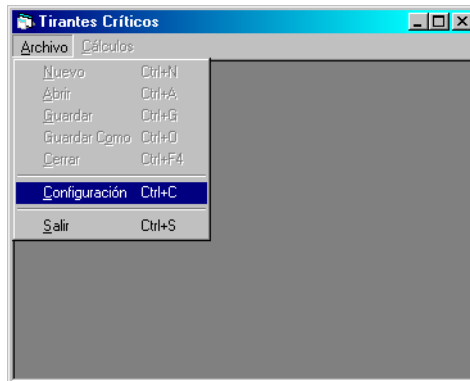
4.1 Arranque el programa. (Ver el apartado 3.) Si es la primera ocasión se verá este mensaje. En caso contrario vea directamente el paso 4.2.



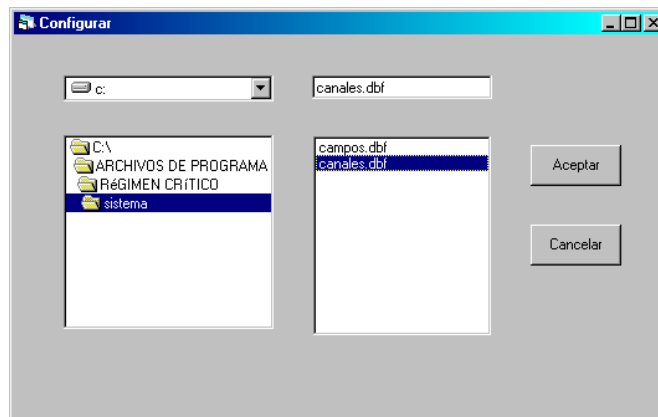
4.2 Al hacer clic en *Aceptar* se abre el programa. Para configurar el sistema hacer clic en el menú Archivo y la opción Configuración, o bien desde el teclado *Ctrl+C*.

---

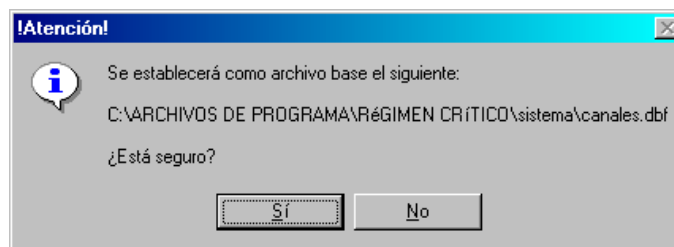
\* Nota: En el caso de ser la primera vez que se inicia el sistema, aparece una ventana que pide se configure el sistema. Ver apartado 4.



4.3 En la ventana que aparece, se debe seleccionar y abrir el archivo *canales.dbf*, que viene incluido en el sistema y se toma como modelo para trabajar. Haga clic en el botón *Aceptar*. La carpeta en la que se encuentra este archivo es ...*tirante\sistema\*



4.4 Para confirmar que el sistema ha sido configurado, aparece la siguiente ventana. Haga clic en el botón *Sí*.



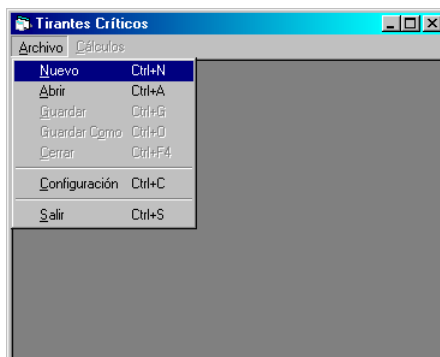
4.5 Aparece la ventana del programa. Ha finalizado la configuración.

Si trata de reconfigurar el sistema, aparecerá una ventana que le recordará que el sistema ya ha sido configurado previamente.

## 5 CREAR UNA SECCIÓN

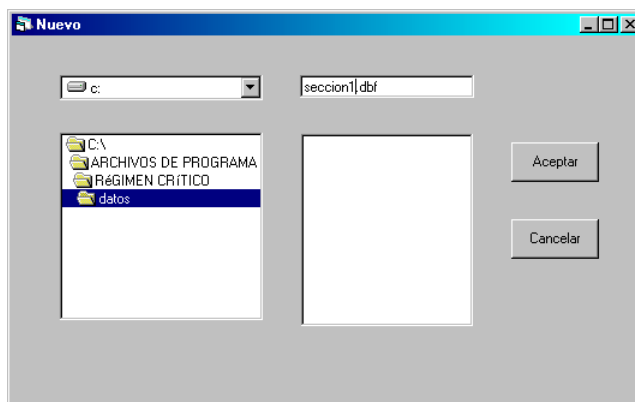
El sistema necesita un archivo con la información de una sección geométrica para poder obtener resultados sobre la misma. En el caso de empezar a usar el sistema, usted deberá de crear ese archivo, de acuerdo con los pasos siguientes:

- 5.1 Haga clic en el menú Archivo y la opción Nuevo, o bien desde el teclado *Ctrl+N*.



- 5.2 Aparece una ventana en la que se debe escribir el nombre del archivo en que se guardará la información geométrica de una sección dada. Este nombre no deberá exceder ocho caracteres y extensión *.dbf*. Haga clic en el botón *Aceptar*.

La carpeta en la que se guardan los archivos de datos es *...tirante\datos\*. Sin embargo, se pueden guardar los archivos en cualquier otra carpeta, sin mayor problema.

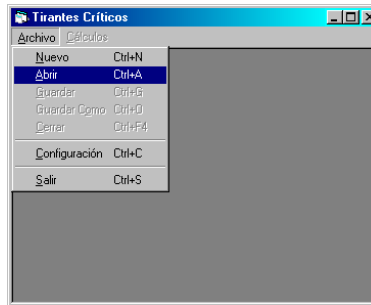


- 5.3 Una vez creado el archivo, se pueden agregar las subsecciones en que se divide la sección compuesta que se quiera analizar. (Ver apartado 7.)

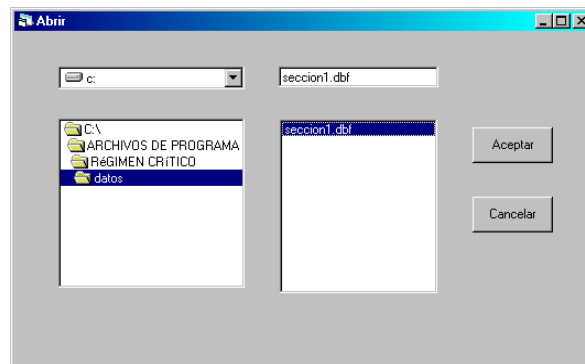
## 6 ABRIR UNA SECCIÓN

En el caso de que ya se tenga alguna sección creada para analizar, puede abrir el archivo respectivo, de acuerdo con los pasos siguientes:

- 6.1 Haga clic en el menú *Archivo* y la opción *Nuevo*, o bien desde el teclado *Ctrl+N*. Aparece una ventana en la que se debe de seleccionar el nombre del archivo que se desea abrir. Por descontado, la carpeta donde se buscan los archivos es *...tirante\datos\*; sin embargo, si se han guardado archivos en alguna otra carpeta, sólo es necesario buscar esa carpeta y abrir el archivo respectivo.

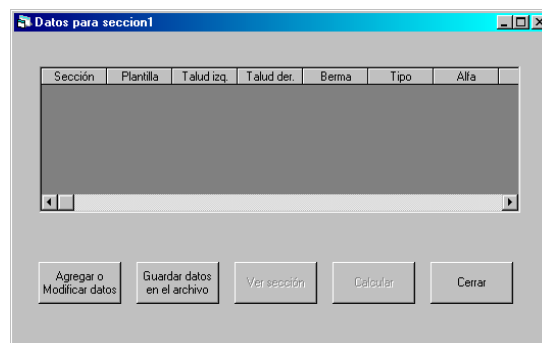


- 6.2 Una vez seleccionado el archivo haga clic en el botón *Aceptar*, y se puede trabajar con esta sección. Ya se pueden agregar o modificar las subsecciones en que se divide la sección compuesta que se quiera analizar. (Ver apartado 7.)



## 7 AGREGAR O MODIFICAR SUBSECCIONES

Una vez creado o abierto un archivo de datos de una sección, se aprecia una ventana similar a de la página siguiente:



7.1 Haga clic en el botón *Agregar o Modificar datos*, se mostrará la ventana siguiente.

7.2 En esta ventana se indican las propiedades geométricas e hidráulicas de cada una de las subsecciones en que se divide el ejemplo que se busca analizar. Cada elemento en la ventana corresponde a una propiedad:

- *Tipo de sección.* Aquí se puede seleccionar una de las tres opciones que aparecen en la lista; central, derecha o izquierda que indican el lado en que se encuentra la subsección en cuestión.
- *Ancho de plantilla.* De la subsección, puede tomar valores desde 0 y positivos, tiene dimensiones en metros.
- *Altura de berma.* Sólo existe para las secciones tipo derecha o izquierda. Indica la distancia desde la plantilla central hasta la plantilla de la subsección, puede tomar valores desde 0 y positivos, también tiene dimensiones en metros.
- *Talud izquierdo  $k_p$ .* Sólo existe para secciones tipo central o izquierda. Indica el talud del lado izquierdo de la subsección respectiva y es adimensional.
- *Talud derecho  $k_p$ .* Sólo existe para secciones tipo central o derecha. Indica el talud del lado derecha de la subsección respectiva y es adimensional.
- *n de Manning.* Es el coeficientes de rugosidad de Manning, tiene dimensiones en el Sistema Internacional.
- *Rugosidad equiv.* Indica la rugosidad equivalente de la subsección (ver teoría de rugosidad en flujos hidráulicos), tiene dimensiones en metros.
- *Coefficiente de coriolis ( $\alpha$ ).* Indica el coeficiente de corrección de la carga de velocidad para la subsección, es adimensional.
- *Coefficiente c.* Indica el coeficiente c de la ecuación de Nikuradse que varía de acuerdo con el comportamiento de la pared y forma de la sección.
- *Coefficiente alfa subn.* Indica el coeficiente  $\alpha_N$  de la ecuación de Nikuradse que varía de acuerdo con el comportamiento de la pared y forma de la sección.

Además, cada botón de la forma tiene las funciones siguientes:

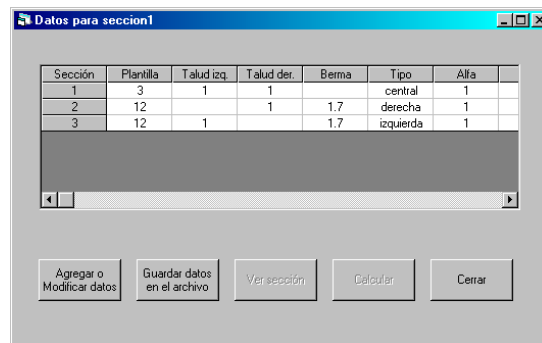
- *Guardar.* Guarda los datos escritos en la ventana para la subsección. Además, el botón *Cancelar* cambia de leyenda a *Cerrar*.
- *Cancelar.* Cancela los cambios de la subsección y cierra esta ventana sin guardar los datos escritos.

- *Salir.* Una vez guardados los datos para la subsección, este botón cierra la ventana.
- *Anterior.* La ventana muestra los datos para la subsección anterior a la que se aprecia en la ventana. Al mismo tiempo guarda los datos en pantalla.
- *Siguiente.* La ventana muestra los datos para la subsección siguiente a la que se aprecia en la ventana. Al mismo tiempo guarda los datos en pantalla. En caso de que no exista una subsección siguiente, crea una nueva para poder agregarla.
- *Eliminar.* Elimina la subsección y todos sus datos que se muestran en la ventana.

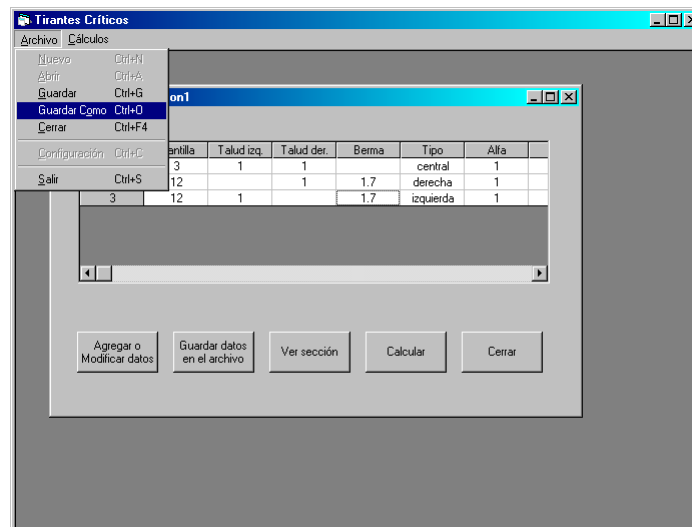
## 8 GUARDAR UNA SECCIÓN

Después de abrir el archivo de una sección, y agregado o modificado en sus subsecciones, se debe de guardar los datos de la sección para poder realizar los cálculos respectivos. Para esto, existen dos opciones:

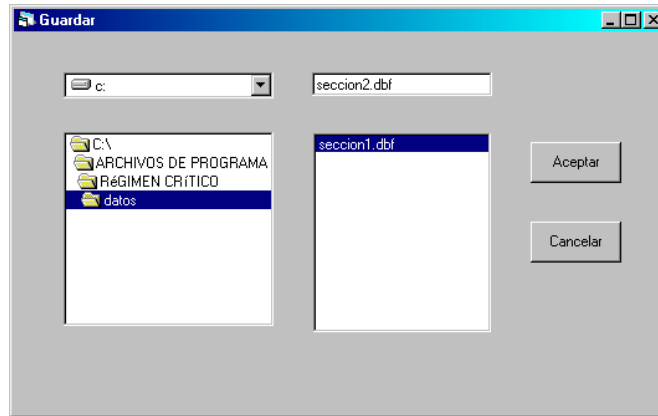
- Guarde los datos en el mismo archivo que se abrió, haciendo clic en el botón *Guardar datos en el archivo* o bien, seleccione en el menú principal la opción *Guardar*, o incluso desde el teclado *Ctrl.+G*. Ya se puede pasar a la etapa de cálculo (Ver apartado 10).



- 2 Guarde los datos en un archivo diferente al que se abrió originalmente. En este caso hay que seleccionar en el menú principal la opción *Guardar como* o desde el teclado *Ctrl.+O*.

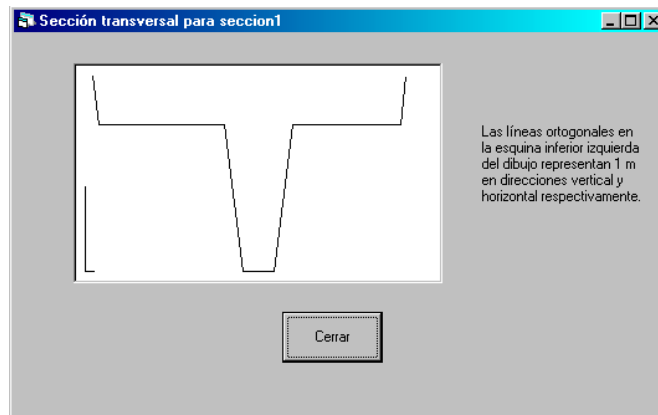


Aparecerá una ventana como la siguiente, en la que se debe de seleccionar la carpeta y nombre de archivo que se desee; para guardar estos datos haga clic en el botón *Aceptar*.



## 9 VER UNA SECCIÓN

Para ver la sección transversal de los datos almacenados, se necesita hacer clic en el botón *Ver sección*. Aparece una ventana como la siguiente.

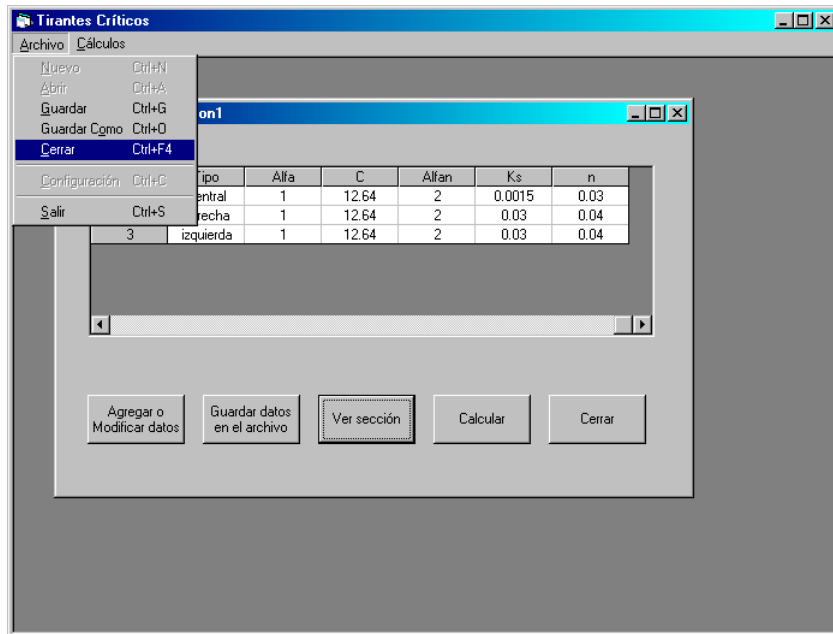


La ventana muestra la sección transversal del canal a analizar, además de dos líneas ortogonales que representan 1 m en direcciones vertical y horizontal. Para regresar a la ventana anterior, basta con hacer clic en el botón *Cerrar*.

## 10 CERRAR UNA SECCIÓN

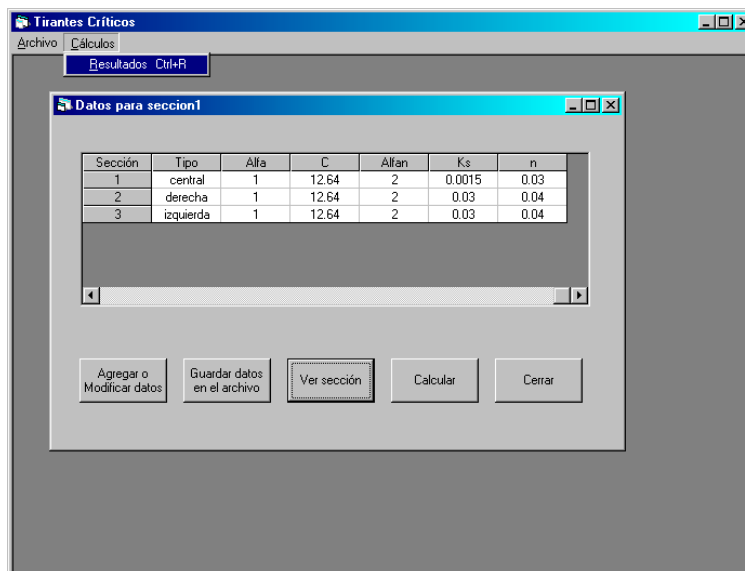
Para dar por terminado el proceso de cálculo, o bien para poder trabajar con otra sección se debe cerrar la sección, de acuerdo con los pasos siguientes:

- 10.1 Haga clic en el menú *Archivo* y la opción *Cerrar* o desde el teclado *Ctrl.+F4*. También basta con hacer clic en el botón *Cerrar*.



## 11 INICIAR EL CÁLCULO

Una vez guardados los datos de la sección a analizar, se puede proceder a iniciar el cálculo de tirantes críticos. Para esto, es necesario seleccionar del menú principal la opción Cálculos y luego Resultados o bien, *Ctrl.+R*; otro camino es hacer clic en el botón *Calcular*.

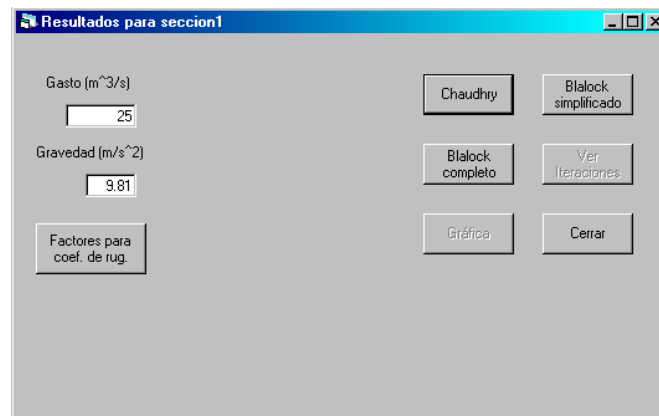


Aparece una ventana en la que se alimentarán las condiciones hidráulicas bajo las que se analizará la sección, se obtendrán los resultados, se visualizarán las iteraciones necesarias para llegar a esos resultados y una gráfica de Energía específica contra Tirante.



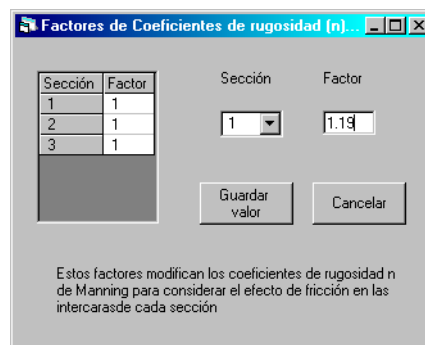
- 11.1 En la caja de texto *Gasto* debe proporcionar el gasto para el cual se desea obtener los resultados, tiene dimensiones de metros cúbicos por segundo.
- 11.2 En la caja de texto *Gravedad* se determina la gravedad con la que se desea obtener los resultados, por descomposición se supone como 9.81, tiene dimensiones de metros sobre segundo cuadrado.

Para salir de esta ventana, basta con hacer clic en el botón *Cerrar*.



## 12 EFECTO DE INTERACCIÓN EN LAS INTERCARAS

Para considerar el efecto de la interacción del flujo en las intercaras de las subsecciones se puede modificar el coeficiente n de Manning de rugosidad. Para esto, haga clic en el botón *Factores para coef. de rug.*, el cual presenta la ventana siguiente.



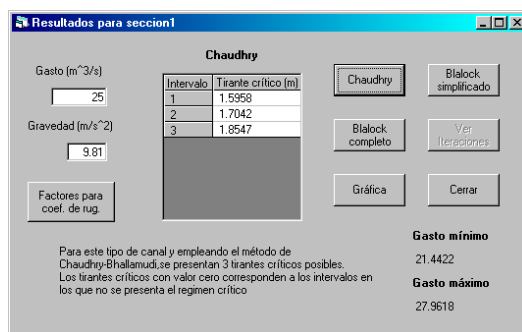
- 12.1 La tabla del lado izquierdo de la ventana presenta los valores de todos los factores para cada subsección. Al seleccionar cada una de las subsecciones en el combo *Sección*, la caja de texto *Factor* presenta el valor respectivo. Este valor se puede modificar y para guardar el valor hay que hacer clic en el botón *Guardar valor*. Con esto último, se puede ver que el valor en la tabla cambia. El botón *Cancelar* cambia a *Cerrar*.
- 12.2 Al hacer clic en el botón *Cancelar* se descarta el valor que se haya escrito en la caja de texto *Factor* y cambia el nombre de este botón por el de *Cerrar*.
- 12.3 Al hacer clic en el botón *Cerrar* se cierra esta ventana y se regresa al proceso de cálculo.

## 13 OBTENCIÓN DE RESULTADOS

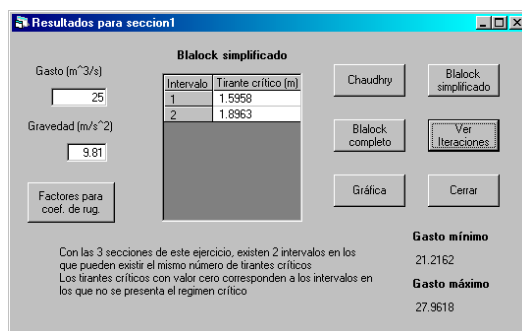
Una vez que se han proporcionado los valores de gasto y gravedad, además de los factores del coeficiente  $n$  de rugosidad de Manning, se puede obtener lo siguiente:

13.1 *Los tirantes críticos.* De acuerdo con los datos proporcionados, el programa puede obtener los tirantes en los que se presenta el régimen crítico. Existen tres opciones:

- Método de Chaudhry.** Si la sección del canal compuesto, es simétrica con dos llanuras de inundación, y coeficiente de coriolis igual a uno en cada subsección, el sistema puede obtener los tirantes críticos usando el método propuesto por Chaudhry. Al hacer clic en el botón *Chaudhry*, se visualiza una tabla que presenta estos resultados. Además, se obtienen los gastos mínimo y máximo que delimitan el rango de caudales donde se presentan tirantes críticos múltiples.

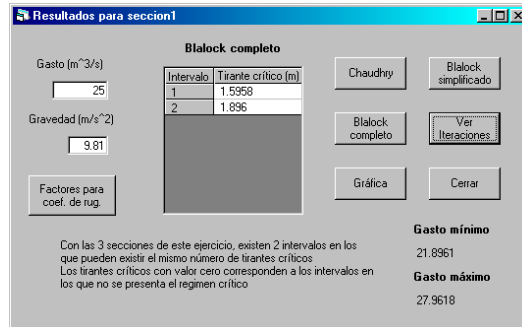


- Método de Blalock simplificado.** Para cualquier tipo de sección compuesta el sistema puede obtener los tirantes críticos usando el método propuesto por Blalock. Al hacer clic en el botón *Blalock simplificado*, el programa obtiene los tirantes críticos sin considerar la variación del coeficiente  $n$  de Manning con respecto al tirante. Se visualiza una tabla que presenta estos resultados. Adicionalmente, si se trata de una sección simétrica con una llanura de inundación a cada lado, se presentan los límites superior e inferior para los que se da la ocurrencia de tirantes críticos múltiples.

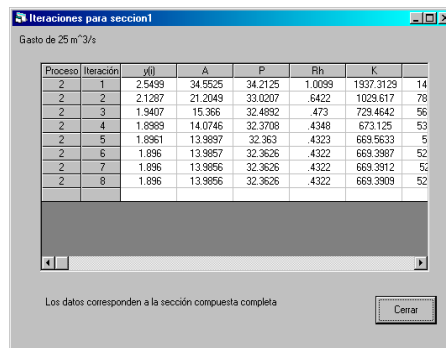


- Método de Blalock completo.** Para cualquier tipo de sección compuesta el sistema puede obtener los tirantes críticos usando el método propuesto por Blalock. Al hacer clic en el botón *Blalock completo*, el programa obtiene los tirantes críticos considerando la

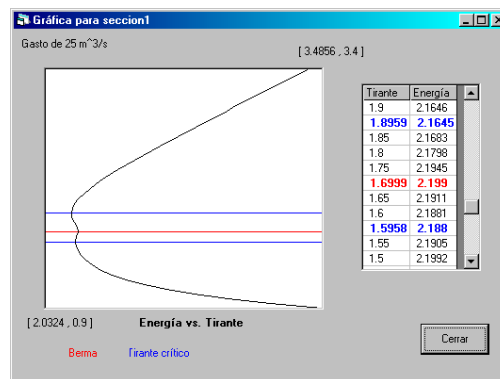
variación del coeficiente  $n$  de Manning con respecto al tirante. Se visualiza una tabla que presenta estos resultados. Los caudales mínimo y máximo se obtienen bajo las mismas condiciones mencionadas en el inciso anterior.



13.2 *Las iteraciones.* Una vez obtenidos los tirantes críticos por el método de Blalock, al hacer clic en el botón *Ver Iteraciones*, el sistema presenta una ventana con una tabla que contiene los valores del proceso de cálculo que llevaron a la obtención de los resultados. Esto no se puede presentar para el caso del método de Chaudhry. Para regresar a la ventana anterior basta con hacer clic en el botón *Cerrar*.



13.3 *La gráfica.* Al hacer clic en el botón *Gráfica*, se muestra una ventana con una gráfica que representa los valores de Energía específica contra Tirante, ambos con dimensiones en metros. Esta gráfica sí se puede generar para el caso del método de Chaudhry. Para regresar a la ventana anterior basta con hacer clic en el botón *Cerrar*.



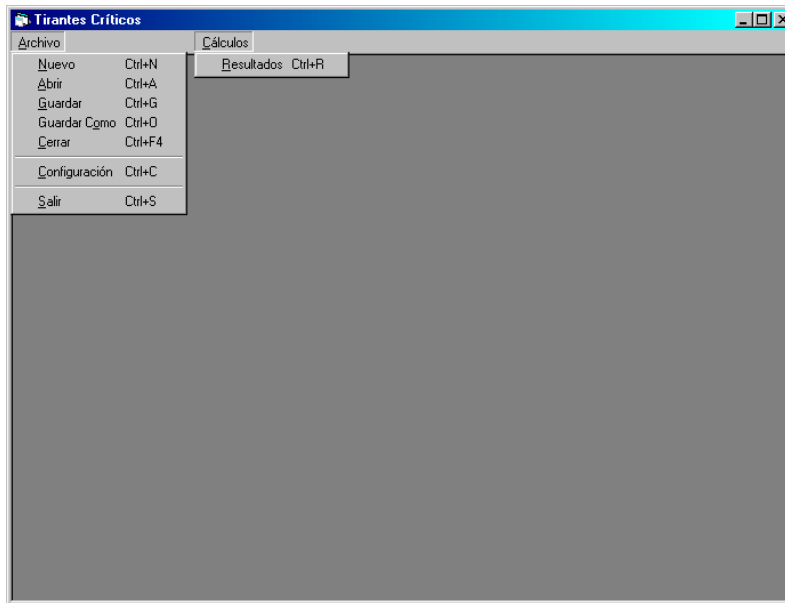
En esta ventana se pueden ver los siguientes elementos:

- *Tabla.* Presenta los valores que se emplearon para dibujar la gráfica. Los valores de tirante crítico y de altura de berma se presentan en negritas y en colores rojo y azul respectivamente.
- *Gráfica.* Presenta los valores de energía específica y tirante para el ejemplo analizado. Los niveles de tirantes críticos y de bermas se presentan con los mismos colores mencionados previamente. Además, si es el caso se presenta la línea de energía específica cero en color morado.

## ANEXO II PANTALLAS DEL PROGRAMA

En este anexo se incluyen todas las pantallas del sistema con una lista de los objetos que contienen.

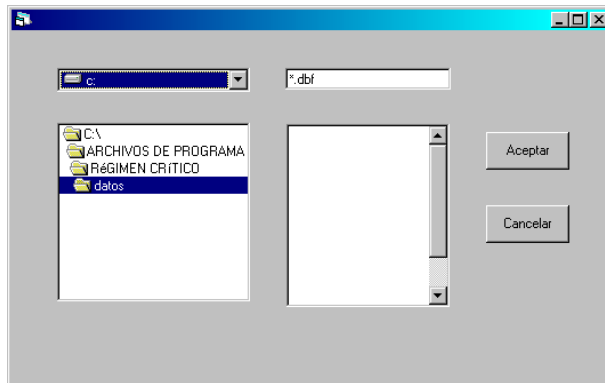
### MDIPrincipal



Es la pantalla principal del sistema y contiene los menús siguientes.

- Archivo. MnuArchivo.
- Nuevo. MnuNuevo.
- Abrir. Mnuabrir.
- Guardar. MnuGuardar.
- Guardar como. MnuGuardarComo
- Cerrar. MnuCerrar
- Configuración. MnuConfigura.
- Salir. MnuSalir.
- Cálculos. MnuCalculo.
- Resultados. MnuResultados.

### FrmArchivo



Esta pantalla tiene los elementos siguientes.

- CmdAceptar. Botón para aceptar.
- CmdCancelar. Botón para cancelar.
- Dir1. Control para mostrar y seleccionar los subdirectorios de una unidad de disco directorio.
- Drive1. Control para mostrar y seleccionar las unidades de disco en la computadora.
- File1. Control para mostrar y seleccionar los archivos de un directorio.
- TxtArchivo. Caja de texto para escribir el nombre del archivo a trabajar.

## FrmDatos



Esta pantalla tiene los elementos siguientes.

- CmdActualiza. Botón para guardar los datos en el archivo abierto.
- CmdCalculos. Botón para abrir la ventana de cálculos.
- CmdCerrar. Botón para cerrar esta ventana.
- CmdEdita. Botón que abre la ventana que permite agregar y modificar los datos de la sección.

- CmdSeccion. Botón que permite visualizar la sección transversal del canal.
- GrdDatos. Malla que contiene los datos geométricos e hidráulicos de cada subsección del canal.
- LblNota. Etiqueta que contiene nota aclaratoria sobre la imagen de la sección transversal.
- PicSeccion. Caja de imagen que contiene el dibujo de la sección transversal.

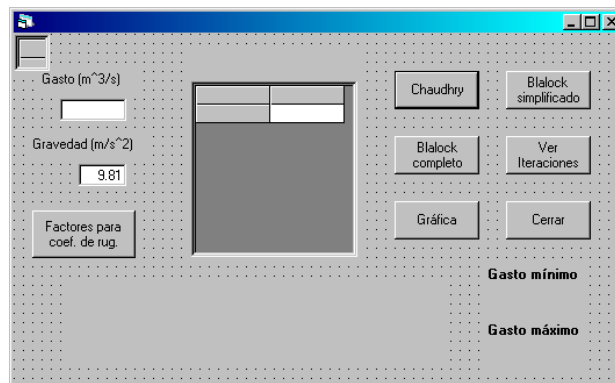
## FrmEdita

Esta pantalla tiene los elementos siguientes.

- CmbTipo. Combo en el que se selecciona el tipo de subsección central, derecha o izquierda.
- CmdAceptar. Botón para guardar los datos.
- CmdAnterior. Botón que presenta los datos de la subsección anterior.
- CmdCancelar. Botón que cancela los datos de pantalla y cierra la ventana.
- CmdEliminar. Botón que elimina los datos de la subsección en pantalla.
- CmdSiguiente. Botón que presenta los datos de la subsección siguiente.
- LblAlfa. Etiqueta que indica el coeficiente de coriolis de la subsección.
- LblAlfan. Etiqueta que indica el coeficiente  $\alpha_n$  de Keulegan de la subsección.
- LblB. Etiqueta que indica el ancho de plantilla de la subsección.
- LblC. Etiqueta que indica el coeficiente  $c$  de Keulegan de la subsección.
- LblKd. Etiqueta que indica el talud derecho de la subsección.
- LblKi. Etiqueta que indica el talud izquierdo de la subsección.
- LblKs. Etiqueta que indica la rugosidad relativa de la subsección.
- LblN. Etiqueta que indica el coeficiente  $n$  de rugosidad de Manning de la subsección.
- LblTipo. Etiqueta que indica el tipo de la subsección.
- LblYm. Etiqueta que indica la altura de la berma de inundación de la subsección.
- TxtAlfa. Caja de texto que contiene el coeficiente de coriolis de la subsección.
- TxtAlfan. Caja de texto que contiene el coeficiente  $\alpha_n$  de Keulegan de la subsección.
- TxtB. Caja de texto que contiene el ancho de plantilla de la subsección.

- TxtC. Caja de texto que contiene el coeficiente  $\epsilon$  de Keulegan de la subsección.
- TxtKd. Caja de texto que contiene el talud derecho de la subsección.
- TxtKi. Caja de texto que contiene el talud izquierdo de la subsección.
- TxtKs. Caja de texto que contiene la rugosidad relativa de la subsección.
- TxtN. Caja de texto que contiene el coeficiente  $n$  de rugosidad de Manning de la subsección.
- TxtYm. Caja de texto que contiene la altura de la berma de inundación de la subsección.

## FrmResultados



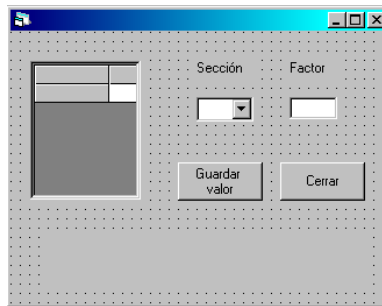
Esta pantalla tiene los elementos siguientes.

- CmdBlalock. Botón que inicia el cálculo de los tirantes críticos y caudales límite para el canal con el método de Blalock simplificado.
- CmdCerrar. Botón para cerrar esta ventana.
- CmdChaudhry. Botón que inicia el cálculo de los tirantes críticos y caudales límite para el canal con el método de Chaudhry.
- CmdFactor. Botón que presenta la pantalla donde se modifican los coeficientes que afectan la rugosidad para considerar la interacción entre las subsecciones del canal.
- CmdGrafica. Botón que presenta la gráfica Energía específica – Tirante para las condiciones analizadas.
- CmdGrid. Botón que presenta las iteraciones realizadas por el programa para obtener los resultados.
- CmdResultados. Botón que inicia el cálculo de los tirantes críticos y caudales límite para el canal con el método de Blalock completo.
- GrdResultados. Malla que presenta los tirantes críticos obtenidos por el programa.
- GrdTem. Malla que almacena los valores de las iteraciones.
- LblGasto. Etiqueta que indica el gasto con el que se analiza el canal.
- LblGravedad. Etiqueta que indica la fuerza de gravedad con el que se analiza el canal.
- LblNota. Etiqueta que contiene nota aclaratoria sobre los resultados obtenidos.
- LblNqmax. Etiqueta que indica el caudal límite superior para el canal.



- LblNqmin. Etiqueta que indica el caudal límite inferior para el canal.
- LblQmax. Etiqueta que contiene el caudal límite superior para el canal.
- LblQmin. Etiqueta que contiene el caudal límite inferior para el canal.
- LblResultado. Etiqueta que indica la malla donde se presentan los resultados.
- TxtGasto. Caja de texto que contiene el gasto con el que se analiza el canal.
- TxtGravedad. Caja de texto que contiene la fuerza de gravedad con el que se analiza el canal.

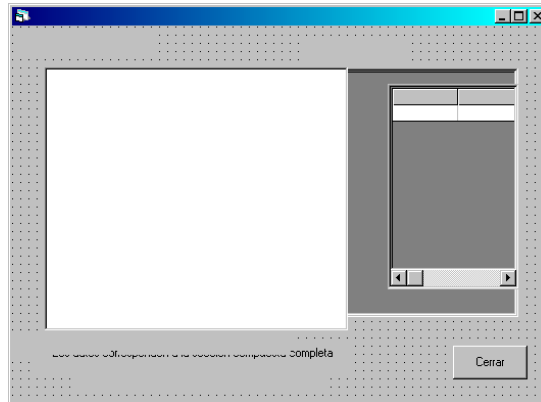
## FrmFactor



Esta pantalla tiene los elementos siguientes.

- CmbSeccion. Combo en el que se selecciona la subsección para la que se va a modificar el coeficiente que afecta la rugosidad para considerar la interacción entre las subsecciones del canal.
- CmdAceptar. Botón para guardar el valor del coeficiente.
- CmdCerrar. Botón para cerrar esta ventana.
- GrdFactor. Malla que contiene los coeficientes mencionados.
- LblFactor. Etiqueta que indica el coeficiente citado.
- LblNota. Etiqueta que contiene nota aclaratoria sobre dichos coeficientes.
- LblSeccion. Etiqueta que indica la subsección.
- TxtFactor. Caja de texto que contiene el coeficiente de la subsección.

## FrmGrafica



Esta pantalla tiene los elementos siguientes.

- CmdCerrar. Botón para cerrar esta ventana.
- GrdGrafica. Malla que presenta los valores de Energía específica y tirante graficados.
- GrdMalla. Malla que presenta los valores de las iteraciones realizadas.
- LblBerma. Etiqueta que indica el nivel de las bermas de inundación.
- LblEjes. Etiqueta que indica los ejes coordenados de la gráfica.
- LblEncabeza. Etiqueta que indica el gasto con que fue analizado el canal.
- LblMax. Etiqueta que indica las coordenadas del punto máximo de la gráfica.
- LblMin. Etiqueta que indica las coordenadas del punto mínimo de la gráfica.
- LblNota. Etiqueta que contiene nota aclaratoria sobre la gráfica.
- LblTitulo. Etiqueta que indica el título de la gráfica.
- LblYc. Etiqueta que indica el nivel de los tirantes críticos.
- PicGrafica. Caja de imagen que contiene la gráfica.

# ANEXO III CÓDIGO DE PROGRAMACIÓN

## MDIPrincipal

```
Private Sub MDIForm_Load()  
'al cargar la forma principal del sistema:  
  On Error GoTo error_handler  
  Me.Top = (Screen.Height - Me.Height) / 2: Me.Left = (Screen.Width - Me.Width) / 2  
  ReDim tipo(3): tipo(0) = "central": tipo(1) = "izquierda": tipo(2) = "derecha"  
  If configurado Then  
  'aquí pregunto si ya esta configurado el sistema  
    With Me  
      .MnuArchivo.Enabled = True: .MnuNuevo.Enabled = True: .MnuAbrir.Enabled = True  
      .MnuGuardar.Enabled = False: .MnuGuardarComo.Enabled = False: .MnuConfigura.Enabled = True  
      .MnuCerrar.Enabled = False: .MnuSalir.Enabled = True: .MnuCalculo.Enabled = False  
    End With  
    'al ser que ya, entonces habilito hacer nuevo archivo, abrir uno existente o volver a configurar pero no puedo guardar  
    'ni calcular algo que no he abierto  
  Else  
    With Me  
      .MnuArchivo.Enabled = True: .MnuNuevo.Enabled = False: .MnuAbrir.Enabled = False  
      .MnuGuardar.Enabled = False: .MnuGuardarComo.Enabled = False: .MnuConfigura.Enabled = True  
      .MnuCerrar.Enabled = False: .MnuSalir.Enabled = True: .MnuCalculo.Enabled = False  
    End With  
    mensaje = MsgBox("Recuerde que necesita configurar el sistema para poder usarlo", vbInformation + vbOKOnly, _  
      "¡Atención!")  
    'si no por errores o primera vez, deshabilito hacer nuevo archivo, abrir uno existente, o volver a configurar, ademas no  
    'puedo guardar ni calcular algo que no he abierto  
  End If  
  Exit Sub  
error_handler:  
'en caso de error salir de la subrutina  
  Exit Sub  
End Sub  
  
Private Sub MDIForm_Unload(Cancel As Integer)  
'termina la ejecución del programa  
  On Error GoTo error_handler  
  End  
  Exit Sub  
error_handler:  
'en caso de error salir de la subrutina  
  Exit Sub  
End Sub  
  
Private Sub mnuAbrir_Click()  
'con este menú se abre un archivo previamente creado  
  On Error GoTo error_handler  
  Me.MnuArchivo.Enabled = False: Me.MnuCalculo.Enabled = False  
  caso = "Abrir": Frmarchivo.Show  
  'si escojo abrir un archivo guardado  
  Exit Sub  
error_handler:  
'en caso de error salir de la subrutina  
  Exit Sub  
End Sub  
  
Private Sub mnucerrar_Click()  
'este menú cierra el archivo con el que se está trabajando
```

```

    On Error GoTo error_handler
    Unload FrmDatos: Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub mnuConfigura_Click()
'este menú configura el sistema
    On Error GoTo error_handler
    caso = "Configurar"
    If configurado Then
'si ya se configuro le pregunto antes de regarla
        mensaje = MsgBox("El sistema ya tiene especificado el archivo base" & Chr(13) & "¿Desea configurar nuevamente?"
-
        , vbYesNo + vbInformation, "¡Atención!")
        If mensaje = vbNo Then Exit Sub
'si decide que no, todo se queda igual, si no se ha configurado el sistema, se procede de forma normal
        Me.MnuArchivo.Enabled = False: Me.MnuCalculo.Enabled = False: Frmarchivo.Show
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub mnuGuardar_Click()
'este menú guarda los datos de pantalla en el archivo abierto
    On Error GoTo error_handler
    If revisadatosgrid(FrmDatos.Grddatos) Then
        MDIPrincipal.MnuCalculo.Enabled = True: FrmDatos.CmdCalculos.Enabled = True:
        FrmDatos.CmdSeccion.Enabled = True
    End If
    If actualiza(FrmDatos.Grddatos, ubicacion) Then
        A = MsgBox("Se actualizaron los datos de pantalla en: " & archivoenuso, vbOKOnly + vbInformation, "¡Atención!")
    Else
        A = MsgBox("No se logró actualizar los datos de pantalla en: " & archivoenuso, vbOKOnly + vbCritical, _
        "¡Atención!")
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub mnuguardarcomo_Click()
'este menú guarda los datos de pantalla en un nuevo archivo
    On Error GoTo error_handler
    Me.MnuArchivo.Enabled = False: Me.MnuCalculo.Enabled = False
    caso = "Guardar": Frmarchivo.Show
'si se guarda el archivo que se esta trabajando
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub mnuNuevo_Click()
'este menú crea un nuevo archivo de datos
    On Error GoTo error_handler
    Me.MnuArchivo.Enabled = False: Me.MnuCalculo.Enabled = False
    caso = "Nuevo": Frmarchivo.Show

```

```

Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub mnuResultados_Click()
'este menú presenta la forma en la que se procede a obtener los tirantes críticos
On Error GoTo error_handler
Me.MnuArchivo.Enabled = False: Me.MnuCalculo.Enabled = False: Me.MnuResultados.Enabled = False
FrmDatos.Enabled = False: FrmResultados.Show
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub mnuSalir_Click()
'al hacer clic en este menú, termina la ejecución del programa
On Error GoTo error_handler
End
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

```

## **FrmArchivo**

```

Private Sub cmdAceptar_Click()
'al aceptar se procede de acuerdo con la opción seleccionada en un principio
On Error GoTo error_handler
Select Case caso
Case "Abrir"
'si se abre el archivo
If configurado Then
'verifico que ya este configurado el sistema
If Not (Right(TxtArchivo.Text, 4) = ".dbf") And Len(TxtArchivo.Text) > 0 Then
mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) & _
"Verifique que la extensión sea .dbf", vbInformation + vbOKOnly, "¡Atención!")
Exit Sub
End If
'verifico la version del archivo
If TxtArchivo.Text = "*.dbf" Or Len(TxtArchivo.Text) = 0 Then
mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) _
, vbInformation + vbOKOnly, "¡Atención!")
Exit Sub
End If
'verifico que no sea el archivo que se marca de default
If Len(TxtArchivo.Text) > 12 Then
mensaje = MsgBox("El nombre del archivo " & TxtArchivo.Text & " tiene más de 8 caracteres" _
& Chr(13) & "El sistema no lo puede abrir" & ", necesitará cambiarle el nombre" & Chr(13), _
vbInformation + vbOKOnly, "¡Atención!")
Exit Sub
End If
'aquí se establece la limitante de sólo permitir archivos con nombre no mayor a ocho caracteres
ubicacion = Dir1.Path & "\\" & TxtArchivo.Text
If Not (accesaarchivo(ubicacion)) Then
mensaje = MsgBox("El archivo que seleccionó no tiene las características" & "necesarias" & Chr(13) _
& "por favor seleccione un archivo adecuado", vbInformation + vbOKOnly, "¡Atención!")
Exit Sub

```

```

    End If
Else
    Exit Sub
    ' si no esta configurado no hace nada
End If
Case "Nuevo"
'si se trata de un nuevo archivo
If configurado Then
'verifico que ya este configurado el sistema
If Not (Right(TxtArchivo.Text, 4) = ".dbf") And Len(TxtArchivo.Text) > 0 Then
    mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) & _
        "Verifique que la extensión sea .dbf", vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
End If
'verifico la version del archivo
If TxtArchivo.Text = "*.dbf" Or Len(TxtArchivo.Text) = 0 Then
    mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) _
        , vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
End If
'verifico que no sea el archivo que se marca de default
If Len(TxtArchivo.Text) > 12 Then
    mensaje = MsgBox("El nombre del archivo " & TxtArchivo.Text & " tiene más de 8 caracteres" & _
        Chr(13) & "El sistema no lo podrá abrir necesita cambiarle el nombre" & Chr(13), vbInformation _
        + vbOKOnly, "¡Atención!")
    Exit Sub
End If
'aquí se establece la limitante de sólo permitir archivos con nombre no mayor a ocho caracteres
TxtArchivo.Text = verificanombre(TxtArchivo.Text)
If Not crea(Dir1, TxtArchivo) Then Exit Sub
'se crea el archivo
Else
    Exit Sub
    ' si no esta configurado me manda al cuerno
End If
Case "Guardar"
'si se trata de guardar el archivo
If Not (Right(TxtArchivo.Text, 4) = ".dbf") And Len(TxtArchivo.Text) > 0 Then
    mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) & _
        "Verifique que la extensión sea .dbf", vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
End If
'verifico la version del archivo
If TxtArchivo.Text = "*.dbf" Or Len(TxtArchivo.Text) = 0 Then
    mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no es válido" & Chr(13) _
        , vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
End If
'verifico que no sea el archivo que se marca de default
If Len(TxtArchivo.Text) > 12 Then
    mensaje = MsgBox("El nombre del archivo " & TxtArchivo.Text & " tiene más de 8 caracteres" & Chr(13) _
        & "El sistema no lo podrá abrir, necesita cambiarle el nombre" & Chr(13), vbInformation + vbOKOnly, _
        "¡Atención!")
    Exit Sub
End If
TxtArchivo.Text = verificanombre(TxtArchivo.Text)
If Not crea(Dir1, TxtArchivo) Then Exit Sub
'creo el archivo para guardar los datos
exito = guarda(FrmDatos.Grddatos, ubicacion)
If exito Then
'guardo los datos del grid en el archivo

```

```

        A = MsgBox("Se guardó el archivo: " & TxtArchivo.Text, vbOKOnly + vbInformation, "¡Atención!")
        If revisadatosgrid(FrmDatos.GrdDatos) Then
            MDIPrincipal.MnuCalculo.Enabled = True
        End If
    Else
        A = MsgBox("No se pudo guardar el archivo: " & TxtArchivo.Text, _vbOKOnly + vbCritical, "¡Atención!")
        destruye Dir1, TxtArchivo
        Exit Sub
    End If
Case "Configurar"
'si se trata de configurar el sistema
    If Not (TxtArchivo.Text = File1.FileName) And Not (TxtArchivo.Text = renglon2) Then
        mensaje = MsgBox("El archivo " & TxtArchivo.Text & " no existe" & Chr(13) & _
            "Seleccione un archivo que sea válido", vbInformation + vbOKOnly, "¡Atención!")
        Exit Sub
    End If
    'se verifica que sea el archivo valido
    If Not configurar(Dir1, File1, TxtArchivo) Then Exit Sub
    'configura el sistema
Case Else
    'si no es ninguno de los anteriores
End Select
Unload Me
If caso = "Nuevo" Or caso = "Abrir" Then
    Load FrmDatos
ElseIf caso = "Guardar" Then
    FrmDatos.Enabled = True
    If accesaarchivo(ubicacion) Then FrmDatos.Caption = Mid(archivoenuso, 1, Len(archivoenuso) - 4)
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdCancelar_Click()
'Al cancelar la selección se revisa si el archivo y los datos son válidos para los cálculos
    On Error GoTo error_handler
    Unload Me
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Dir1_Change()
'cuando selecciona el directorio, actualiza las carpetas en pantalla
    On Error GoTo error_handler
    File1.Path = Dir1.Path
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Dir1_KeyPress(KeyAscii As Integer)
'al presionar la tecla de aceptar sobre el directorio, actualiza las carpetas en pantalla
    On Error GoTo error_handler
    If KeyAscii = 13 Then
        Dir1.Path = Dir1.List(Dir1.ListIndex)
    End If
End Sub

```

```

    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Drive1_Change()
'al cambiar la unidad de disco actualiza los directorios en esa unidad
    On Error GoTo error_handler
    Dir1.Path = Drive1.Drive
    Exit Sub
error_handler:
'en caso de error envía un mensaje y no se pierde el sistema
    If Err.Number = 68 Then
        x = MsgBox("La unidad " & Drive1.Drive & " no está disponible", vbOKOnly + vbCritical, "Error")
        Drive1.SetFocus: Drive1.Drive = Mid(Dir1.Path, 1, 2): Dir1.Path = Drive1.Drive
    End If
End Sub

Private Sub File1_Click()
'al dar clic sobre un archivo, aparece el nombre del mismo en la caja de texto
'respectiva
    On Error GoTo error_handler
    If Len(File1.FileName) > 0 Then
        TxtArchivo.Text = File1.FileName
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub File1_DblClick()
'al dar doble clic sobre un archivo, selecciona el archivo y lo abre
    On Error GoTo error_handler
    If Len(File1.FileName) = 0 Or File1.FileName = "*.dbf" Then
        Exit Sub
    Else
        cmdaceptar_Click
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub File1_KeyPress(KeyAscii As Integer)
'al presionar la tecla de aceptar teniendo un archivo seleccionado lo abre
    On Error GoTo error_handler
    If KeyAscii = 13 Then
        If Len(File1.FileName) = 0 Or File1.FileName = "*.dbf" Then
            Exit Sub
        Else
            cmdaceptar_Click
        End If
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub

```



End Sub

Private Sub Form\_KeyPress(KeyAscii As Integer)

'esta función pasa la tecla de aceptar de la forma a las cajas de texto

On Error GoTo error\_handler

If KeyAscii = 13 Then cmdAceptar\_Click

'cuando presiono enter en la caja de texto actúa como si hubiera hecho clic en aceptado

If KeyAscii = 27 Then CmdCancelar\_Click

'cuando presiono esc en la caja de texto actúa como si hubiera hecho clic en cancelar

Exit Sub

error\_handler:

'en caso de error salir de la subrutina

Exit Sub

End Sub

Private Sub Form\_Load()

'al cargar la forma pone el nombre de la acción a efectuar como título de la ventana

On Error GoTo error\_handler

Me.Caption = caso: Me.Left = 500: Me.Top = 500: Me.Height = 5000: Me.Width = 8000

If configurado Then

Select Case caso

'ubico la forma para seleccionar el archivo segun sea el caso seleccionado

Case "Abrir"

'si se trata de dar de abrir un nuevo archivo para trabajar un problema

If hallacarpeta(App.Path, App.Path & "\datos") Then

'verifica si ya existe la carpeta de datos para abrir los archivos guardados previamente. si no existe, la crea

Dir1.Path = App.Path & "\datos"

'presenta esta carpeta como default

End If

Case "Nuevo"

'si se trata de dar de alta un nuevo archivo para trabajar un problema

If hallacarpeta(App.Path, App.Path & "\datos") Then

'verifica si ya existe la carpeta de datos para guardar los archivos nuevos si no existe, la crea

Dir1.Path = App.Path & "\datos"

'presenta esta carpeta como default

End If

Case "Guardar"

'si se trata de dar de guardar los cambios de un archivo que se está trabajando

If hallacarpeta(App.Path, App.Path & "\datos") Then

'verifica si ya existe la carpeta de datos para abrir los archivos guardados previamente. si no existe, la crea

If accesaarchivo(ubicacion) Then Dir1.Path = carpetaenuso: TxtArchivo.Text = archivoenuso

'presenta esta carpeta como default

End If

FrmDatos.Enabled = False

Case "Configurar"

'si se reconfigura el sistema

Dir1.Path = renglon1: TxtArchivo.Text = renglon2

'pongo el directorio predefinido en la configuracion del sistema

Case Else

End Select

Else

'si no esta configurado,

Select Case caso

'ubico la forma para seleccionar el archivo segun sea el caso seleccionado

Case "Configurar"

'si se reconfigura el sistema

Dir1.Path = App.Path & "\sistema"

End Select

End If

Exit Sub

error\_handler:

```

'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Form_Unload(Cancel As Integer)
'al salir la función activa y desactiva algunos elementos de la forma principal
    On Error GoTo error_handler
    Me.Hide
    Select Case caso
        Case "Configurar"
            MDIPrincipal.MnuArchivo.Enabled = True: MDIPrincipal.MnuCalculo.Enabled = False
            'al salir de configurar que no habilite el cálculo de algo que no se ha abierto
        Case "Nuevo"
            MDIPrincipal.MnuArchivo.Enabled = True
            'después de crear un archivo de trabajo no habilita el cálculo ya que no hay datos
        Case "Abrir"
            MDIPrincipal.MnuArchivo.Enabled = True: MDIPrincipal.MnuCalculo.Enabled = False
            'después de abrir un archivo de trabajo no habilita el cálculo ya que no se ha verificado que haya datos
        Case "Guardar"
            If revisadatosgrid(FrmDatos.GrdDatos) And central Then
                MDIPrincipal.MnuCalculo.Enabled = True: FrmDatos.CmdCalculos.Enabled = True
            End If
            FrmDatos.Enabled = True
            MDIPrincipal.MnuArchivo.Enabled = True
            'después de guardar los datos en el archivo de trabajo no habilita el cálculo ya que no se ha verificado que
            'haya datos (puede guardar ningún registro)
        Case Else
            MDIPrincipal.MnuArchivo.Enabled = True: MDIPrincipal.MnuCalculo.Enabled = True
    End Select
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub TxtArchivo_KeyPress(KeyAscii As Integer)
'esta función permite que al presionar las teclas que se indican , se proceda a las subrutinas respectivas
    On Error GoTo error_handler
    If KeyAscii = 13 Then cmdaceptar_Click
    'cuando presiono enter en la caja de texto actúa como si hubiera hecho clic en aceptado
    If KeyAscii = 27 Then CmdCancelar_Click
    'cuando presiono esc en la caja de texto actúa como si hubiera hecho clic en cancelar
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

```

## **FrmDatos**

```

Private Sub cmdactualiza_Click()
'este botón actualiza los valores que aparecen en el grid en el archivo que está en uso
    On Error GoTo error_handler
    If revisadatosgrid(FrmDatos.GrdDatos) And central Then
        MDIPrincipal.MnuCalculo.Enabled = True: FrmDatos.CmdCalculos.Enabled = True
        FrmDatos.CmdSeccion.Enabled = True
    End If
    If actualiza(GrdDatos, ubicacion) Then
        A = MsgBox("Se actualizaron los datos de pantalla en: " & archivoenuso, vbOKOnly + vbInformation, "¡Atención!")
    Else
        A = MsgBox("No se logró actualizar los datos de pantalla en: " & archivoenuso, vbOKOnly + vbCritical, _

```

```

        ",¡Atención!")
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdCalculos_Click()
'este botón activa la ventana donde se pueden realizar los cálculos
    On Error GoTo error_handler
    With MDIPrincipal
        .MnuArchivo.Enabled = False: .MnuCalculo.Enabled = False: .MnuResultados.Enabled = False
    End With
    FrmDatos.Enabled = False: FrmResultados.Show
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Cmdcerrar_Click()
'este botón cierra la forma
    On Error GoTo error_handler
    Unload Me
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Cmdedita_Click()
'con este botón llamo a la forma que me permite actualizar los valores de cada sección
    On Error GoTo error_handler
    Frmmedita.Show: Me.Enabled = False
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Cmdseccion_Click()
'con este botón se presenta un dibujo de la sección transversal del canal
    On Error GoTo error_handler
    If CmdSeccion.Caption = "Ver sección" Then
'para ver la sección
        With FrmDatos
            .Caption = "Sección transversal para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
            .CmdActualiza.Visible = False: .CmdCalculos.Visible = False: .CmdCerrar.Visible = False
            .CmdEdita.Visible = False: .GrdDatos.Visible = False: .PicSeccion.Visible = True
            .LblNota.Visible = True: .CmdSeccion.Caption = "Cerrar": .CmdSeccion.ToolTipText = "Cierre esta ventana"
            If dibujo Then
                .LblNota.Caption = "Las líneas ortogonales en la esquina inferior izquierda del dibujo representan 1m en "& _
                    "direcciones vertical y horizontal respectivamente."
            End If
        End With
    ElseIf CmdSeccion.Caption = "Cerrar" Then
'para cerrar la ventana de imagen de la sección y regresar a la pantalla de datos
        With FrmDatos
            .Caption = "Datos para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
            .CmdActualiza.Visible = True: .CmdCalculos.Visible = True: .CmdCerrar.Visible = True

```

```

        .CmdEdita.Visible = True: .GrdDatos.Visible = True: .PicSeccion.Visible = False
        .LblNota.Visible = False: .CmdSeccion.Caption = "Ver sección": .CmdSeccion.ToolTipText = "Vea la sección"
    End With
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Form_Load()
'al cargar la forma donde se presentan los datos, le pongo el nombre del archivo al título de la forma
On Error GoTo error_handler
accesaarchivo ubicacion
Me.Caption = "Datos para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
Me.Left = 500: Me.Top = 500: Me.Height = 5000: Me.Width = 8000: GrdDatos.Clear
'limpio el grid
With MDIPrincipal
.MnuNuevo.Enabled = False: .MnuAbrir.Enabled = False: .MnuCerrar.Enabled = True
.MnuConfigura.Enabled = False: .MnuCalculo.Enabled = False: FrmDatos.CmdCalculos.Enabled = False
FrmDatos.CmdSeccion.Enabled = False
'habilito y deshabilito algunos menús
If titulos("campos") Then
'cargo los títulos de campos en la variable respectiva
estado = llenagrid(GrdDatos, ubicacion, Me)
'lleno el grid y obtengo la respuesta de la cantidad de registros
If estado = "lleno" Then
.MnuGuardar.Enabled = True: .MnuGuardarComo.Enabled = True:
ordenagrid GrdDatos, 4
'esta función ordena los datos del grid
actualiza GrdDatos, ubicacion
'esta función actualiza los datos del grid en el archivo en uso
If revisadatosgrid(GrdDatos) And central Then
MDIPrincipal.MnuCalculo.Enabled = True
FrmDatos.CmdCalculos.Enabled = True: FrmDatos.CmdSeccion.Enabled = True
End If
'esta función revisa que los datos cargados en el grid sean correctos y en tal caso, habilita la opción del
'cálculo
ElseIf estado = "vacío" Then .MnuGuardar.Enabled = True: .MnuGuardarComo.Enabled = True
ElseIf estado = "falla" Then .MnuGuardar.Enabled = False: .MnuGuardarComo.Enabled = False
End If
End If
End With
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Form_Unload(Cancel As Integer)
'esta subrutina activa y desactiva elementos de la forma principal al cerrar la ventana
On Error GoTo error_handler
With MDIPrincipal
'al salir habilito y deshabilito algunos menús
.MnuArchivo = True: .MnuNuevo.Enabled = True: .MnuAbrir.Enabled = True
.MnuGuardar.Enabled = False: .MnuGuardarComo.Enabled = False: FrmDatos.CmdCalculos.Enabled = False
FrmDatos.CmdSeccion.Enabled = False: .MnuCerrar.Enabled = False: .MnuConfigura.Enabled = True
.MnuCalculo.Enabled = False
End With
Exit Sub
error_handler:

```

```
'en caso de error salir de la subrutina
Exit Sub
End Sub
```

## FrmEdita

```
Private Sub cmbtipo_Change()
'con esta función predetermino los valores que pueden tomar los taludes en función del tipo de sección si se trata de la sección
'central, deshabilito la altura de berma, pues no es coherente y viceversa si es una sección lateral
```

```
On Error GoTo error_handler
If CmbTipo.Text = "central" Then
'sección central
TxtKi.Enabled = True: LblKi.Enabled = True
If TxtKi.Text = "" Then TxtKi.Text = 1
TxtKd.Enabled = True: LblKd.Enabled = True
If TxtKd.Text = "" Then TxtKd.Text = 1
TxtYm.Enabled = False: LblYm.Enabled = False
ElseIf CmbTipo.Text = "izquierda" Then
'sección izquierda
TxtKi.Enabled = True: LblKi.Enabled = True
If TxtKi.Text = "" Then TxtKi.Text = 1
TxtKd.Text = "": TxtKd.Enabled = False: LblKd.Enabled = False
TxtYm.Enabled = True: LblYm.Enabled = True
ElseIf CmbTipo.Text = "derecha" Then
'sección derecha
TxtKi.Text = "": TxtKi.Enabled = False: LblKi.Enabled = False
TxtKd.Enabled = True: LblKd.Enabled = True
If TxtKd.Text = "" Then TxtKd.Text = 1
TxtYm.Enabled = True: LblYm.Enabled = True:
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub
```

```
Private Sub cmbtipo_Click()
'esta función llama a la de cambio en el combo
On Error GoTo error_handler
cmbtipo_Change
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub
```

```
Private Sub cmdaceptar_Click()
'esta función guarda los valores de las cajas de texto en el grid
On Error GoTo error_handler
With FrmDatos
datoscorrectos = verificadatos(Mid(Me.Caption, 9, Len(Me.Caption) - 8), CmbTipo.Text, TxtB.Text, TxtKi.Text,
TxtKd.Text, TxtYm.Text, TxtAlfa.Text, TxtC.Text, TxtAlfan.Text, TxtKs.Text, GrdDatos, TxtN.Text)
'verifico que los datos que estoy dejando no provoquen incoherencias en el cálculo posterior
If datoscorrectos <> "" Then
'si hay un error, lo hago saber
z = MsgBox(datoscorrectos, vbInformation + vbOKOnly, "¡Atención!")
Exit Sub
End If
z = llenadatosformagrid(Me, FrmDatos.GrdDatos, registro)
'vacío los datos de las cajas de texto en la fila respectiva del grid
End With
```

```

    CmdCancelar.Caption = "Cerrar": CmdCancelar.ToolTipText = "Cerrar esta ventana"
Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdAnterior_Click()
'esta función carga los valores de la fila anterior en las cajas de texto de la forma
On Error GoTo error_handler
With FrmDatos
    datoscorrectos = verificadatos(Mid(Me.Caption, 9, Len(Me.Caption) - 8), CmbTipo.Text, TxtB.Text, TxtKi.Text, _
    TxtKd.Text, TxtYm.Text, TxtAlfa.Text, TxtC.Text, TxtAlfan.Text, TxtKs.Text, .GrdDatos, TxtN.Text)
'verifico que los datos que estoy dejando no provoquen incoherencias en el cálculo posterior
If datoscorrectos <> "" Then
'si hay un error, lo hago saber
    z = MsgBox(datoscorrectos, vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
End If
If llenadatosformagrid(Me, FrmDatos.GrdDatos, registro) Then
'vacío los datos de las cajas de texto en la fila respectiva del grid
    registro = registro - 1
'me voy al registro anterior
    .GrdDatos.Row = registro
'y me ubico en la fila respectiva
If .GrdDatos.Row > 1 Then
'si la fila a la que llego en este proceso es la primera de datos
    CmdAnterior.Enabled = True
'habilito este botón
Else
    CmdAnterior.Enabled = False
'deshabilito este botón
End If
A = llenadatosforma(.GrdDatos, Me, registro)
'lleno las cajas de texto de la forma con los datos de la fila a la que llego
End If
End With
CmdCancelar.Caption = "Cancelar"
Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdEliminar_Click()
'este botón elimina del grid el registro que aparece en la forma
On Error GoTo error_handler
TxtB.Text = "": TxtKi.Text = "": TxtKd.Text = "": TxtYm.Text = "": TxtAlfa.Text = ""
TxtC.Text = "": TxtAlfan.Text = "": TxtKs.Text = ""
'vacío todas las cajas de texto
If eliminafila(FrmDatos.GrdDatos, registro) Then
'elimino del grid el registro que está mostrado en la forma
If registro = 1 And FrmDatos.GrdDatos.Rows > 1 Then
'si el registro es el primero y hay más datos en el grid
    registro = registro
'me muevo al siguiente automáticamente, pues ya hay una fila menos
ElseIf registro = 1 Then
'si el registro es el primero y no hay más datos en el grid
    A = limpiafilas(FrmDatos.GrdDatos)
    Me.Caption = "Sección 1": CmbTipo.ListIndex = 0: TxtAlfa.Text = "1"
    TxtAlfan.Text = "2": TxtC.Text = "12.64": FrmDatos.GrdDatos.Rows = 2

```

```

    CmdEliminar.Enabled = False
    z = MsgBox("Se han borrado todas las secciones" & Chr(13) & "En la pantalla se muestran los valores sugeridos"_
    & Chr(13) & "para la primera sección", vbInformation + vbOKOnly, "¡Atención!")
    Exit Sub
ElseIf registro > 1 Then
'si el registro es no el primero
    registro = registro - 1
    'me muevo al anterior
End If
With FrmDatos
    A = limpiafilas(GrdDatos)
    'elimino todas las filas vacías
    If llenadatosforma(FrmDatos.GrdDatos, Me, registro) Then
    'lleno los datos de la forma con el nuevo registro
        .GrdDatos.Row = registro
        If .GrdDatos.Row > 1 Then
            'si la fila a la que llevo en este proceso es la primera de datos
            CmdAnterior.Enabled = True
            'habilito este botón
        Else
            CmdAnterior.Enabled = False
            'deshabilito este botón
        End If
    End If
    If .GrdDatos.Rows = 1 Then CmdEliminar.Enabled = False
    'si me han quedado ningún registro de datos, pues no puedo eliminar
End With
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdCancelar_Click()
'este botón descarga la forma
    On Error GoTo error_handler
    Unload Me
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub CmdSiguiente_Click()
'esta función carga los valores de la fila siguiente en las cajas de texto de la forma
    On Error GoTo error_handler
    With FrmDatos
        datoscorrectos = verificadatos(Mid(Me.Caption, 9, Len(Me.Caption) - 8), CmbTipo.Text, TxtB.Text, TxtKi.Text, _
        TxtKd.Text, TxtYm.Text, TxtAlfa.Text, TxtC.Text, TxtAlfan.Text, TxtKs.Text, .GrdDatos, TxtN.Text)
        'verifico que los datos que estoy dejando no provoquen incoherencias en el cálculo posterior
        If datoscorrectos <> "" Then
            'si hay un error, lo hago saber
            z = MsgBox(datoscorrectos, vbInformation + vbOKOnly, "¡Atención!")
            Exit Sub
        End If
        If llenadatosformagrid(Me, FrmDatos.GrdDatos, registro) Then
            'vacío los datos de las cajas de texto en la fila respectiva del grid
            registro = registro + 1
            'me voy al registro siguiente
            If registro = .GrdDatos.Rows Then

```

```

' si ya llegué al límite superior y me muevo al siguiente
  .GrdDatos.Rows = .GrdDatos.Rows + 1
  'creo una fila más
  .GrdDatos.Row = registro
  'me ubico en dicha fila
  z = verificatipo(FrmDatos.GrdDatos)
  'busco la sección central, si existe
  A = llenadatosforma(.GrdDatos, Me, registro)
  'lleno las cajas de texto de la forma con los datos del grid
  If central Then
  'en caso de que existiese ya una sección central
    CmbTipo.Text = "derecha"
    'asigno el valor de sección derecha como predeterminada
  Else
    CmbTipo.Text = "central"
    'si no predetermino la sección central
  End If
  TxtAlfa.Text = "1": TxtC.Text = "12.64": TxtAlfan = "2": TxtKs = "0.03"
  'doy los siguientes valores predeterminados para las variables respectivas
Else
'si aun no llego al límite superior
  .GrdDatos.Row = registro
  A = llenadatosforma(.GrdDatos, Me, registro)
  'lleno las cajas de texto de la forma con los datos de la fila a la que llego
End If
TxtB.SetFocus
If registro > 1 Then
  CmdAnterior.Enabled = True: CmdEliminar.Enabled = True
  'si la fila a la que llego es la segunda o posterior habilito el botón de moverme al registro anterior
End If
End If
End With
CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Form_Load()
'al cargar la forma de edición de datos
On Error GoTo error_handler
MDIPrincipal.MnuArchivo.Enabled = False: MDIPrincipal.MnuCalculo.Enabled = False
FrmDatos.CmdCalculos.Enabled = False: FrmDatos.CmdSeccion.Enabled = False
'deshabilito los menús del sistema
Me.Left = 500: Me.Top = 500: Me.Height = 5000: Me.Width = 8000
registro = 0
'esta variable me indica el registro en que me encuentro, inicializa con el valor 0
With CmbTipo
'limpio el combo del tipo de sección y cargo los valores que usaré
.Clear: .AddItem "central", 0: .AddItem "izquierda", 1: .AddItem "derecha", 2
'verifico si ya existe alguna sección central, si no, al cargar los datos se predeterminará la sección tipo central
If central = False Then
.ListIndex = 0
Else
.ListIndex = 1
End If
End With
'inicio con el registro uno
registro = 1
If FrmDatos.GrdDatos.Rows = 1 Then

```



```

'si no hay datos, se cargan estos por descuento
    Me.Caption = "Sección 1": TxtAlfa.Text = "1": TxtAlfan.Text = "2":          TxtC.Text = "12.64"
    FrmDatos.GrdDatos.Rows = 2: CmdEliminar.Enabled = False: CmbTipo.ListIndex = 0
Else
'si si existen datos, cargo los que corresponden a la fila primera
    A = llenadatosforma(FrmDatos.GrdDatos, Me, registro)
    If CmbTipo.Text = "" Then CmbTipo.ListIndex = 0
    CmdEliminar.Enabled = True
    'if verificar para habilitar el cálculo
End If
CmdAnterior.Enabled = False: CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Form_Unload(Cancel As Integer)
'al salir de la forma
    On Error GoTo error_handler
    If CmdCancelar.Caption = "Cancelar" Then
        z = MsgBox("No se guardarán los datos capturados en esta sección" & Chr(13) & "ni los cambios que haya "& _
            "hecho en la misma", vbInformation + vbOKOnly, "¡Atención!")
        'advierito que no se guardarán los cambios realizados en la última pantalla
    End If
    A = limpiafilas(FrmDatos.GrdDatos)
    'elimino todas las filas vacías
    A = ordenagrid(FrmDatos.GrdDatos, 4)
    FrmDatos.Enabled = True: MDIPrincipal.MnuArchivo.Enabled = True
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub txtalfa_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
    On Error GoTo error_handler
    CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub txtalfan_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
    On Error GoTo error_handler
    CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub txtB_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
    On Error GoTo error_handler
    CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
    Exit Sub

```

```

error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtc_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler
  CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtkd_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler
  CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
  Exit Sub
error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtki_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler
  CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
  Exit Sub
error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtks_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler: CmdCancelar.Caption = "Cancelar"
  CmdCancelar.ToolTipText = "Cancelar los cambios"
  Exit Sub
error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtym_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler
  CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
  Exit Sub
error_handler:
'en caso de error salir de la subrutina
  Exit Sub
End Sub

Private Sub txtn_Change()
'al cambiar este dato, se cambia el botón de cerrar a cancelar
  On Error GoTo error_handler
  CmdCancelar.Caption = "Cancelar": CmdCancelar.ToolTipText = "Cancelar los cambios"
  Exit Sub
error_handler:

```

```
'en caso de error salir de la subrutina
    Exit Sub
End Sub
```

## **FrmFactor**

```
Private Sub cmbseccion_Change()
'al cambiar el texto de este combo, se actualiza en la caja de texto el valor del factor de la sección mostrada
    On Error GoTo error_handler
    Me.TxtFactor = nFactorn(CmbSeccion.ListIndex + 1)
    If nfactornc(CmbSeccion.ListIndex + 1) Then
        Me.TxtFactor.Enabled = True: Me.CmdAceptar.Enabled = True
        Me.CmdCerrar.Caption = "Cancelar": Me.CmdCerrar.ToolTipText = "Cancelar los cambios"
    Else
        Me.TxtFactor.Enabled = False: Me.CmdAceptar.Enabled = False: Me.CmdCerrar.Caption = "Cerrar"
        Me.CmdCerrar.ToolTipText = "Salir de esta ventana"
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub
```

```
Private Sub cmbseccion_Click()
'al hacer clic en este combo se llama a la subrutina de cambio en el texto del combo
    On Error GoTo error_handler
    cmbseccion_Change
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub
```

```
Private Sub cmdAceptar_Click()
'Al presionar este botón se guardan los cambios para el factor respectivo, y se cambia el botón de cancelar a salir
    On Error GoTo error_handler
    If Not continuafactor(TxtFactor.Text) Then
        Exit Sub
    End If
    nFactorn(CmbSeccion.ListIndex + 1) = TxtFactor.Text: llenagridfactor Me.GrdFactor
    Me.CmdCerrar.Caption = "Cerrar": Me.CmdCerrar.ToolTipText = "Salir de esta ventana"
    Me.CmbSeccion.SetFocus
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub
```

```
Private Sub Cmdcerrar_Click()
'al presionar este botón, se cancelan los cambios aplicados para el factor respectivo, o bien se slae de la forma, dependiendo
del 'punto en donde nos encontremos
    On Error GoTo error_handler
    If CmdCerrar.Caption = "Cerrar" Then
        Unload Me
    ElseIf CmdCerrar.Caption = "Cancelar" Then
        Me.CmbSeccion.SetFocus: CmdCerrar.Caption = "Cerrar"
        Me.CmdCerrar.ToolTipText = "Salir de esta ventana"
    End If
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
```

```
Exit Sub
End Sub
```

```
Private Sub Form_Load()
'al cargar la forma llena la malla con los datos de los factores para el coeficiente de rugosidad de Manning n, para ayudar a
'considerar los efectos de la transferencia de momentum en las intercaras
On Error GoTo error_handler
accesaarchivo ubicacion
With Me
.Left = 500: .Top = 500: .Height = 4000: .Width = 5000
.Caption = "Factores de Coeficientes de rugosidad (n) para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
End With
FrmResultados.Enabled = False
'considerar para solo un lado con llanuras
If UBound(nymder) = 1 And nymder(1) = mayorlista(nymizq) * 1000 Then
nfactornc(encuentramayor("izquierda")) = False
ElseIf UBound(nymizq) = 1 And nymizq(1) = mayorlista(nymder) * 1000 Then
nfactornc(encuentramayor("derecha")) = False
Else
nfactornc(encuentramayor("izquierda")) = False: nfactornc(encuentramayor("derecha")) = False
End If
Me.LblNota.Caption = "Estos factores modifican los coeficientes de rugosidad n de Manning para considerar el "& _
"efecto de fricción en las intercaras de cada sección"
llenacombo Me.CmbSeccion: llenagridfactor Me.GrdFactor: Me.CmbSeccion.ListIndex = 0
Me.CmdCerrar.Caption = "Cancelar": Me.CmdCerrar.ToolTipText = "Cancelar los cambios"
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
'al salir de la forma, vuelve a activar la forma anterior y manda el foco del sistema a la caja de texto que obtendrá el valor del
'gasto
On Error GoTo error_handler
With FrmResultados
.Enabled = True: .TxtGasto.SetFocus
End With
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub
Private Sub txtfactor_KeyDown(KeyCode As Integer, Shift As Integer)
'cuando presiono la tecla de aceptar, ejecuta la subrutina de aceptar
On Error GoTo error_handler
If KeyCode = 13 Then cmdaceptar_Click
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub
```

## **FrmGrafica**

```
Private Sub Cmdcerrar_Click()
'al hacer clic en este botón se llama a la función de descargar la forma
On Error GoTo error_handler
Unload Me
Exit Sub
error_handler:
```

```

'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Form_Load()
'al cargar la forma, de acuerdo a lo seleccionado en la forma anterior, se muestran las iteraciones o la grafica con los
indicadores
    On Error GoTo error_handler
    accesaarchivo ubicacion
    With Me
        .Left = 500: .Top = 500: .Height = 6000: .Width = 8000
        .lblcabeza.Caption = "Gasto de " & Q & " m^3/s"
        If caso1 = "graficar" Then
            .Caption = "Gráfica para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
            .LblBerma.Visible = True: .LblMax.Visible = True: .LblMin.Visible = True
            .GrdGrafica.Visible = True: .LblYc.Visible = True: .LblTitulo.Visible = True
            .PicGrafica.Visible = True: .GrdMalla.Visible = False: .lblnota.Visible = False
        ElseIf caso1 = "malla" Then
            .Caption = "Iteraciones para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)
            .LblBerma.Visible = False: .LblEjes.Visible = False: .LblMax.Visible = False
            .LblMin.Visible = False: .GrdGrafica.Visible = False: .LblYc.Visible = False
            .LblTitulo.Visible = False: .PicGrafica.Visible = False: .GrdMalla.Visible = True
            .lblnota.Visible = True
        End If
    End With
    FrmResultados.Enabled = False
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

Private Sub Form_Unload(Cancel As Integer)
'al descargar la forma se desactivan todos los objetos de la forma
    On Error GoTo error_handler
    With Frmgrafica
        .LblBerma.Visible = False: .LblEjes.Visible = False: .LblMax.Visible = False: .LblMin.Visible = False
        .GrdGrafica.Visible = False: .LblYc.Visible = False: .LblTitulo.Visible = False: .PicGrafica.Visible = False
        .GrdMalla.Visible = False
    End With
    With FrmResultados
        .Enabled = True
        If caso1 = "malla" Then
            .CmdGrafica.SetFocus
        ElseIf caso1 = "graficar" Then
            .TxtGasto.SetFocus
        End If
    End With
    Exit Sub
error_handler:
'en caso de error salir de la subrutina
    Exit Sub
End Sub

```

## FrmResultados

```

Private Sub cmdblalock_Click()
'aquí se procede a obtener los valores de los tirantes criticos que existen en la sección para el ejercicio
    On Error GoTo error_handler
    .LblNqmax.Visible = False: .LblQmax.Visible = False: .LblNqmin.Visible = False: .LblQmin.Visible = False
    simple = True: Chaudhry = False

```

```

'aquí indico que el cálculo se hará sin considerar la variación del coeficiente n de Manning con respecto al tirante
GrdResultados.Visible = False: GrdResultados.Clear: LblResultado.Caption = "Blalock simplificado"
If Not continuare Resultados(TxtGasto.Text, TxtGravedad.Text) Then Exit Sub
Q = Val(TxtGasto.Text): g = Val(TxtGravedad.Text)
With GrdResultados
    .Clear: .Rows = 1: .Cols = 2: .ColWidth(0) = 750
    .ColWidth(1) = 1500: .ColAlignment(0) = 2: .ColAlignment(1) = 0
End With
ReDim nArea(total) As Double, nAreap(3) As Double
ReDim nAncho(total) As Double, nDAncho(total) As Double
ReDim nPerimetro(total) As Double, nPerimetrop(3) As Double, nDPerimetro(total) As Double
ReDim nRadio(total) As Double
ReDim nn(total) As Double, nFactor(total) As Double
ReDim ndn(total) As Double, nSigma1(total) As Double, nSigma2(total) As Double, nSigma3(total) As Double
ReDim nyc(total) As Double, ny(total) As Double
If procesos = 1 Then
'aquí obtengo el tirante para la sección central
    inicial = 1: nyc(1) = criticosencillo(inicial)
    GoSub presenta
Else
    If nymdif(2) > 0 Then inicial = nymdif(2) / 2
        nyc(1) = criticosencillo(inicial)
    End If
    With Me.GrdTem
        .Cols = 15: .Rows = 2: .Row = 0
        For z = 0 To .Cols - 1
            If z = 1 Or z = 0 Then
                .ColWidth(z) = 700
            ElseIf z = 7 Or z = 8 Or z = 9 Then
                .ColWidth(z) = 1500
            Else
                .ColWidth(z) = 1000
            End If
            .ColAlignment(z) = 2
        Next
        .Col = 0: .Text = "Proceso": .Col = 1: .Text = "Iteración": .Col = 2: .Text = "y(i)"
        .Col = 3: .Text = "A": .Col = 4: .Text = "P": .Col = 5: .Text = "Rh"
        .Col = 6: .Text = "K": .Col = 7: .Text = "sigma1": .Col = 8: .Text = "sigma2"
        .Col = 9: .Text = "sigma3": .Col = 10: .Text = "alfa": .Col = 11: .Text = "E"
        .Col = 12: .Text = "Froude": .Col = 13: .Text = "y(i+1)": .Col = 14: .Text = "Error"
        j = 1
        For i = 2 To procesos
'aquí inicia el proceso de cálculo para todos los intervalos subsecuentes, considerando los efectos de una sección
            'compuesta
            If i = procesos Then
'aquí se determina el tirante inicial para cada proceso
                nTirante = nymdif(i) * 1.5: seccionesproceso = total
            Else
                nTirante = (nymdif(i) + nymdif(i + 1)) / 2: seccionesproceso = encuentraseccionesproceso(nymdif(i))
            End If
            nerror = 1
            K = 1
            While nerror > 0.00001
                .Row = j: .Col = 0: .Text = I: .Col = 1: .Text = Str(K): .Col = 2: .Text = Str(redondea(nTirante))
                nAreaT = AreaTotal(seccionesproceso, nTirante)
                .Col = 3: .Text = Str(redondea(nAreaT))
                nAnchoT = AnchoTotal(seccionesproceso, nTirante)
                nDAnchoT = DAnchoTotal(seccionesproceso, nTirante)
                nPerimetroT = PerimetroTotal(seccionesproceso, nTirante)
                .Col = 4: .Text = Str(redondea(nPerimetroT))
                nDPerimetroT = DPerimetroTotal(seccionesproceso, nTirante)
            End While
        Next
    End With
End If

```

```

RadioTotal seccionesproceso
nRadioT = nAreaT / nPerimetroT
.Col = 5: .Text = Str(redondea(nRadioT))
nTotal seccionesproceso
nFactorT = FactorTotal(seccionesproceso)
.Col = 6: .Text = Str(redondea(nFactorT))
dnTotal seccionesproceso
nSigma1T = Sigma1Total(seccionesproceso)
.Col = 7: .Text = Str(redondea(nSigma1T)):
nSigma2T = Sigma2Total(seccionesproceso)
.Col = 8: .Text = Str(redondea(nSigma2T))
nSigma3T = Sigma3Total(seccionesproceso)
.Col = 9: .Text = Str(redondea(nSigma3T))
nAlfaT = coriolis(seccionesproceso)
.Col = 10: .Text = Str(redondea(nAlfaT))
nEnergia = nTirante + nAlfaT * Q ^ 2 / (2 * g * nAreaT ^ 2)
.Col = 11: .Text = Str(redondea(nEnergia))
nTirantesig = nEnergia - (nSigma2T / (nSigma2T * nSigma3T / nFactorT - nSigma1T))
nFroude = Sqr((nSigma2T * nSigma3T / nFactorT - nSigma1T) * (Q ^ 2 / 2 / g / nFactorT ^ 3))
.Col = 12: .Text = Str(redondea(nFroude)): .Col = 13: .Text = Str(redondea(nTirantesig))
nerror = Abs((nTirante - nTirantesig) / nTirante * 100)
.Col = 14: .Text = Str(redondea(nerror) * 100) & "%"
nTirante = nTirantesig
.Rows = .Rows + 1
j = j + 1: K = K + 1
If nTirante < nymdif(i) Then
    nerror = 0.00001: nyc(i) = 0
Else
    nyc(i) = nTirante
End If
Wend
If nyc(i - 1) > nymdif(i) Then nyc(i - 1) = 0
Next
End With

```

presenta:

```

Llenagridresult GrdResultados, procesos
lblnota.Caption = "Con las " & total & " secciones de este ejercicio, existen " & procesos & " intervalos en los " & _
"que pueden existir el mismo número de tirantes críticos" & Chr(13) & "Los tirantes críticos con valor cero " & _
"corresponden a los intervalos en los que no se presenta el regimen crítico"
If procesos > 1 Then
    CmdGrafica.Enabled = True: CmdGrid.Enabled = True: CmdGrid.SetFocus
Else
    Me.CmdGrafica.Enabled = True: Me.CmdGrafica.SetFocus
End If
If total = 3 And procesos = 2 And nym(2) = nym(3) Then
'aquí se calculan los gastos máximo y mínimo para el caso de una sección simétrica con una llanura de inundación a cada
'lado
    Qmax = redondea(gastomax()): FFmax = calculaFFmmax: Qmin = redondea(Qmax / FFmax)
    LblNqmax.Visible = True: LblQmax.Visible = True: LblNqmin.Visible = True
    LblQmin.Visible = True: LblNqmax.Caption = Str(Qmax) & " m^3/s": LblNqmin.Caption = Str(Qmin) & " m^3/s"
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Cmdcerrar_Click()
'al presionar el botón para cerrar, se descarga la forma
On Error GoTo error_handler

```

```

Unload Me
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub CmdChaudhry_Click()
'esta función obtiene los tirantes críticos con el método de Chaudhry
On Error GoTo error_handler
LblNqmax.Visible = False: LblQmax.Visible = False: LblNqmin.Visible = False: LblQmin.Visible = False
GrdResultados.Clear: GrdResultados.Visible = False: simple = True: Chaudhry = True
'aquí indico que no se considera la variación del coeficiente n de Manning con respecto del tirante
If Not continuare resultados(TxtGasto.Text, TxtGravedad.Text) Then
Exit Sub
End If
Q = Val(TxtGasto.Text): g = Val(TxtGravedad.Text)
ReDim nArea(3) As Double, nAreap(3) As Double
ReDim nAncho(3) As Double, nDAncho(total) As Double
ReDim nPerimetro(3) As Double, nPerimetrop(3) As Double, nDPerimetro(3) As Double
ReDim nRadio(3) As Double
ReDim nn(3) As Double, nFactor(3) As Double
ReDim ndn(3) As Double, nyc(3) As Double, ny(3) As Double
Dim c As Double
Cmax = calculaCmax()
With GrdResultados
.Clear: .Rows = 4: .Cols = 2: .ColWidth(0) = 750: .ColWidth(1) = 1500: .ColAlignment(0) = 2: .ColAlignment(1) = 0
End With
c = g * nAm ^ 3 / (Q ^ 2 * nTm)
inicial = nyndif(2) / 2
nyc(1) = criticosencillo(inicial)
If nyc(1) > nyndif(2) Then nyc(1) = 0
If c > Cmax Then nyc(2) = 0: nyc(3) = 0
If c > 1 And c < Cmax Then
nTirante = nyndif(2) + 0.00001: nyc(2) = encuentranyc(nTirante, c)
nTirante = nyCmax: nyc(3) = encuentranyc(nTirante, c)
End If
If c < 1 Then nTirante = nyCmax: nyc(2) = 0: nyc(3) = encuentranyc(nTirante, c)
Llenagridresult GrdResultados, 3: LblResultado.Caption = "Chaudhry"
lblnota.Caption = "Para este tipo de canal y empleando el método de Chaudhry-Bhallamudi, se presentan 3 tirantes " & _
"críticos posibles." & Chr(13) & _"Los tirantes críticos con valor cero corresponden a los intervalos en los que no se " & _
"presenta el regimen crítico"
'aquí se calculan los gastos máximo y mínimo para el caso de una sección simétrica con una llanura de inundación a cada
'lado
Qmax = redondea(gastomax()): Qmin = redondea(Sqr(g * nAm ^ 3 / (Cmax * nTm)))
LblNqmax.Visible = True: LblQmax.Visible = True: LblNqmin.Visible = True: LblQmin.Visible = True
LblNqmax.Caption = Str(Qmax) & " m^3/s": LblNqmin.Caption = Str(Qmin) & " m^3/s": CmdGrafica.Enabled = True
CmdGrid.Enabled = False
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub cmdfactor_Click()
'al presionar este botón se presenta la forma en la que se modifican los factores que afectan a los coeficientes de rugosidad de
'Manning
On Error GoTo error_handler
CmdGrafica.Enabled = False: CmdGrid.Enabled = False: frmfactor.Show
Exit Sub
error_handler:

```



```

'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Cmdgrafica_Click()
'aquí se procede a obtener los valores de los tirantes críticos que existen en la sección para el ejercicio
On Error GoTo error_handler
caso1 = "graficar"
Dim energiaprom As Double, energiadesv As Double, bandera As Boolean, j As Integer
Dim abcisamenor As Double, abcisamayor As Double, inicial As Integer, final As Integer
Dim abcisa As Double, xenergiamin As Double, xenergiamax As Double, proceso As Boolean
Dim ytirantemin As Double, ytirantemax As Double, bermas As Boolean
Determinapuntos: llenapuntos: bandera = False
If procesos > 1 Then
    abcisamenor = minimo(nymdif(2), nyc(1)): abcisamayor = maximo(mayorlista(nymdif), mayorlista(nyc))
End If
Do While bandera = False
    energiaprom = promedio(XEnergiatem1): energiadesv = desviacion(XEnergiatem1, energiaprom)
    If energiadesv < 1 And proceso = False Then Exit Do
    proceso = True
    limpiamayores energiaprom, XEnergiatem1, XEnergiatem, YTirantetem1, YTirantetem
    If energiaprom > energiadesv Then bandera = True
Loop
'aquí comparo con los datos para presentar todas las bermas y tirantes críticos
inicial = encuentra(YTirante, menorlista(YTirantetem1))
final = encuentra(YTirante, mayorlista(YTirantetem1))
inicial1 = inicial: final1 = final
If abcisamenor < menorlista(YTirantetem) And abcisamenor <> 0 Then
    inicial1 = encuentra(YTirante, abcisamenor) - 3
    If inicial1 < 1 Then inicial1 = 1
    'problemas hacia abajo
End If
If abcisamayor > mayorlista(YTirantetem) Then
    final1 = encuentra(YTirante, abcisamayor) + 3
    If final1 > encuentra(YTirante, mayorlista(YTirantetem)) Then
        final1 = encuentra(YTirante, mayorlista(YTirantetem))
    End If
    'problemas hacia arriba
End If
If Abs(inicial - inicial1) < 5 Then inicial = inicial1
If Abs(final - final1) < 5 Then final = final1
ajuste inicial, final
xenergiamin = menorlista(XEnergia): xenergiamax = mayorlista(XEnergia)
ytirantemin = menorlista(YTirante): ytirantemax = mayorlista(YTirante)
xenergiamin = xenergiamin - (xenergiamax - xenergiamin) / 10
bermas = False
For i = 1 To procesos
    If nymdif(i) <> 0 And nymdif(i) > ytirantemin And nymdif(i) < ytirantemax Then bermas = True
Next
With Frmgrafica
    .PicGrafica.Scale (xenergiamin, ytirantemax)-(xenergiamax, ytirantemin)
    .LblMin.Caption = "[" & redondea(xenergiamin) & ", " & redondea(ytirantemin) & "]"
    .LblMax.Caption = "[" & redondea(xenergiamax) & ", " & redondea(ytirantemax) & "]"
    .LblTitulo.Caption = "Energía vs. Tirante": .LblBerma.ForeColor = vbRed: .LblBerma.Caption = "Berma"
    If bermas Then
        .LblBerma.Visible = True
    Else
        .LblBerma.Visible = False
    End If
    .LblYc.ForeColor = vbBlue: .LblYc.Caption = "Tirante crítico"
End With

```

```

Frmgrafica.PicGrafica.Cls
For i = 1 To procesos
    Frmgrafica.PicGrafica.Line (xenergiamin, nymdif(i))-(xenergiamax, nymdif(i)), vbRed
    Frmgrafica.PicGrafica.Line (xenergiamin, nyc(i))-(xenergiamax, nyc(i)), vbBlue
Next
Frmgrafica.PicGrafica.Line (0, ytirantemin)-(0, ytirantemax), vbMagenta
Frmgrafica.PicGrafica.Line (xenergiamin, 0)-(xenergiamax, 0), vbMagenta
Frmgrafica.LblEjes.ForeColor = vbMagenta
Frmgrafica.LblEjes.Caption = "Eje de Energía=0"
If xenergiamin < 0 Or ytirantemin < 0 Then
    Frmgrafica.LblEjes.Visible = True
Else
    Frmgrafica.LblEjes.Visible = False
End If
'aquí se mete la línea de los tirantes críticos de Chaudhry
If Chaudhry Then
    Frmgrafica.PicGrafica.Line (xenergiamin, nyc(3))-(xenergiamax, nyc(3)), vbBlue
End If
Frmgrafica.PicGrafica.PSet (XEnergia(1), YTirante(1))
For i = 2 To UBound(XEnergia)
    Frmgrafica.PicGrafica.Line -(XEnergia(i), YTirante(i))
Next
With Frmgrafica.GrdGrafica
    .Cols = 2: .ColWidth(0) = 750: .ColWidth(1) = 750: .Rows = UBound(YTirante) + 1
    .Col = 0: .Row = 0: .Text = "Tirante"
    .Col = 1: .Text = "Energía"
    For j = 1 To UBound(YTirante)
        .Row = .Rows - j: .Col = 0: .CellForeColor = verificacolor(j)
        If .CellForeColor <> 0 Then .CellFontBold = True
        'aquí se mete el color para los valores de tirante según Chaudhry
        abcisa = YTirante(j)
        If Abs(Abs((Int(abcisa * 100) - abcisa * 100) * 100) - 99) < 1 Then
            abcisa = (Int(abcisa * 100) + 1) / 100
        End If
        .Text = Str(redondea(abcisa))
        .Col = 1: .CellForeColor = verificacolor(j)
        If .CellForeColor <> 0 Then .CellFontBold = True
        abcisa = YTirante(j)
        'aquí se mete el color para los valores de tirante según Chaudhry
        .Text = Str(redondea(XEnergia(j)))
    Next
End With
Frmgrafica.Show
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

Private Sub Cmdgrid_Click()
'con este botón se presenta la malla de iteraciones realizadas para obtener los resultados
On Error GoTo error_handler
caso1 = "malla"
copiagrid Frmgrafica.GrdMalla, FrmResultados.GrdTem
Frmgrafica.Show
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub
End Sub

```

```

Private Sub CmdResultados_Click()
'aquí se procede a obtener los valores de los tirantes críticos que existen en la sección para el ejercicio
On Error GoTo error_handler
LblNqmax.Visible = False: LblQmax.Visible = False: LblNqmin.Visible = False: LblQmin.Visible = False
GrdResultados.Visible = False: GrdResultados.Clear: LblResultado.Caption = "Blalock completo"
simple = False: Chaudhry = False
'aquí indico que el cálculo se hará considerando la variación del coeficiente n de Manning con respecto al tirante
If Not continuareresultados(TxtGasto.Text, TxtGravedad.Text) Then
Exit Sub
End If
Q = Val(TxtGasto.Text)
g = Val(TxtGravedad.Text)
With GrdResultados
.Clear: .Rows = 1: .Cols = 2: .ColWidth(0) = 750: .ColWidth(1) = 1500: .ColAlignment(0) = 2: .ColAlignment(1) = 0
End With
ReDim nArea(total) As Double, nAreap(3) As Double
ReDim nAncho(total) As Double, nDAncho(total) As Double
ReDim nPerimetro(total) As Double, nPerimetrop(3) As Double, nDPerimetro(total) As Double
ReDim nRadio(total) As Double
ReDim nn(total) As Double, nFactor(total) As Double
ReDim ndn(total) As Double, nSigma1(total) As Double, nSigma2(total) As Double, nSigma3(total) As Double
ReDim nyc(total) As Double, ny(total) As Double
If procesos = 1 Then
'aquí obtengo el tirante para la sección central
inicial = 1: nyc(1) = criticosencillo(inicial): GoSub presenta
Else
If nymdif(2) > 0 Then inicial = nymdif(2) / 2
nyc(1) = criticosencillo(inicial)
End If
With Me.GrdTem
.Cols = 15: .Rows = 2: .Row = 0
For z = 0 To .Cols - 1
If z = 1 Or z = 0 Then
.ColWidth(z) = 700
ElseIf z = 7 Or z = 8 Or z = 9 Then
.ColWidth(z) = 1500
Else
.ColWidth(z) = 1000
End If
.ColAlignment(z) = 2
Next
.Col = 0: .Text = "Proceso": .Col = 1: .Text = "Iteración": .Col = 2: .Text = "y(i)"
.Col = 3: .Text = "A": .Col = 4: .Text = "P": .Col = 5: .Text = "Rh": .Col = 6: .Text = "K"
.Col = 7: .Text = "sigma1": .Col = 8: .Text = "sigma2": .Col = 9: .Text = "sigma3"
.Col = 10: .Text = "alfa": .Col = 11: .Text = "E": .Col = 12: .Text = "Froude"
.Col = 13: .Text = "y(i+1)": .Col = 14: .Text = "Error"
j = 1
For i = 2 To procesos
'aquí inicia el proceso de cálculo para todos los intervalos subsecuentes, considerando los efectos de una sección
compuesta
If i = procesos Then
'aquí se determina el tirante inicial para cada proceso
nTirante = nymdif(i) * 1.5: seccionesproceso = total
Else
nTirante = (nymdif(i) + nymdif(i + 1)) / 2: seccionesproceso = encuentraseccionesproceso(nymdif(i))
End If
nerror = 1
K = 1
While nerror > 0.00001
.Row = j: .Col = 0: .Text = I: .Col = 1: .Text = Str(K): .Col = 2: .Text = Str(redondea(nTirante))
nAreaT = AreaTotal(seccionesproceso, nTirante)

```

```

.Col = 3: .Text = Str(redondea(nAreaT))
nAnchoT = AnchoTotal(seccionesproceso, nTirante)
nDAnchoT = DAnchoTotal(seccionesproceso, nTirante)
nPerimetroT = PerimetroTotal(seccionesproceso, nTirante)
.Col = 4: .Text = Str(redondea(nPerimetroT))
nDPerimetroT = DPerimetroTotal(seccionesproceso, nTirante)
RadioTotal seccionesproceso
nRadioT = nAreaT / nPerimetroT
.Col = 5: .Text = Str(redondea(nRadioT))
nTotal seccionesproceso
nFactorT = FactorTotal(seccionesproceso)
.Col = 6: .Text = Str(redondea(nFactorT))
dnTotal seccionesproceso
nSigma1T = Sigma1Total(seccionesproceso)
.Col = 7: .Text = Str(redondea(nSigma1T))
nSigma2T = Sigma2Total(seccionesproceso)
.Col = 8: .Text = Str(redondea(nSigma2T))
nSigma3T = Sigma3Total(seccionesproceso)
.Col = 9: .Text = Str(redondea(nSigma3T))
nAlfaT = coriolis(seccionesproceso)
.Col = 10: .Text = Str(redondea(nAlfaT))
nEnergia = nTirante + nAlfaT * Q ^ 2 / (2 * g * nAreaT ^ 2)
.Col = 11: .Text = Str(redondea(nEnergia))
nTirantesig = nEnergia - (nSigma2T / (nSigma2T * nSigma3T / nFactorT - nSigma1T))
nFroude = Sqr((nSigma2T * nSigma3T / nFactorT - nSigma1T) * (Q ^ 2 / 2 / g / nFactorT ^ 3))
.Col = 12: .Text = Str(redondea(nFroude)): .Col = 13: .Text = Str(redondea(nTirantesig))
nerror = Abs((nTirante - nTirantesig) / nTirante * 100)
.Col = 14: .Text = Str(redondea(nerror) * 100) & "%"
nTirante = nTirantesig
.Rows = .Rows + 1: j = j + 1: K = K + 1
If nTirante < nymdif(i) Then
    nerror = 0.00001: nyc(i) = 0
Else
    nyc(i) = nTirante
End If
Wend
If nyc(i - 1) > nymdif(i) Then nyc(i - 1) = 0
Next
End With
presenta:
Llenagridresult GrdResultados, procesos
lblnota.Caption = "Con las " & total & " secciones de este ejercicio, existen " & procesos & " intervalos en los " & _
"que pueden existir el mismo número de tirantes críticos" & Chr(13) & "Los tirantes críticos con valor cero " & _
"corresponden a los intervalos en los que no se presenta el regimen crítico"
If procesos > 1 Then
    CmdGrafica.Enabled = True: CmdGrid.Enabled = True: CmdGrid.SetFocus
Else
    Me.CmdGrafica.Enabled = True: Me.CmdGrafica.SetFocus
End If
If total = 3 And procesos = 2 And nym(2) = nym(3) Then
'aquí se calculan los gastos máximo y mínimo para el caso de una sección simétrica con una llanura de inundación a cada
'lado
    Qmax = redondea(gastomax())
    FFmax = calculaFFmmax: Qmin = redondea(Qmax / FFmax)
    LblNqmax.Visible = True: LblQmax.Visible = True: LblNqmin.Visible = True
    LblQmin.Visible = True: LblNqmax.Caption = Str(Qmax) & " m^3/s": LblNqmin.Caption = Str(Qmin) & " m^3/s"
End If
Exit Sub
error_handler:
'en caso de error salir de la subrutina
Exit Sub

```

End Sub

Private Sub Form\_Load()

'al cargar esta forma, se pasan los valores presentes en la malla de datos a variables que permiten realizar los cálculos

On Error GoTo error\_handler

accesaarchivo ubicacion

Me.Caption = "Resultados para " & Mid(archivoenuso, 1, Len(archivoenuso) - 4)

Me.Left = 500: Me.Top = 500: Me.Height = 5000: Me.Width = 8000

lblnota.Caption = "": CmdGrafica.Enabled = False: CmdGrid.Enabled = False: CmdChaudhry.Enabled = False

llenavariabes FrmDatos.Grddatos: procesos = cuentaprosesos

If procesos = 1 Then Me.CmdFactor.Enabled = False: Me.CmdGrid.Enabled = False

If Chaudhrys() Then CmdChaudhry.Enabled = True

LblNqmax.Visible = False: LblQmax.Visible = False: LblNqmin.Visible = False: LblQmin.Visible = False

Exit Sub

error\_handler:

'en caso de error salir de la subrutina

Exit Sub

End Sub

Private Sub Form\_Unload(Cancel As Integer)

'al descargar la forma se regresa a la forma con los datos geométricos de la sección

On Error GoTo error\_handler

MDIPrincipal.MnuArchivo.Enabled = True: MDIPrincipal.MnuCalculo.Enabled = True

MDIPrincipal.MnuResultados.Enabled = True: FrmDatos.Enabled = True

Exit Sub

error\_handler:

'en caso de error salir de la subrutina

Exit Sub

End Sub

Private Sub txtgasto\_Change()

'en caso de que se cambie el gasto, se desactiva el botón que muestra gráfica y el que muestra la malla de las iteraciones

On Error GoTo error\_handler

CmdGrafica.Enabled = False: CmdGrid.Enabled = False

Exit Sub

error\_handler:

'en caso de error salir de la subrutina

Exit Sub

End Sub

Private Sub txtgravedad\_Change()

'en caso de que se cambie la gravedad, se desactiva el botón que muestra gráfica y el que muestra la malla de las iteraciones

On Error GoTo error\_handler

CmdGrafica.Enabled = False: CmdGrid.Enabled = False

Exit Sub

error\_handler:

'en caso de error salir de la subrutina

Exit Sub

End Sub

## Mod\_archivo

Function configurado() As Boolean

'esta función verifica que el sistema este configurado

On Error GoTo error\_handler

Open App.Path & "\sistema\tirantes.ini" For Input As #1: 'abre el archivo tirantes.ini para leer la configuración

Line Input #1, renglon1: 'lee el primer renglon, es la dirección del archivo base

Line Input #1, renglon2: 'lee el segundo renglon, es el nombre del archivo base

Line Input #1, renglon3: 'lee el tercer renglon, es la dirección y nombre del archivo base

Close #1: 'cierra el archivo

'aquí debe de conectarse al archivo base

```

Set accesodbase = OpenDatabase(renglon1, dbDriverComplete, False, "dBASE IV;")
'se define el directorio de conexion a dbase
Set rsdbase = accesodbase.OpenRecordset(Mid(renglon2, 1, Len(renglon2) - 4), dbOpenTable)
'se abre el archivo base
If rsdbase.Fields.Count < 10 Then
    malconfigurado = "El archivo que seleccionó no tiene las características necesarias" & Chr(13) & _
        "por favor seleccione el archivo correcto"
    configurado = False
    Exit Function
End If
configurado = True: ' Si todo salió bien confirmo que no hay problema
accesodbase.Close: 'Hay que terminar todo lo que se empieze
Exit Function
error_handler:
    configurado = False: ' Esto quiere decir que no se ha configurado el sistema
    Exit Function
End Function

Function configurar(dir As DirListBox, file As FileListBox, texto As TextBox) As Boolean
'esta función configura el sistema
    On Error GoTo error_handler: 'por si ocurre un error
    If Len(Trim(file.FileName)) = 0 And Len(texto.Text) = 0 Then
        A = MsgBox("El nombre del archivo es incorrecto." & (Chr(13)) & Error, vbOKOnly, "Mensaje")
        Screen.MousePointer = vbDefault: Exit Function
    End If
    'si no se elige el archivo mandara este mensaje de error y salimos de aqui
    archivo = file.FileName
    If Len(archivo) = 0 Then archivo = texto.Text: 'paso el valor del archivo a la variable <archivo>
    resp = MsgBox("Se establecerá como archivo base el siguiente: " & (Chr(13)) & (Chr(13)) & dir.Path & "\" & archivo _
        & (Chr(13)) & (Chr(13)) & "¿Está seguro?", vbYesNo + vbInformation + vbDefaultButton2, "!Atención!")
    'se confirma si se desea continuar
    If resp = vbNo Then Screen.MousePointer = vbDefault: Exit Function: 'no se desea continuar asi que bye bye
    Open App.Path & "\sistema\tirantes.ini" For Output As #1
    Print #1, dir.Path: Print #1, archivo: Print #1, dir.Path & "\" & archivo: Close #1
    'abre el archivo <tirantes.ini> para escritura y escribe la ruta sola en un renglón, el nombre del archivo en el segundo
    'renglón y ambos datos concatenados en el tercer renglón
    If configurado Then
        With MDIPrincipal
            .MnuArchivo.Enabled = True: .MnuNuevo.Enabled = True: .MnuAbrir.Enabled = True
            .MnuGuardar.Enabled = False: .MnuGuardarComo.Enabled = False: .MnuConfigura.Enabled = True
            .MnuCalculo.Enabled = False
        End With
    Else
        If malconfigurado = "" Then
            A = MsgBox("No se puede abrir el archivo", vbOKOnly, "!Atención!")
        Else
            A = MsgBox(malconfigurado, vbOKOnly, "!Atención!")
        End If
        malconfigurado = "": Exit Function
    End If: 'al configurar y poder leer el archivo, habilito las opciones de menú de Nuevo, Abrir y Configurar
    configurar = True: 'al no haber problemas doy el valor positivo a la función
    Exit Function
error_handler:
    Screen.MousePointer = vbDefault
    A = MsgBox("Ocurrió un error." & (Chr(13)) & _
        archivo & (Chr(13)) & ". ERROR MESSAGE: " & Error, vbOKOnly, "Mensaje")
    Close #1: 'si hubo error hay que cerrar el archivo: Exit Function
End Function

Function crea(dir As DirListBox, file As TextBox) As Boolean
'esta función crea un archivo dado en la dirección indicada por el directorio

```

```

On Error GoTo error_handler
If archivoexiste(dir.Path & "\" & file.Text) Then
'verificamos si ya existe el archivo que se va a crear
    mensaje = MsgBox("El archivo ya existe, ¿Desea sobrescribir?" & Chr(13) & Chr(13) & _
        "Recuerde que se borrará el archivo: " & file.Text, vbInformation + vbYesNo, "¡Atención!")
    If mensaje = vbNo Then
        crea = False: Exit Function
    End If: 'si no se quiere sobrescribir se sale de aquí
End If
ubicacion = dir.Path & "\" & file.Text:
'aqui establezco cual es la direccion del archivo que acabo de crear y que será el archivo en uso
FileCopy renglon3, ubicacion: 'aquí se crea una copia limpia del archivo nuevo
If accesaarchivo(ubicacion) Then
'verifico que se puede acceder al archivo
    crea = True: ' si todo sale bien, asigno valor verdadero a la función
Else
    crea = False: 'si falla el intento pues no se puede dar valor verdadero
End If
Exit Function
error_handler:
Exit Function: 'si falla me salgo y ya: crea = False
End Function

```

```

Function destruye(Dir1 As DirListBox, file As TextBox) As Boolean
'esta función borra un archivo dado en la dirección indicada por el directorio
    On Error GoTo error_handler
    If archivoexiste(Dir1.Path & "\" & file.Text) Then
'verificamos si ya existe el archivo que se va a borrar
        ubicacion = Dir1.Path & "\" & file.Text: 'aqui establezco cual es la direccion del archivo que quiero borrar
        Kill ubicacion: 'aquí se destruye el mencionado archivo: destruye = True
    End If
    Exit Function
error_handler:
'en caso de error
    destruye = False: Exit Function
End Function

```

```

Function hallacarpeta(sistema As String, datos As String) As Boolean
'esta función busca una carpeta especifica, si no existe la crea, ya que es necesaria para el sistema las variables representan, un
'carpeta (con el path) y una subcarpeta de esta última, también con todo y el path
    Dim objcarpsis As Object, carpeta, subcarpeta, nueva
'declaro las variables de objeto de archivo y de texto que necesito
    On Error GoTo ErrorHandler
    ' si falla en encontrar la carpeta que busco, pues se va a otra rutina donde la crea
    Set objcarpsis = CreateObject("Scripting.FileSystemObject")
'declaro esta variable como un objeto archivo de sistema
    Set carpeta = objcarpsis.getfolder(datos)
'busco la subcarpeta de datos, donde voy a almacenar los archivos de trabajo
    If carpeta <> "" Then hallacarpeta = True: 'si la encuentra, todo bien
    Exit Function
ErrorHandler:
'en caso de que no encontrar la carpeta indicada se procede a crearla
    Set carpeta = objcarpsis.getfolder(sistema): 'localizo la carpeta del sistema, y donde se creará la subcarpeta de datos
    Set subcarpeta = carpeta.subfolders: 'indico el procedimiento de crear un subcarpeta
    Set nueva = subcarpeta.Add("datos"): 'creo la carpeta nueva
    hallacarpeta = True: 'ahora sí, ya existe la carpeta de datos
    Err.Clear
    Exit Function
End Function

```

```

Function archivoexiste(archivo As String) As Boolean

```

```
'con esta función busco un archivo para verificar si existe o no antes de crear uno para evitar sobrescribir en un archivo que
'sea necesario la variable nuevo indica el nombre de un archivo con todo y ubicación del sistema
Dim objarchsis As Object, verifica: 'declaro la variable como un objeto de sistema
On Error GoTo error_handler: 'en este caso, si no se encuentra el archivo, el valor de la función es negativo
Set objarchsis = CreateObject("Scripting.FileSystemObject"): 'declaro esta variable como un objeto archivo de sistema
Set verifica = objarchsis.getfile(archivo): 'busco el archivo, en caso de que se encuentre, doy el valor verdadero a la función
archivoexiste = True
Exit Function
error_handler:
'si no se encuentra el archivo, entonces sólo asigno valor falso a la función
Err.Clear: archivoexiste = False
Exit Function
End Function
```

```
Function accesaarchivo(ubicacion As String) As Boolean
'aquí debe de conectarse al archivo en uso
On Error GoTo error_handler
A = 0
While Not (B = "\")
c = Len(ubicacion) - A: B = Mid(ubicacion, Len(ubicacion) - A, 1)
carpetaenuso = Left(ubicacion, c - 1): archivoenuso = Right(ubicacion, Len(ubicacion) - Len(carpetanuso) - 1)
A = A + 1
Wend
Set accesodbase = OpenDatabase(carpetanuso, dbDriverComplete, False, "dBASE IV;")
'se define el directorio de conexión a dbase
Set rsdbase = accesodbase.OpenRecordset(Mid(archivoenuso, 1, Len(archivoenuso) - 4), dbOpenTable)
'se abre el archivo en uso
acesaarchivo = True: 'Si todo salió bien confirmo que no hay problema
If rsdbase.Fields.Count < 10 Then
acesaarchivo = False: Exit Function
End If
accesodbase.Close: 'Hay que terminar todo lo que se empieza: Exit Function
error_handler:
acesaarchivo = False: 'Esto quiere decir que no se ha podido acceder al archivo en uso:
Exit Function
End Function
```

```
Function actualiza(grid As MSFlexGrid, ubicacion As String) As Boolean
'Esta función guarda los datos del grid en un archivo determinado
On Error GoTo error_handler
acesaarchivo ubicación: 'con la ubicación obtengo el archivoenuso y la carpetaenuso
FileCopy renglon3, ubicación: 'elimino el archivo y creo uno nuevo vacío
Set accesodbase = OpenDatabase(carpetanuso, dbDriverComplete, False, "dBASE IV;")
'se define el directorio de conexión a dbase
Set rsdbase = accesodbase.OpenRecordset(Mid(archivoenuso, 1, Len(archivoenuso) - 4), dbOpenTable)
'se abre el archivo en uso
For i = 1 To grid.Rows - 1
'recorro todo el grid, renglón por renglón, sin el encabezado
grid.Row = i: rsdbase.AddNew: 'agrego un registro al recordset
For j = 0 To grid.Cols - 1
grid.Col = j
If j = indicatipo Then
If grid.Text = "central" Then
rsdbase.Fields(j) = 0
ElseIf grid.Text = "izquierda" Then
rsdbase.Fields(j) = 1
ElseIf grid.Text = "derecha" Then
rsdbase.Fields(j) = 2
Else
rsdbase.Fields(j) = Mid(grid.Text, 1, 1)
End If
End For
End For
```



```

        Else
            If grid.Text = "" Then
                rsdbase.Fields(j) = Null
            Else
                rsdbase.Fields(j) = grid.Text
            End If
        End If
        'agrego un campo al registro del recordset abierto
    Next
    rsdbase.Update: 'actualizo los datos en el archivo
Next
rsdbase.Close: 'cierro el recordset: actualiza = True
Exit Function
error_handler:
    actualiza = False
    Exit Function
End Function

Function guarda(grid As MSFlexGrid, ubicacion As String) As Boolean
'Esta función guarda los datos del grid en un archivo determinado
    On Error GoTo error_handler
    accesaarchivo ubicacion
    'con la ubicacion obtengo el archivoenuso y la carpetaenuso
    Set accesodbase = OpenDatabase(carpetaanuso, dbDriverComplete, False, "dBASE IV;")
    'se define el directorio de conexión a dbase
    Set rsdbase = accesodbase.OpenRecordset(Mid(archivoenuso, 1, Len(archivoenuso) - 4), dbOpenTable)
    'se abre el archivo en uso
    For i = 1 To grid.Rows - 1
        'recorro todo el grid, renglón por renglón, sin el encabezado
        grid.Row = I: rsdbase.AddNew
        'agrego un registro al recordset
        For j = 0 To grid.Cols - 1
            grid.Col = j
            If j = indicatipo Then
                If grid.Text = "central" Then
                    rsdbase.Fields(j) = 0
                ElseIf grid.Text = "izquierda" Then
                    rsdbase.Fields(j) = 1
                ElseIf grid.Text = "derecha" Then
                    rsdbase.Fields(j) = 2
                Else
                    rsdbase.Fields(j) = Mid(grid.Text, 1, 1)
                End If
            ElseIf grid.Text = "" Then
                rsdbase.Fields(j) = Null
            ElseIf Not (grid.Text = "") Then
                rsdbase.Fields(j) = grid.Text
            End If
            'agrego un campo al registro del recordset abierto
        Next
        rsdbase.Update: 'actualizo los datos en el archivo
    Next
    rsdbase.Close: 'cierro el recordset: guarda = True
    Exit Function
error_handler:
    'en caso de error
    guarda = False
    Exit Function
End Function

Function verificanombre(cadena As String) As String

```

'esta función revisa que los caracteres contenidos en una cadena no contengan vocales acentuadas, y las cambia por vocales sin 'acento

```
On Error GoTo error_handler
verificanombre = ""
For i = 1 To Len(cadena)
    caracter = Mid(cadena, i, 1)
    Select Case caracter
        Case "Á"
            caracter = "A"
        Case "á"
            caracter = "a"
        Case "É"
            caracter = "E"
        Case "é"
            caracter = "e"
        Case "Í"
            caracter = "I"
        Case "í"
            caracter = "i"
        Case "Ó"
            caracter = "O"
        Case "ó"
            caracter = "o"
        Case "Ú"
            caracter = "U"
        Case "ú"
            caracter = "u"
        Case "*"
            caracter = ""
    End Select
    verificanombre = verificanombre + caracter
Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function
```

## Mod\_datos

```
Function llenagrid(grid As MSFlexGrid, ubicacion As String, Forma As Form) As String
'aquí debe de conectarse al archivo en uso y vaciar los datos en el grid de datos
On Error GoTo error_handler
Set accesodbase = OpenDatabase(carpetaaenuso, dbDriverComplete, False, "dBASE IV;")
'se define el directorio de conexión a dbase
Set rsdbase = accesodbase.OpenRecordset(Mid(archivoenuso, 1, Len(archivoenuso) - 4), dbOpenTable)
'se abre el archivo en uso
grid.Cols = rsdbase.Fields.Count: grid.Rows = rsdbase.RecordCount + 1: 'dimensiono el grid
If rsdbase.RecordCount = 0 Then
    llenagrid = "vacío": grid.Row = 0
    'voy la fila de los encabezados
    For j = 0 To grid.Cols - 1
        grid.Col = j: ' y columna por columna
        grid.Text = campos(j)
        If campos(j) = "Tipo" Then indicatipo = j
        grid.ColAlignment(j) = flexAlignCenterCenter
    Next
    Exit Function
End If
rsdbase.MoveFirst: 'me pongo en el inicio
For i = 0 To grid.Rows - 1
```

```

grid.Row = I: 'voy fila por fila
For j = 0 To grid.Cols - 1
    grid.Col = j: 'y columna por columna
    If i = 0 Then
        grid.Text = campos(j): grid.ColAlignment(j) = flexAlignCenterCenter
        If campos(j) = "Tipo" Then indicatipo = j: 'en caso de que sea el encabezado, lo cargo desde la variable
    Else
        If j = indicatipo Then
            'el tipo de sección se guarda en el archivo de formato DBASE con los valores de 0,1 y 2 que corresponde a
            'central o laterales, aquí se modifica el caso
            If rsdbase(j) = 0 Or rsdbase(j) = 1 Or rsdbase(j) = 2 Then
                grid.Text = tipo(rsdbase.Fields(j))
            Else
                grid.Text = ""
            End If
        Else
            If IsNull(rsdbase.Fields(j)) Then
                'en caso de que el campo sea nulo, se carga una cadena vacía
                grid.Text = ""
            Else
                grid.Text = rsdbase.Fields(j)
            End If
        End If
        grid.ColAlignment(j) = flexAlignCenterCenter: 'en caso de que sea la malla de datos los obtengo del recordset
    End If
Next
If i > 0 Then rsdbase.MoveNext
'sólo si no es el encabezado, me muevo al siguiente registro
Next
rsdbase.Close: 'cierro el recordset
llenagrid = "lleno"
Exit Function
error_handler:
'en caso de error
llenagrid = "falla"
Exit Function
End Function

Function titulos(archivo As String) As Boolean
'aquí debe de conectarse al archivo de datos de los campos y vaciar los nombres de campos en el los encabezados del grid de
'datos
On Error GoTo error_handler
If configurado Then
'abro el archivo ini para saber donde esta la configuración
Set accesodbase = OpenDatabase(renglon1, dbDriverComplete, False, "dBASE IV;")
'se define el directorio de conexión a dbase para los encabezados
Set rsdbase = accesodbase.OpenRecordset(archivo, dbOpenTable): 'se abre el archivo en uso
ReDim campos(rsdbase.RecordCount): 'redimensiono la variable que contiene los nombres de campos
For i = 0 To rsdbase.RecordCount - 1
    campos(i) = rsdbase.Fields(1): 'vacío el nombre de los campos a la variable
    rsdbase.MoveNext
Next
rsdbase.Close: 'cierro el recordset
End If
titulos = True: 'doy el valor positivo a la función
Exit Function
error_handler:
'en caso de error
titulos = False
Exit Function
End Function

```

```

Function llenadatosforma(grid As MSFlexGrid, Forma As Form, i As Integer) As Boolean
'esta funcion llena los datos desde el grid en la forma de edición de registros esto es así por que se puede tener mejor control
'sobre las características de cada sección se encesan los parámetros de: el grid, la forma donde se cargan los datos y el
'renglón
'del grid de donde se van a sacar los datos
  On Error GoTo error_handler
  With grid
    .Row = i: 'me ubico en el renglon enviado y voy recorriendo columna por columna
    .Col = 0: If .Text = "" Then .Text = i
    Forma.Caption = "Sección " & .Text
    .Col = 1: Forma.TxtB.Text = .Text
    .Col = 2: Forma.TxtKi.Text = .Text
    .Col = 3: Forma.TxtKd.Text = .Text
    .Col = 4: Forma.TxtYm.Text = .Text
    .Col = 5: Forma.CmbTipo.Text = .Text
    .Col = 6: Forma.TxtAlfa.Text = .Text
    .Col = 7: Forma.TxtC.Text = .Text
    .Col = 8: Forma.TxtAlfan.Text = .Text
    .Col = 9: Forma.TxtKs.Text = .Text
    .Col = 10: Forma.TxtN.Text = .Text
  End With
  'si no encuentra problemas, la función vale verdadero
  llenadatosforma = True
  Exit Function
error_handler:
  llenadatosforma = False
  Exit Function: 'si algo falla la función vale falso
End Function
Function verificatipo(grid As MSFlexGrid) As Boolean
'esta función verifica si ya existe una sección central en un grid dado eso es con la intención de no permitir dos secciones
'centrales, no es lógico
  On Error GoTo error_handler
  central = False
  'esta es una variable global que usaré en adelante para saber si ya se determinó la sección central del caso a revisar y toma
  'valor verdadero, falso si no hay sección determinada aún
  seccioncentral = 0: 'esta variable global determina la fila en la que se encuentra la sección central
  cuentacentral = 0
  'esta variable global contiene el número de secciones centrales que existen en un ejemplo a analizar. Es para el caso de
  'archivos generados fuera del sistema
  If grid.Rows > 1 Then
    'si ya existen datos verifica donde está la sección central si es que existe
    With grid
      For i = 1 To .Rows - 1
        .Row = i: .Col = 5: 'es en la columna 5 donde se encuentra el tipo de sección
        If .Text = "central" Then
          central = True: .Col = 0: seccioncentral = .Text: cuentacentral = cuentacentral + 1
        End If
      Next
    End With
  Else
    'si no existe ningún registro, evidentemente no habrá sección central determinada
    central = False
  End If
  verificatipo = True
  Exit Function
error_handler:
'en caso de error
  verificatipo = False
  Exit Function
End Function

```

```

Function verificaberma(grid As MSFlexGrid, cseccion As String, ctipo As String, cym As String) As Boolean
'esta función determina si de acuerdo a los datos proporcionados, ya existe alguna sección lateral con la misma altura de berma
'del mismo lado, ya que ésto provoca errores en el cálculo
Static vseccion As String, vtipo As String, vym As String
On Error GoTo error_handler
verificaberma = False
With grid
If .Rows > 1 Then
'si se tienen más de un renglón en el grid
For i = 1 To .Rows - 1
'recorremos todas las filas
.Row = i.Col = 0: vseccion = .Text: 'de la columna uno, tomamos el número de sección
.Col = 5: vtipo = .Text: 'de la columna 5 tomamos el tipo de sección
.Col = 4: vym = .Text: 'de la columna 4 se toma la altura de berma
If vtipo = ctipo And vym = cym And Not (cseccion = vseccion) Then
'si el tipo es el mismo (del mismo lado), la altura de berma es igual y no se trata de la misma sección, la
'función toma el valor verdadero
verificaberma = True: Exit Function: 'y nos salimos de la función
End If
Next
Else
'si no hay datos, pues esta función toma el valor de falso, ya que
verificaberma = False
End If
End With
Exit Function
error_handler:
'en caso de error
verificaberma = False
Exit Function
End Function

```

```

Function verificadatos(cseccion As String, ctipo As String, cB As String, cki As String, ckd As String, cym As String, calfa As _
String, cc As String, calfan As String, cks As String, grid As MSFlexGrid, cn As String) As String
'con esta función verifico que los datos ingresados para una sección son correctos, además que no se dupliquen las secciones
'centrales. por eso necesito todos los parametros, además del grid donde se encuentran los datos completos del ejemplo
On Error GoTo error_handler
verificadatos = ""
'aquí el valor es cadena vacía en caso de que haya algun error, la función toma el valor
If verificatipo(grid) Then
'aquí llamo a la función que me dice si ya hay secciones centrales y en que fila está
If verificaberma(grid, cseccion, ctipo, cym) Then
'verifico si ya existe una sección del mismo lado con la misma altura de berma
verificadatos = "Ya existe una sección " & ctipo & ", con esa altura de berma" & _
Chr(13) & "no puede haber dos secciones " & ctipo & "s con estas condiciones"
Exit Function
End If
If ctipo = "central" And central = True And Trim(Str(seccioncentral)) <> Trim(cseccion) Then
'si existe ya la sección central y no está en fila la que estamos revisando marca el error
verificadatos = "Ya existe una sección central," & Chr(13) & "no puede haber dos secciones de este tipo"
Exit Function
ElseIf ctipo = "central" And Not IsNumeric(cki) Then
'en el caso de la sección central, el talud izquierdo debe ser un número
verificadatos = "El talud izquierdo debe de ser un número"
Exit Function
ElseIf ctipo = "central" And Val(cki) < 0 Then
'en el caso de la sección central, no debe de tener un talud izquierdo con valor negativo
verificadatos = "El talud izquierdo es negativo," & Chr(13) & "debe de ser al menos igual a cero"
Exit Function
ElseIf ctipo = "central" And Not IsNumeric(ckd) Then

```

```

'en el caso de la sección central, el talud derecho debe ser un número
    verificadatos = "El talud derecho debe de ser un número"
    Exit Function
ElseIf ctipo = "central" And Val(ckd) < 0 Then
'en el caso de la sección central, no debe de tener un talud derecho con valor negativo
    verificadatos = "El talud derecho es negativo," & Chr(13) & "debe de ser al menos igual a cero"
    Exit Function
End If
If ctipo = "izquierda" And ckd <> "" Then
'al ser sección izquierda revisa que no exista talud derecho
    verificadatos = "La sección es izquierda," & Chr(13) & "no puede tener talud derecho"
    Exit Function
ElseIf ctipo = "izquierda" And Not IsNumeric(cki) Then
'en el caso de la sección izquierda, el talud izquierdo debe ser un número
    verificadatos = "El talud izquierdo debe de ser un número"
    Exit Function
ElseIf ctipo = "izquierda" And Val(cki) < 0 Then
'en el caso de la sección izquierda, no debe de tener un talud izquierdo con valor negativo
    verificadatos = "El talud izquierdo es negativo," & Chr(13) & "debe de ser al menos igual a cero"
    Exit Function
End If
If ctipo = "derecha" And cki <> "" Then
'al ser sección derecha revisa que no exista talud izquierdo
    verificadatos = "La sección es derecha," & Chr(13) & "no puede tener talud izquierdo"
    Exit Function
ElseIf ctipo = "derecha" And Not IsNumeric(ckd) Then
'en el caso de la sección derecha, el talud derecho debe ser un número
    verificadatos = "El talud derecho debe ser un número"
    Exit Function
ElseIf ctipo = "derecha" And Val(ckd) < 0 Then
'en el caso de la sección derecha, no debe de tener un talud derecho con valor negativo
    verificadatos = "El talud derecho es negativo," & Chr(13) & "debe de ser al menos igual a cero"
    Exit Function
End If
If Not IsNumeric(cB) Then
'verifica que el ancho de pantalla sea un número
    verificadatos = "El ancho de pantalla debe ser un número"
    Exit Function
ElseIf Val(cB) < 0 Then
'verifica que el ancho de pantalla no sea negativo en cada sección
    verificadatos = "El ancho de pantalla es negativo," & Chr(13) & "debe de ser al menos igual a cero"
    Exit Function
End If
If cB = "" Then
'verifica que el ancho de pantalla no sea cadena vacía en cada sección
    verificadatos = "El ancho de pantalla está vacío," & Chr(13) & "debe de ser al menos igual a cero"
    Exit Function
End If
If (ctipo = "izquierda" Or ctipo = "derecha") And (Val(cym) <= 0 Or Not IsNumeric(cym)) Then
'verifica que la altura de la berma sea positiva en cada sección
    verificadatos = "La altura de berma es negativa o cero," & Chr(13) & "debe de ser mayor que cero"
    Exit Function
ElseIf (ctipo = "izquierda" Or ctipo = "derecha") And (Val(cym) <= 0 Or Not IsNumeric(cym)) Then
'verifica que la altura de la berma sea positiva en cada sección
    verificadatos = "La altura de berma es negativa o cero," & Chr(13) & "debe de ser mayor que cero"
    Exit Function
End If
If Val(calfa) <= 0 Or Not IsNumeric(calfa) Then
'verifica que el coeficiente de coriolis sea positivo en cada sección
    verificadatos = "El coeficiente de Coriolis es negativo o cero," & Chr(13) & "debe de ser mayor que cero"
    Exit Function

```

```

End If
If Val(cc) <= 0 Or Not IsNumeric(cc) Then
'verifica que el coeficiente C de Keulegan sea positivo en cada sección
verificadatos = "El coeficiente C de Keulegan es negativo o cero," & Chr(13) & "debe de ser mayor que cero"
Exit Function
End If
If Val(cks) <= 0 Or Not IsNumeric(cks) Then
'verifica que la rugosidad relativa sea positiva en cada sección
verificadatos = "La rugosidad relativa de la sección es negativa o cero " & Chr(13) & "debe de ser mayor que
cero"
Exit Function
End If
If Val(calfan) <= 0 Or Not IsNumeric(calfan) Then
'verifica que el coeficiente alfa sub n de Keulegan sea positivo en cada sección
verificadatos = "El coeficiente alfan de Keulegan es negativo o cero," & Chr(13) & "debe de ser mayor que cero"
End If
If Val(cn) <= 0 Or Not IsNumeric(cn) Then
'verifica que el coeficiente n de Manning sea positivo en cada sección
verificadatos = "El coeficiente n de Manning es negativo o cero," & Chr(13) & "debe de ser mayor que cero"
End If
End If
Exit Function
error_handler:
'en caso de error
verificadatos = "error"
Exit Function
End Function
Function llenadatosformagrid(Forma As Form, grid As MSFlexGrid, i As Integer) As Boolean
'esta función pasa los datos de la forma al grid, se necesitan los parametros de la forma donde se encuentran, el grid al que
'descarga y la fila a la que se cargarán
On Error GoTo error_handler
With grid
.Row = i
.Col = 0: .Text = Mid(Forma.Caption, 9, Len(Forma.Caption) - 8)
.Col = 1: .Text = Forma.TxtB.Text: .Col = 2: .Text = Forma.TxtKi.Text
.Col = 3: .Text = Forma.TxtKd.Text: .Col = 4: .Text = Forma.TxtYm.Text
.Col = 5: .Text = Forma.CmbTipo.Text: .Col = 6: .Text = Forma.TxtAlfa.Text
.Col = 7: .Text = Forma.TxtC.Text: .Col = 8: .Text = Forma.TxtAlfan.Text
.Col = 9: .Text = Forma.TxtKs.Text: .Col = 10: .Text = Forma.TxtN.Text
End With
'al terminar de transferir los datos, la función toma el valor verdadero
llenadatosformagrid = True
Exit Function
error_handler:
llenadatosformagrid = False
Exit Function
End Function

Function ordenagrid(grid As MSFlexGrid, columna As Variant) As Boolean
'esta función ordena un grid por las alturas de berma de cada sección, por ende la primera sección será la central
On Error GoTo error_handler
total = grid.Rows - 1
ReDim oseccion(total) As String, otipo(total) As String, oB(total) As String, oki(total) As String, okd(total) As String, _
oym(total) As String, oalfa(total) As String, oc(total) As String, oalfan(total) As String, oks(total) As String, ordenado _
(total) As String, onn(total) As String
'defino variables temporales para guardar los datos de todas las filas
z = limpiafilas(grid)
'aquí mando llamar a la función que limpia todas las filas que están vacías
With grid
'vacío el valor de todas las celdas del grid a las variables temporales
For i = 1 To total

```

```

        .Row = i
        .Col = 0: oseccion(i) = .Text: .Col = 1: oB(i) = .Text: .Col = 2: oki(i) = .Text: .Col = 3: okd(i) = .Text
        .Col = 4: oym(i) = .Text: .Col = 5: otipo(i) = .Text: .Col = 6: oalfa(i) = .Text: .Col = 7: oc(i) = .Text
        .Col = 8: oalfan(i) = .Text: .Col = 9: oks(i) = .Text: .Col = 10: onn(i) = .Text: ordenado(i) = False
        'esta variable me indica si ya he ordenado la fila respectiva al tomar el valor de verdadero
    Next
    z = verificatipo(grid)
    'obtengo el valor de la fila en que se encuentra la sección central si es que ésta existe
    For i = 1 To total
        'aquí inicia el proceso de ordenar los datos del grid
        mayor = 0: menor = 0: indice = 0
        For j = 1 To total
            'en este proceso encuentro el valor de berma myor que existe en el grid para que al buscar el menor tenga un valor
            'de referencia para empezar
            grid.Col = columna
            If Val(oym(j)) > mayor And ordenado(j) = False Then mayor = Val(oym(j)): indice = j
        Next j
        menor = mayor
        'aquí es donde asigno el valor de menor como el mayor que existe hasta el momento
        For j = 1 To total
            'en este proceso ubico cual es el valor menor de altura de berma en todo el grid, sólo en los datos que no han sido
            'ordenados
            grid.Col = columna
            If menor > Val(oym(j)) And ordenado(j) = False Then menor = Val(oym(j)): indice = j
        Next j
        If central Then indice = seccioncentral: central = False
        'esta parte funciona para que cuando exista una sección central, ésta sea la primera en ser ordenada
        'aquí envío los valores desde las variables hacia la fila del grid que les corresponde al ordenar los valores
        .Row = i: .Col = 0: .Text = i: .Col = 1: .Text = oB(indice): .Col = 2: .Text = oki(indice)
        .Col = 3: .Text = okd(indice): .Col = 4: .Text = oym(indice): .Col = 5: .Text = otipo(indice)
        .Col = 6: .Text = oalfa(indice): .Col = 7: .Text = oc(indice): .Col = 8: .Text = oalfan(indice)
        .Col = 9: .Text = oks(indice): .Col = 10: .Text = onn(indice): ordenado(indice) = True
        'aquí determino que el renglón ha sido ordenado
    Next i
End With
ordenagrid = True
'al no haber ningún error, la función toam el valor verdadero
Exit Function
error_handler:
'en caso de error
    ordenagrid = False
    Exit Function
End Function

Function encuentraflavacia(grid As MSFlexGrid) As Integer
'esta función encuentra la fila que está vacía en un grid y regresa el valor de la fila en que se encuentra la fila vacía
    On Error GoTo error_handler
    encuentraflavacia = 0: 'el valor inicial de la función es 0, lo cual equivale al encabezado, que nunca se eliminará
    With grid
        For i = 1 To .Rows - 1
            'me voy fila por fila
            .Row = i: vacio = 0: 'esta variable me sirve de indicador en caso de que la fila esté vacía
            For j = 1 To .Cols - 1
                'aquí reviso columna por columna de la fila i
                .Col = j
                If .Text <> "" Then vacio = 1
                'en caso de alguna celda tenga valor de cadena diferente de vacío el indicador vacío cambiará del valor original
            Next
            If vacio = 0 Then
                'al tener una fila vacía se le da a la función ese valor y se sale
                encuentraflavacia = i
            End If
        Next
    End With
End Function

```



```

        Exit Function
    End If
Next
End With
Exit Function
'al terminar el proceso y no cambiar el valor de la variable, se sabe que no existe ninguna fila vacía
error_handler:
'en caso de error
    encuentraflavacia = 0
    Exit Function
End Function

Function eliminafila(grid As MSFlexGrid, indice As Integer) As Boolean
'esta función tiene la utilidad de eliminar una fila en el grid se le alimenta el grid y la fila a eliminar, sin embargo al enviar el
'valor de fila 0, permite encontrar la o las filas vacías
    On Error GoTo error_handler
    If indice = 0 Then indice = encuentraflavacia(grid)
    'en caso de que la función sea llamada y no se especifique la fila (el encabezado es el valor por descontado) busca la fila
    'vacía
    If indice = 0 Then eliminafila = False: Exit Function
    'en caso de que no se haya encontrado ninguna fila vacía el proceso termina
    With grid
        .Col = 0
        If indice < .Rows - 1 Then
            'verifico si la fila a eliminar es la última
            For i = indice To .Rows - 2
                'este proceso sustituye los valores de la fila que quiero borrar por los de la fila siguiente y así sucesivamente hasta
                'terminar con todas las filas del grid
                .Row = i
                For j = 0 To .Cols - 1
                    .Col = j: .Row = .Row + 1: temp = .Text: .Row = .Row - 1: .Text = temp
                Next
            Next
        End If
        .Rows = .Rows - 1
        'reduzco el número de filas en uno
    End With
    eliminafila = True
    Exit Function
error_handler:
'en caso de error
    eliminafila = False
    Exit Function
End Function

Function limpiafilas(grid As MSFlexGrid) As Boolean
'esta función cumple el objetivo de limpiar todas las filas del grid que están vacías
    On Error GoTo error_handler
    Static limpia As Integer
    limpia = 1
    'esta variable me indicará qué fila eliminar si es el caso, al tomar el valor de 0 la función terminará
    While limpia <> 0
        limpia = encuentraflavacia(grid)
        'encuentro la primera fila vacía
        If limpia = 0 Then limpiafilas = True: Exit Function
        'si no existe alguna fila vacía, se termina este proceso
        z = eliminafila(grid, limpia)
        'elimino la fila vacía encontrada
    Wend
    limpiafilas = True
    Exit Function

```

```

error_handler:
'en caso de error
    limpiafilas = False
    Exit Function
End Function

Function revisadatosgrid(grid As MSFlexGrid) As Boolean
'esta función revisa los datos del grid para habilitar o no el menú de cálculo
    On Error GoTo error_handler
    revisadatosgrid = True: total = grid.Rows - 1
    If total = 0 Then GoSub error_handler
    ReDim oseccion(total) As String, otipo(total) As String, oB(total) As String, oki(total) As String, okd(total) As String, _
    oym(total) As String, oalfa(total) As String, oc(total) As String, oalfan(total) As String, oks(total) As String, onn(total) _
    As String
    'defino variables temporales para guardar los datos de todas las filas
    With grid
    'vacío el valor de todas las celdas del grid a las variables temporales
        For i = 1 To total
            .Row = i:.Col = 0: oseccion(i) = .Text: .Col = 1: oB(i) = .Text: .Col = 2: oki(i) = .Text
            .Col = 3: okd(i) = .Text: .Col = 4: oym(i) = .Text: .Col = 5: otipo(i) = .Text: .Col = 6: oalfa(i) = .Text
            .Col = 7: oc(i) = .Text: .Col = 8: oalfan(i) = .Text: .Col = 9: oks(i) = .Text: .Col = 10: onn(i) = .Text
            correcto = verificadatos(oseccion(i), otipo(i), oB(i), oki(i), okd(i), oym(i), oalfa(i), oc(i), oalfan(i), oks(i), grid,
            (i)): 'aquí reviso los datos de todo el grid, si no existe ningún problema, la función toma el valor verdadero
            If Not (correcto = "") Then revisadatosgrid = False: Exit For
        Next
    End With
    Exit Function
error_handler:
'en caso de error
    revisadatosgrid = False
    Exit Function
End Function

Function verificadatosgasto(gasto As String, gravedad As String) As String
'esta función verifica que los datos de gasto, gravedad y factor del coeficiente de rugosidad sean correctos
    On Error GoTo error_handler
    verificadatosgasto = ""
    If Not IsNumeric(gasto) Then
    'el gasto debe ser un número
        verificadatosgasto = "El gasto debe ser un número": Exit Function
    ElseIf Not (Val(gasto) > 0) Then
    'el gasto debe ser mayor que 0
        verificadatosgasto = "El gasto debe ser mayor que 0": Exit Function
    End If
    If Not IsNumeric(gravedad) Then
    'la gravedad debe ser un número
        verificadatosgasto = "La gravedad debe ser un número": Exit Function
    ElseIf Not (Val(gravedad) > 0) Then
    'la gravedad debe ser mayor de 0
        verificadatosgasto = "La gravedad debe ser mayor que 0": Exit Function
    End If
    If Abs((Val(gravedad) - 9.81) / 9.81) * 100 > 5 Then verificadatosgasto = verificadatosgasto & "95g": Exit Function
    'si la gravedad indicada varía en mas de 5% de 9.81 lo indico
    Exit Function
error_handler:
'en caso de error
    verificadatosgasto = "Error"
End Function

Function continuare resultados(gasto As String, gravedad As String) As Boolean
'esta función determina si se continúa con el proceso de obtención de resultados

```

```

On Error GoTo error_handler
continuaresultados = True
mensaje = verificadatosgasto(gasto, gravedad)
If mensaje <> "" And InStr(mensaje, "95g") = 0 And InStr(mensaje, "1.19") = 0 Then
'si hay un error, lo hago saber
    z = MsgBox(mensaje, vbInformation + vbOKOnly, "¡Atención!"); continuaresultados = False: Exit Function
End If
If Not (InStr(mensaje, "95g") = 0) Then
'si la gravedad indicada difiere en más e 5% con la estándar de 9.81
    mensaje1 = "La gravedad indicada difiere en mas de 5% de 9.81" & Chr(13) & "¿Desea continuar de todas formas?"
    z = MsgBox(mensaje1, vbInformation + vbYesNo, "¡Atención!")
    If z = vbNo Then continuaresultados = False: Exit Function
    continuaresultados = True
End If
Exit Function
error_handler:
'en caso de error
    continuaresultados = False
    Exit Function
End Function

```

```

Function copiagrid(grid1 As MSFlexGrid, grid2 As MSFlexGrid) As Boolean
'copia la malla 2 a la malla 1
    On Error GoTo error_handler
    grid1.Cols = grid2.Cols: grid1.Rows = grid2.Rows
    For i = 0 To grid2.Rows - 1
        grid2.Row = i: grid1.Row = i
        For j = 0 To grid2.Cols - 1
            grid2.Col = j: grid1.Col = j: grid1.CellAlignment = 4: grid1.Text = grid2.Text
            grid1.ColWidth(j) = grid2.ColWidth(j)
        Next
    Next
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function verificafactor(factor As String) As String
'esta función verifica que el valor asignado al factor para coeficiente de rugosidad sea un número positivo y esté dentro de
'cierto rango de valores
    On Error GoTo error_handler
    If Not IsNumeric(factor) Then
        'el factor del coeficiente de rugosidad debe ser un número
        verificafactor = "El factor del coeficiente de rugosidad debe ser un número": Exit Function
    ElseIf Not (Val(factor) > 0) Then
        'el factor del coeficiente de rugosidad debe ser mayor de 0
        verificafactor = "El factor del coeficiente de rugosidad debe ser mayor que 0": Exit Function
    End If
    If Abs(Val(factor) - 1) > 0.2 Then verificafactor = verificafactor & "1.19"
    'el factor del coeficiente de rugosidad no es 1.19
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function continuafactor(factor As String) As Boolean
'esta función verifica que el valor asignado al factor para coeficiente de rugosidad sea un número positivo y esté dentro de
'cierto rango de valores
    On Error GoTo error_handler
    mensaje = verificafactor(factor): continuafactor = True

```

```

If mensaje <> "" And InStr(mensaje, "1.19") = 0 Then
'si hay un error, lo hago saber
z = MsgBox(mensaje, vbInformation + vbOKOnly, "¡Atención!"); continuafactor = False: Exit Function
End If
If Not (InStr(mensaje, "1.19") = 0) Then
'si el factor del coeficiente de rugosidad es diferente de 1.19
mensaje1 = "El factor del coeficiente de rugosidad tiene más de " & Chr(13) & "20% de diferencia respecto de la" _
& "unidad" & Chr(13) & "¿Desea continuar de todas formas?"
z = MsgBox(mensaje1, vbInformation + vbYesNo, "¡Atención!")
If z = vbNo Then continuafactor = False: Exit Function
continuafactor = True
End If
Exit Function
error_handler:
'en caso de error
continuafactor = False
End Function

Function dibujo() As Boolean
'esta función determina las dimensiones de la sección del canal a analizar a partir de los datos proporcionados
On Error GoTo error_handler
nyderpar = 0: nyizqpar = 0: nxderpar = 0: nxizqpar = 0: nx = 0: nyderult = 0: nyizqult = 0: nkderult = 0: nkizqult = 0
llenavariabes Frmdatos.Grddatos: procesos = cuentaprosesos
If Not (sender And senizq) Then
'en caso de que la sección no sea sencilla
If Not senizq Then nyizqpar = nyizqpar + nymizq(1): nxizqpar = nxizqpar + nymizq(1) * nki(1)
If Not sender Then nyderpar = nyderpar + nymder(1): nxderpar = nxderpar + nymder(1) * nk(1)
For i = 1 To nymderecha - 1
nyderpar = nyderpar + (nymder(i + 1) - nymder(i))
nxderpar = nxderpar + nbder(i) + nkder(i) * (nymder(i + 1) - nymder(i))
Next
nxderpar = nxderpar + nbder(nymderecha): nyderult = nyprom: nkderult = nkder(nymderecha)
For i = 1 To nymizquierda - 1
nyizqpar = nyizqpar + (nymizq(i + 1) - nymizq(i))
nxizqpar = nxizqpar + nbizq(i) + nkizq(i) * (nymizq(i + 1) - nymizq(i))
Next
nxizqpar = nxizqpar + nbizq(nymizquierda): nyizqult = nyprom: nkizqult = nkizq(nymizquierda)
End If
If sender Then nyderult = nB(1): nkderult = nk(1)
'si no existen bermas del lado derecho
If senizq Then nyizqult = nB(1): nkizqult = nki(1)
'si no existen bermas del lado izquierdo
nyder = nyderpar + nyderult: nyizq = nyizqpar + nyizqult
If nyder > nyizq Then
nyizqult = nyder - nyizqpar: nyizq = nyizqpar + nyizqult
Else
nyderult = nyizq - nyderpar: nyder = nyderpar + nyderult
End If
nxderult = nyderult * nkderult: nxder = nxderpar + nxderult: nxizqult = nyizqult * nkizqult
nxizq = nxizqpar + nxizqult: nx = nxder + nxizq + nB(1)
Frmdatos.PicSeccion.Scale (0, nyder * 1.1)-(nx * 1.1, 0): Frmdatos.PicSeccion.Cls
'aquí se definen los límites del área de dibujo
x0 = nx * 0.05: y0 = nyder * 1.05: X1 = x0 + nxizqult: Y1 = y0 - nyizqult
Frmdatos.PicSeccion.PSet (x0, y0): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
If Not senizq Then
X1 = x0 + nbizq(nymizquierda): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
For i = nymizquierda - 1 To 1 Step -1
X1 = x0 + nkizq(i) * (nymizq(i + 1) - nymizq(i)): Y1 = y0 - (nymizq(i + 1) - nymizq(i))
Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1: X1 = x0 + nbizq(i)
Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
Next

```

```

X1 = x0 + nki(1) * nymizq(1): Y1 = y0 - nymizq(1): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack
x0 = X1: y0 = Y1
End If
X1 = x0 + nB(1): Y1 = y0: Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
If Not sender Then
    X1 = x0 + nkd(1) * nymder(1): Y1 = y0 + nymder(1): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack
    For i = 1 To nymderecha - 1
        X1 = x0 + nbder(i): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
        X1 = x0 + nkd(i + 1) * (nymder(i + 1) - nymder(i)): Y1 = y0 + nymder(i + 1) - nymder(i)
        Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
    Next
    X1 = x0 + nbder(nymderecha): Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack: x0 = X1: y0 = Y1
End If
X1 = x0 + nxderult: Y1 = y0 + nyderult: Frmdatos.PicSeccion.Line -(X1, Y1), vbBlack
x0 = nx * 0.025: y0 = nyder * 0.05: X1 = x0: Y1 = y0 + 1: X2 = x0 + 1: Y2 = y0
Frmdatos.PicSeccion.Line (x0, y0)-(X1, Y1), vbBlack: Frmdatos.PicSeccion.Line (x0, y0)-(X2, Y2), vbBlack
dibujo = True: Exit Function
error_handler:
'en caso de error
    dibujo = False
End Function

```

## Mod\_ppal

```

Global caso As String, accesodbase As Database, rsdbase As Recordset
Global caso1 As String, malconfigurado As String
Global renglon1 As String, renglon2 As String, renglon3 As String
Global ubicacion As String, archivoenuso As String, carpetaenuso As String
Global campos() As String, tipo() As String, indicatipo As Variant
Global cuentacentral As Integer, seccioncentral As Integer, central As Boolean
Global registro As Integer, simple As Boolean, Chaudhry As Boolean
Global nseccion() As Integer, ntipo() As String, nB() As Double, nki() As Double, nkd() As Double, nym() As Double, _
nalfa() As Double, nc() As Double, nalfan() As Double, nks() As Double, nnm() As Double
'declaro las variables de los datos para realizar los cálculos
Global nTirante As Double, nTirantesig As Double, nderecha As Integer, nizquierda As Integer
Global total As Integer, nymdif() As Double, nyprom As Double
Global procesos As Integer, seccionesproceso As Integer
Global nymderecha As Integer, nymizquierda As Integer
Global nymder() As Double, nymizq() As Double, nkder() As Double, nkizq() As Double
Global nbder() As Double, nbizq() As Double, sender As Boolean, senizq As Boolean
Global Qmax As Double, Qmin As Double
Global nArea() As Double, nAreap() As Double
Global nAncho() As Double, nDAncho() As Double
Global nPerimetro() As Double, nDPerimetro() As Double, nPerimetrop() As Double
Global nRadio() As Double, nFactor() As Double
Global nn() As Double, nFactorn() As Double, nfactornc() As Boolean
Global ndn() As Double, nSigma1() As Double, nSigma2() As Double, nSigma3() As Double
Global nyc() As Double, ny() As Double
Global puntos() As Single, puntosT As Integer, diferencia() As Double, diferenciamax As Double
Global XEnergia() As Double, YTirante() As Double, XEnergiatem() As Double, YTirantetem() As Double, _
XEnergiatem1() As Double, YTirantetem1() As Double
Global Q As Double, g As Double, nerror As Double, nEnergia As Double, nFroude As Double
Global nAreaT As Double, nAnchoT As Double, nDAnchoT As Double, nPerimetroT As Double
Global nRadioT As Double, nFactorT As Double, nAlfaT As Double
Global nSigma1T As Double, nSigma2T As Double, nSigma3T As Double
Global nAm As Double, nTm As Double, nBetaT As Double, nBetapT As Double, nBeta() As Double
Global nBetap() As Double, nDFactor() As Double, nDFactorT As Double
Global Cmax As Double, nyCmax As Double
'aquí deben de declararse todas las variables para el cálculo

```

## Mod\_tirante

Function llenavariabales(grid As MSFlexGrid) As Boolean

'esta función llena las variables con los datos que aparecen en el grid de datos

```

On Error GoTo error_handler
With grid
    total = .Rows - 1
    ReDim nseccion(total) As Integer, ntipo(total) As String, nB(total) As Double, nki(total) As Double, nkd(total) As _
    Double, nym(total) As Double, nalfa(total) As Double, nc(total) As Double, nalfan(total) As Double, nks(total) As _
    Double, nFactornc(total) As Double, nfactornc(total) As Boolean, nnm(total) As Double
    'dimensiono las variables de los datos para realizar los cálculos
    For i = 1 To total
        .Row = i:.Col = 0: nseccion(i) = Val(.Text): .Col = 1: nB(i) = Val(.Text): .Col = 2: nki(i) = Val(.Text)
        .Col = 3: nkd(i) = Val(.Text): .Col = 4: nym(i) = Val(.Text): .Col = 5: ntipo(i) = .Text
        .Col = 6: nalfa(i) = Val(.Text): .Col = 7: nc(i) = Val(.Text): .Col = 8: nalfan(i) = Val(.Text)
        .Col = 9: nks(i) = Val(.Text): .Col = 10: nnm(i) = Val(.Text): nFactornc(i) = 1: nfactornc(i) = True
    Next
End With
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function cuentaprosesos() As Integer

'esta función calcula el número de procesos a realizar de acuerdo con el número de secciones, el lado en que se encuentran y la altura de la berma de cada una

```

On Error GoTo error_handler
cuentaprosesos = 1: nymprom = 0
For i = 1 To total - 1
    nymprom = nymprom + (nym(i + 1) - nym(i)): berma = nym(i + 1)
    If berma <> nym(i) Then cuentaprosesos = cuentaprosesos + 1
Next
nymprom = nymprom / total
ReDim nymdif(cuentaprosesos) As Double
nymdif(1) = nym(1): nymderecha = 0: nymizquierda = 0: j = 2
For i = 1 To total - 1
    If ntipo(i + 1) = "derecha" Then nymderecha = nymderecha + 1
    If ntipo(i + 1) = "izquierda" Then nymizquierda = nymizquierda + 1
    berma = nym(i + 1)
    If berma <> nym(i) Then nymdif(j) = nym(i + 1): j = j + 1
Next
If nymderecha = 0 Then nymderecha = 1
If nymizquierda = 0 Then nymizquierda = 1
ReDim nymder(nymderecha) As Double, nymizq(nymizquierda) As Double
ReDim nbder(nymderecha) As Double, nbizq(nymizquierda) As Double
ReDim nkder(nymderecha) As Double, nkizq(nymizquierda) As Double
sender = False: senizq = False: j = 1: K = 1
For i = 1 To total
    If ntipo(i) = "derecha" Then nymder(j) = nym(i): nbder(j) = nB(i): nkder(j) = nkd(i): j = j + 1
    If ntipo(i) = "izquierda" Then nymizq(K) = nym(i): nbizq(K) = nB(i): nkizq(K) = nki(i): K = K + 1
Next
If nymderecha = 1 And nymder(1) = 0 Then sender = True: nymder(1) = 1000 * mayorlista(nymizq)
If nymizquierda = 1 And nymizq(1) = 0 Then senizq = True: nymizq(1) = 1000 * mayorlista(nymder)
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function Area(B As Double, y As Double, ki As Double, kd As Double, tipo As Integer) As Double

'esta función calcula el área de una sección transversal recibiendo los valores de: B, es el ancho de plantilla y, es el tirante 'ki, es el talud izquierdo kd, es el talud derecho la variable tipo me indica el tipo de área a calcular, tomando los siguientes

'valores:  
'0 cuando se consideran ambos taludes, y la plantilla está en contacto con el suelo  
'1 cuando sólo se considera el talud izquierdo, ya que el talud derecho es la frontera vertical entre dos secciones, y la plantilla está en contacto con el suelo  
'2 cuando se considera el talud derecho, ya que el talud izquierdo es la frontera vertical entre dos secciones, y la plantilla está en contacto con el suelo  
'3 cuando sólo se considera el talud izquierdo, ya que el talud derecho es la frontera vertical entre dos secciones, y la plantilla es frontera entre dos secciones  
'4 cuando se considera el talud derecho, ya que el talud izquierdo es la frontera vertical entre dos secciones, y la plantilla es frontera entre dos secciones  
'5 cuando ambos taludes son fronteras verticales entre dos secciones, y la plantilla es frontera entre dos secciones

```
On Error GoTo error_handler
Select Case tipo
Case 0
Area = B * y + (kd + ki) / 2 * y ^ 2
Case 1
Area = B * y + ki / 2 * y ^ 2
Case 2
Area = B * y + kd / 2 * y ^ 2
Case 3
Area = B * y + ki / 2 * y ^ 2
Case 4
Area = B * y + kd / 2 * y ^ 2
Case 5
Area = B * y
End Select
Exit Function
```

```
error_handler:
'en caso de error
Exit Function
End Function
```

Function Ancho(B As Double, y As Double, ki As Double, kd As Double, tipo As Integer) As Double

'esta función calcula el ancho de superficie libre de la sección transversal, recibiendo los mismos valores de la función área

```
On Error GoTo error_handler
Select Case tipo
Case 0
Ancho = B + (kd + ki) * y
Case 1
Ancho = B + ki * y
Case 2
Ancho = B + kd * y
Case 3
Ancho = B + ki * y
Case 4
Ancho = B + kd * y
Case 5
Ancho = B
End Select
Exit Function
```

```
error_handler:
'en caso de error
Exit Function
End Function
```

Function DAncho(ki As Double, kd As Double, tipo As Integer) As Double

'esta función calcula la derivada del ancho de superficie libre respecto del tirante de la sección transversal, recibiendo los datos de taludes y el tipo de sección únicamente

```
On Error GoTo error_handler
Select Case tipo
Case 0
DAncho = kd + ki
```

```

    Case 1
        DAncho = ki
    Case 2
        DAncho = kd
    Case 3
        DAncho = ki
    Case 4
        DAncho = kd
    Case 5
        DAncho = 0
End Select
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function Perimetro(B As Double, y As Double, ki As Double, kd As Double, tipo As Integer ) As Double  
 'esta función calcula el perímetro mojado de la sección transversal, recibiendo los mismos valores de la función área

```

    On Error GoTo error_handler
    Select Case tipo
    Case 0
        Perimetro = B + (Sqr(ki ^ 2 + 1) + Sqr(kd ^ 2 + 1)) * y
    Case 1
        Perimetro = B + Sqr(ki ^ 2 + 1) * y
    Case 2
        Perimetro = B + Sqr(kd ^ 2 + 1) * y
    Case 3
        Perimetro = Sqr(ki ^ 2 + 1) * y
    Case 4
        Perimetro = Sqr(kd ^ 2 + 1) * y
    Case 5
        Perimetro = 0
    End Select
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function DPerimetro(ki As Double, kd As Double, tipo As Integer) As Double  
 'esta función calcula la derivada del perímetro mojado respecto del tirante de la sección transversal, recibiendo los datos de taludes y el tipo de sección únicamente

```

    On Error GoTo error_handler
    Select Case tipo
    Case 0
        DPerimetro = Sqr(ki ^ 2 + 1) + Sqr(kd ^ 2 + 1)
    Case 1
        DPerimetro = Sqr(ki ^ 2 + 1)
    Case 2
        DPerimetro = Sqr(kd ^ 2 + 1)
    Case 3
        DPerimetro = Sqr(ki ^ 2 + 1)
    Case 4
        DPerimetro = Sqr(kd ^ 2 + 1)
    Case 5
        DPerimetro = 0
    End Select
Exit Function
error_handler:
'en caso de error

```



```
Exit Function
End Function
```

```
Function Radio(A As Double, P As Double) As Double
'esta función calcula el radio hidráulico de una sección transversal
  On Error GoTo error_handler
  Radio = A / P
  Exit Function
error_handler:
'en caso de error
  Exit Function
End Function
```

```
Function n(Rh As Double, g As Double, alfa As Double, c As Double, ks As Double, nFactorn As Double) As Double
'esta función calcula el coeficiente n de rugosidad de Manning
  On Error GoTo error_handler
  n = nFactorn * Rh ^ (1 / 6) / (log10(c * Rh / ks) * alfa * Sqr(8 * g))
  Exit Function
error_handler:
'en caso de error
  Exit Function
End Function
```

```
Function factor(A As Double, Rh As Double, n As Double) As Double
'esta función obtiene el Factor de conducción de una sección transversal
  On Error GoTo error_handler
  factor = A * Rh ^ (2 / 3) / n
  Exit Function
error_handler:
'en caso de error
  Exit Function
End Function
```

```
Function dn(alfa As Double, n As Double, g As Double, Rh As Double, T As Double, dP As Double) As Double
'esta función obtiene la variación del coeficiente n de rugosidad de Manning con respecto al tirante (A/n)(dn/dy)
  On Error GoTo error_handler
  dn = (1 / 6 - (alfa * n * (8 * g) ^ 0.5 * log10(Exp(1)))) / Rh ^ (1 / 6) * (T - Rh * dP)
  Exit Function
error_handler:
'en caso de error
  Exit Function
End Function
```

```
Function csigma1(alfa As Double, K As Double, A As Double, T As Double, Rh As Double, dP As Double, dn As Double) As Double
'esta función obtiene el valor de sigma1 para la sección dada
  On Error GoTo error_handler
  csigma1 = alfa * (K / A) ^ 3 * (3 * T - 2 * Rh * dP - 3 * dn)
  Exit Function
error_handler:
'en caso de error
  Exit Function
End Function
```

```
Function csigma2(alfa As Double, K As Double, A As Double)
'esta función obtiene el valor de sigma2 para la sección dada
  On Error GoTo error_handler
  csigma2 = alfa / A ^ 2 * K ^ 3
  Exit Function
error_handler:
'en caso de error
```

```
Exit Function
End Function
```

```
Function csigma3(K As Double, A As Double, T As Double, Rh As Double, dP As Double, dn As Double) As Double
```

```
'esta función obtiene el valor de sigma3 para la sección dada
```

```
On Error GoTo error_handler
```

```
csigma3 = K / A * (5 * T - 2 * Rh * dP - 3 * dn)
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function log10(x As Double) As Double
```

```
'esta función obtiene el logaritmo base 10 de el número que se le alimenta
```

```
On Error GoTo error_handler
```

```
log10 = Log(x) / Log(10)
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function criticosencillo(inicial) As Double
```

```
'esta función obtiene el tirante crítico para una sección sencilla se emplea en este caso como primer paso del procedimiento
```

```
On Error GoTo error_handler
```

```
Dim y As Double: ny(1) = inicial: nerror = 1
```

```
While nerror > 0.000001
```

```
nArea(1) = Area(nB(1), ny(1), nki(1), nkd(1), 0)
```

```
nAncho(1) = Ancho(nB(1), ny(1), nki(1), nkd(1), 0)
```

```
nDAncho(1) = DAncho(nki(1), nkd(1), 0)
```

```
y = ny(1) - F(nArea(1), nAncho(1), nalfa(1)) / DF(nArea(1), nAncho(1), nDAncho(1))
```

```
nerror = Abs((y - ny(1)) / y * 100)
```

```
ny(1) = y
```

```
Wend
```

```
criticosencillo = ny(1)
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function F(A As Double, T As Double, alfa As Double) As Double
```

```
'esta función se refiere a la del método numérico de Newton para obtener el tirante crítico en una sección sencilla
```

```
On Error GoTo error_handler
```

```
F = A ^ 3 / T - alfa * Q ^ 2 / g
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function DF(A As Double, T As Double, dT As Double) As Double
```

```
'esta función se refiere a la derivada de la función para el método numérico de Newton para obtener el tirante crítico en una
```

```
'sección sencilla
```

```
On Error GoTo error_handler
```

```
DF = 3 * A ^ 2 - A ^ 3 / (T ^ 2) * dT
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

End Function

```
Function encuentraseccionesproceso(referencia As Double) As Integer
'esta función determina las secciones que se utilizan en el proceso de cálculo
On Error GoTo error_handler
For z = 1 To total
    If nym(z) <= referencia Then
        encuentraseccionesproceso = nseccion(z)
    Else
        Exit Function
    End If
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function
```

```
Function encuentraderecha(y As Double) As Integer
'esta función determina la posición de la sección derecha en proceso de cálculo
On Error GoTo error_handler
For z = 1 To nymderecha
    If nymder(z) = y Then encuentraderecha = z: Exit Function
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function
```

```
Function encuentraizquierda(y As Double) As Integer
'esta función determina la posición de la sección izquierda en proceso de cálculo
On Error GoTo error_handler
For z = 1 To nymizquierda
    If nymizq(z) = y Then encuentraizquierda = z: Exit Function
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function
```

```
Function AreaTotal(secciones As Integer, nTirante As Double) As Double
'esta función calcula el área de todas las secciones para un tirante determinado
On Error GoTo error_handler
If secciones = 1 Then ny(1) = nTirante - nym(1): nArea(1) = Area(nB(1), ny(1), nki(1), nk(1), 0): AreaTotal = nArea(1)
For j = 1 To secciones
    If ntipo(j) = "central" Then
        'para las secciones centrales
        ny(j) = nTirante - nym(j)
        If nTirante < nymder(1) And nTirante < nymizq(1) Then
            nArea(j) = Area(nB(j), ny(j), nki(j), nk(j), 0)
        End If
        If nTirante > nymder(1) And nymizq(1) >= nymder(1) Then
            nAreap(1) = Area(nB(j), nymder(1), nki(j), nk(j), 0)
            If nTirante < nymizq(1) Then
                nAreap(2) = Area(Ancho(nB(j), nymder(1), nki(j), nk(j), 0), nTirante - nymder(1), nki(j), nk(j), 3)
                nArea(j) = nAreap(1) + nAreap(2)
            ElseIf nTirante > nymizq(1) Then
                nAreap(2) = Area(Ancho(nB(j), nymder(1), nki(j), nk(j), 0), nymizq(1) - nymder(1), nki(j), nk(j), 3)
                nAreap(3) = Area(Ancho(Ancho(nB(j), nymder(1), nki(j), nk(j), 0), nymizq(1) - nymder(1), nki(j), nk(j), _
```

```

        1), ny(j) - nymizq(1), nki(j), nkd(j), 5)
        nArea(j) = nAreap(1) + nAreap(2) + nAreap(3)
    End If
End If
If nTirante > nymizq(1) And nymizq(1) < nymder(1) Then
    nAreap(1) = Area(nB(j), nymizq(1), nki(j), nkd(j), 0)
    If nTirante < nymder(1) Then
        nAreap(2) = Area(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nTirante - nymizq(1), nki(j), nkd(j), 4)
        nArea(j) = nAreap(1) + nAreap(2)
    ElseIf nTirante > nymder(1) Then
        nAreap(2) = Area(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nymder(1) - nymizq(1), nki(j), nkd(j), 4)
        nAreap(3) = Area(Ancho(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nymder(1) - nymizq(1), nki(j), _
        nkd(j), 1), ny(j) - nymder(1), nki(j), nkd(j), 5)
        nArea(j) = nAreap(1) + nAreap(2) + nAreap(3)
    End If
End If
ElseIf ntipo(j) = "derecha" Then
'para las secciones derechas
    nderecha = encuentraderecha(nym(j)): ny(j) = nTirante - nym(j)
    If nderecha = nymderecha Then
        nArea(j) = Area(nB(j), ny(j), nki(j), nkd(j), 2)
    ElseIf nderecha < nymderecha Then
        If nTirante < nymder(nderecha + 1) Then
            nArea(j) = Area(nB(j), ny(j), nki(j), nkd(j), 2)
        ElseIf nTirante > nymder(nderecha + 1) Then
            nAreap(1) = Area(nB(j), nymder(nderecha + 1) - nymder(nderecha), nki(j), nkd(j), 2)
            nAreap(2) = Area(Ancho(nB(j), nymder(nderecha + 1) - nymder(nderecha), nki(j), nkd(j), 2), nTirante - _
            nymder(nderecha + 1), nki(j), nkd(j), 5)
            nArea(j) = nAreap(1) + nAreap(2)
        End If
    End If
ElseIf ntipo(j) = "izquierda" Then
'para las secciones izquierdas
    nizquierda = encuentraizquierda(nym(j)): ny(j) = nTirante - nym(j)
    If nizquierda = nymizquierda Then
        nArea(j) = Area(nB(j), ny(j), nki(j), nkd(j), 1)
    ElseIf nizquierda < nymizquierda Then
        If nTirante < nymizq(nizquierda + 1) Then
            nArea(j) = Area(nB(j), ny(j), nki(j), nkd(j), 1)
        ElseIf nTirante > nymizq(nizquierda + 1) Then
            nAreap(1) = Area(nB(j), nymizq(nizquierda + 1) - nymizq(nizquierda), nki(j), nkd(j), 1)
            nAreap(2) = Area(Ancho(nB(j), nymizq(nizquierda + 1) - nymizq(nizquierda), nki(j), nkd(j), 1), nTirante _
            - nymizq(nizquierda + 1), nki(j), nkd(j), 5)
            nArea(j) = nAreap(1) + nAreap(2)
        End If
    End If
End If
AreaTotal = AreaTotal + nArea(j)
Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function AnchoTotal(secciones As Integer, nTirante As Double) As Double

'esta función calcula el ancho de superficie libre de todas las secciones para un tirante determinado

On Error GoTo error\_handler

For j = 1 To secciones

    If ntipo(j) = "central" Then

        'para las secciones centrales

```

ny(j) = nTirante - nym(j)
If nTirante < nymder(1) And nTirante < nymizq(1) Then nAncho(j) = Ancho(nB(j), ny(j), nki(j), nkd(j), 0)
If nTirante > nymder(1) And nymizq(1) >= nymder(1) Then
  If nTirante < nymizq(1) Then
    nAncho(j) = Ancho(Ancho(nB(j), nymder(1), nki(j), nkd(j), 0), nTirante - nymder(1), nki(j), nkd(j), 3)
  ElseIf nTirante > nymizq(1) Then
    nAncho(j) = Ancho(Ancho(Ancho(nB(j), nymder(1), nki(j), nkd(j), 0), nymizq(1) - nymder(1), nki(j), _
      nkd(j), 3), ny(j) - nymizq(1), nki(j), nkd(j), 5)
  End If
End If
If nTirante > nymizq(1) And nymizq(1) < nymder(1) Then
  If nTirante < nymder(1) Then
    nAncho(j) = Ancho(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nTirante - nymizq(1), nki(j), nkd(j), 4)
  ElseIf nTirante > nymder(1) Then
    nAncho(j) = Ancho(Ancho(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nymder(1) - nymizq(1), nki(j), _
      nkd(j), 4), ny(j) - nymder(1), nki(j), nkd(j), 5)
  End If
End If
ElseIf ntipo(j) = "derecha" Then
'para las secciones derechas
nderecha = encuentraderecha(nym(j)): ny(j) = nTirante - nym(j)
If nderecha = nymderecha Then
  nAncho(j) = Ancho(nB(j), ny(j), nki(j), nkd(j), 2)
ElseIf nderecha < nymderecha Then
  If nTirante < nymder(nderecha + 1) Then
    nAncho(j) = Ancho(nB(j), ny(j), nki(j), nkd(j), 2)
  ElseIf nTirante > nymder(nderecha + 1) Then
    nAncho(j) = Ancho(Ancho(nB(j), nymder(nderecha + 1) - nymder(nderecha), nki(j), nkd(j), 2), nTirante _
      - nymder(nderecha + 1), nki(j), nkd(j), 5)
  End If
End If
ElseIf ntipo(j) = "izquierda" Then
'para las secciones izquierdas
nizquierda = encuentraizquierda(nym(j)): ny(j) = nTirante - nym(j)
If nizquierda = nymizquierda Then
  nAncho(j) = Ancho(nB(j), ny(j), nki(j), nkd(j), 1)
ElseIf nizquierda < nymizquierda Then
  If nTirante < nymizq(nizquierda + 1) Then
    nAncho(j) = Ancho(nB(j), ny(j), nki(j), nkd(j), 1)
  ElseIf nTirante > nymizq(nizquierda + 1) Then
    nAncho(j) = Ancho(Ancho(nB(j), nymizq(nizquierda + 1) - nymizq(nizquierda), nki(j), nkd(j), 1), _
      nTirante - nymizq(nizquierda + 1), nki(j), nkd(j), 5)
  End If
End If
End If
AnchoTotal = AnchoTotal + nAncho(j)
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

Function DAnchoTotal(secciones As Integer, nTirante As Double) As Double

'esta función calcula la derivada del Ancho de superficie libre respecto al tirante de todas las secciones para un tirante determinado

```

On Error GoTo error_handler
For j = 1 To secciones
  If ntipo(j) = "central" Then
'para las secciones centrales
  ny(j) = nTirante - nym(j)

```

```

    If nTirante < nymder(1) And nTirante < nymizq(1) Then
        nDAncho(j) = DAncho(nki(j), nkd(j), 0)
    End If
    If nTirante > nymder(1) And nymizq(1) >= nymder(1) Then
        If nTirante < nymizq(1) Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 3)
        ElseIf nTirante > nymizq(1) Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 5)
        End If
    End If
    If nTirante > nymizq(1) And nymizq(1) < nymder(1) Then
        If nTirante < nymder(1) Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 4)
        ElseIf nTirante > nymder(1) Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 5)
        End If
    End If
    ElseIf ntipo(j) = "derecha" Then
        'para las secciones derechas
        nderecha = encuentraderecha(nym(j)): ny(j) = nTirante - nym(j)
        If nderecha = nymderecha Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 2)
        ElseIf nderecha < nymderecha Then
            If nTirante < nymder(nderecha + 1) Then
                nDAncho(j) = DAncho(nki(j), nkd(j), 2)
            ElseIf nTirante > nymder(nderecha + 1) Then
                nDAncho(j) = DAncho(nki(j), nkd(j), 5)
            End If
        End If
    ElseIf ntipo(j) = "izquierda" Then
        'para las secciones izquierdas
        nizquierda = encuentraizquierda(nym(j)): ny(j) = nTirante - nym(j)
        If nizquierda = nymizquierda Then
            nDAncho(j) = DAncho(nki(j), nkd(j), 1)
        ElseIf nizquierda < nymizquierda Then
            If nTirante < nymizq(nizquierda + 1) Then
                nDAncho(j) = DAncho(nki(j), nkd(j), 1)
            ElseIf nTirante > nymizq(nizquierda + 1) Then
                nDAncho(j) = DAncho(nki(j), nkd(j), 5)
            End If
        End If
    End If
    DAnchoTotal = DAnchoTotal + nDAncho(j)
Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function PerimetroTotal(secciones As Integer, nTirante As Double) As Double
'esta función calcula el perímetro de todas las secciones para un tirante determinado
    On Error GoTo error_handler
    If secciones = 1 Then
        ny(1) = nTirante - nym(1)
        nPerimetro(1) = Perimetro(nB(1), ny(1), nki(1), nkd(1), 0)
        PerimetroTotal = nPerimetro(1)
        Exit Function
    End If
    For j = 1 To secciones
        If ntipo(j) = "central" Then

```

```

'para las secciones centrales
ny(j) = nTirante - nym(j)
If nTirante < nymder(1) And nTirante < nymizq(1) Then nPerimetro(j) = Perimetro(nB(j), ny(j), nki(j), nkd(j), 0)
If nTirante > nymder(1) And nymizq(1) >= nymder(1) Then
nPerimetro(1) = Perimetro(nB(j), nymder(1), nki(j), nkd(j), 0)
If nTirante < nymizq(1) Then
nPerimetro(2) = Perimetro(Ancho(nB(j), nymder(1), nki(j), nkd(j), 0), nTirante - nymder(1), nki(j), _
nkd(j), 3)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2)
ElseIf nTirante > nymizq(1) Then
nPerimetro(2) = Perimetro(Ancho(nB(j), nymder(1), nki(j), nkd(j), 0), nymizq(1) - nymder(1), nki(j), _
nkd(j), 3)
nPerimetro(3) = Perimetro(Ancho(Ancho(nB(j), nymder(1), nki(j), nkd(j), 0), nymizq(1) - nymder(1), _
nki(j), nkd(j), 1), ny(j) - nymizq(1), nki(j), nkd(j), 5)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2) + nPerimetro(3)
End If
End If
If nTirante > nymizq(1) And nymizq(1) < nymder(1) Then
nPerimetro(1) = Perimetro(nB(j), nymizq(1), nki(j), nkd(j), 0)
If nTirante < nymder(1) Then
nPerimetro(2) = Perimetro(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nTirante - nymizq(1), nki(j), _
nkd(j), 4)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2)
ElseIf nTirante > nymder(1) Then
nPerimetro(2) = Perimetro(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nymder(1) - nymizq(1), nki(j), _
nkd(j), 4)
nPerimetro(3) = Perimetro(Ancho(Ancho(nB(j), nymizq(1), nki(j), nkd(j), 0), nymder(1) - nymizq(1), _
nki(j), nkd(j), 1), ny(j) - nymder(1), nki(j), nkd(j), 5)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2) + nPerimetro(3)
End If
End If
ElseIf ntipo(j) = "derecha" Then
'para las secciones derechas
nderecha = encuentraderecha(nym(j)): ny(j) = nTirante - nym(j)
If nderecha = nymderecha Then
nPerimetro(j) = Perimetro(nB(j), ny(j), nki(j), nkd(j), 2)
ElseIf nderecha < nymderecha Then
If nTirante < nymder(nderecha + 1) Then
nPerimetro(j) = Perimetro(nB(j), ny(j), nki(j), nkd(j), 2)
ElseIf nTirante > nymder(nderecha + 1) Then
nPerimetro(1) = Perimetro(nB(j), nymder(nderecha + 1) - nymder(nderecha), nki(j), nkd(j), 2)
nPerimetro(2) = Perimetro(Ancho(nB(j), nymder(nderecha + 1) - nymder(nderecha), nki(j), nkd(j), 2), _
nTirante - nymder(nderecha + 1), nki(j), nkd(j), 5)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2)
End If
End If
ElseIf ntipo(j) = "izquierda" Then
'para las secciones izquierdas
nizquierda = encuentraizquierda(nym(j)): ny(j) = nTirante - nym(j)
If nizquierda = nymizquierda Then
nPerimetro(j) = Perimetro(nB(j), ny(j), nki(j), nkd(j), 1)
ElseIf nizquierda < nymizquierda Then
If nTirante < nymizq(nizquierda + 1) Then
nPerimetro(j) = Perimetro(nB(j), ny(j), nki(j), nkd(j), 1)
ElseIf nTirante > nymizq(nizquierda + 1) Then
nPerimetro(1) = Perimetro(nB(j), nymizq(nizquierda + 1) - nymizq(nizquierda), nki(j), nkd(j), 1)
nPerimetro(2) = Perimetro(Ancho(nB(j), nymizq(nizquierda + 1) - nymizq(nizquierda), nki(j), nkd(j), _
1), nTirante - nymizq(nizquierda + 1), nki(j), nkd(j), 5)
nPerimetro(j) = nPerimetro(1) + nPerimetro(2)
End If
End If
End If

```

```

    End If
    PerimetroTotal = PerimetroTotal + nPerimetro(j)
Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

Function DPerimetroTotal(secciones As Integer, nTirante As Double) As Double

'esta función calcula la derivada del perímetro respecto del tirante de todas las secciones para un tirante determinado

```

On Error GoTo error_handler
For j = 1 To secciones
    If ntipo(j) = "central" Then
        'para las secciones centrales
        ny(j) = nTirante - nym(j)
        If nTirante < nymder(1) And nTirante < nymizq(1) Then
            nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 0)
        End If
        If nTirante > nymder(1) And nymizq(1) >= nymder(1) Then
            If nTirante < nymizq(1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 3)
            ElseIf nTirante > nymizq(1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 5)
            End If
        End If
        If nTirante > nymizq(1) And nymizq(1) < nymder(1) Then
            If nTirante < nymder(1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 4)
            ElseIf nTirante > nymder(1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 5)
            End If
        End If
    ElseIf ntipo(j) = "derecha" Then
        'para las secciones derechas
        nderecha = encuentraderecha(nym(j)): ny(j) = nTirante - nym(j)
        If nderecha = nymderecha Then
            nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 2)
        ElseIf nderecha < nymderecha Then
            If nTirante < nymder(nderecha + 1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 2)
            ElseIf nTirante > nymder(nderecha + 1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 5)
            End If
        End If
    ElseIf ntipo(j) = "izquierda" Then
        'para las secciones izquierdas
        nizquierda = encuentraizquierda(nym(j)): ny(j) = nTirante - nym(j)
        If nizquierda = nymizquierda Then
            nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 1)
        ElseIf nizquierda < nymizquierda Then
            If nTirante < nymizq(nizquierda + 1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 1)
            ElseIf nTirante > nymizq(nizquierda + 1) Then
                nDPerimetro(j) = DPerimetro(nki(j), nkd(j), 5)
            End If
        End If
    End If
    DPerimetroTotal = DPerimetroTotal + nDPerimetro(j)
Next

```



```

Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

```

Function RadioTotal(secciones As Integer) As Boolean
'esta función calcula el radio hidráulico de todas las secciones para un tirante determinado
On Error GoTo error_handler
For j = 1 To secciones
nRadio(j) = Radio(nArea(j), nPerimetro(j))
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

```

Function nTotal(secciones As Integer) As Boolean
'esta función calcula el coeficiente de rugosidad de Manning de todas las secciones para un tirante determinado
On Error GoTo error_handler
For j = 1 To secciones
If simple Then
nn(j) = nnm(j) * nFactorn(j)
Else
nn(j) = n(nRadio(j), g, nalfan(j), nc(j), nks(j), nFactorn(j))
End If
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

```

Function FactorTotal(secciones As Integer) As Double
'esta función calcula el factor de conducción de todas las secciones para un tirante determinado
On Error GoTo error_handler
For j = 1 To secciones
nFactor(j) = factor(nArea(j), nRadio(j), nn(j)): FactorTotal = FactorTotal + nFactor(j)
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

```

Function dnTotal(secciones As Integer) As Boolean
'esta función calcula la derivada del coeficiente de rugosidad de Manning respecto del tirante de todas las secciones para un
tirante determinado ( $\Lambda/n$ )(dn/dy)
On Error GoTo error_handler
For j = 1 To secciones
If simple Then
ndn(j) = 0
Else
ndn(j) = dn(nalfan(j), nn(j), g, nRadio(j), nAncho(j), nDPerimetro(j))
End If
Next
Exit Function
error_handler:
'en caso de error
Exit Function

```

End Function

Function Sigma1Total(secciones As Integer) As Double

'esta función calcula el valor de sigma1 de todas las secciones para un tirante determinado

On Error GoTo error\_handler

For j = 1 To secciones

nSigma1(j) = csigma1(nalfa(j), nFactor(j), nArea(j), nAncho(j), nRadio(j), nDPerimetro(j), ndn(j))

Sigma1Total = Sigma1Total + nSigma1(j)

Next

Exit Function

error\_handler:

'en caso de error

Exit Function

End Function

Function Sigma2Total(secciones As Integer) As Double

'esta función calcula el valor de sigma2 de todas las secciones para un tirante determinado

On Error GoTo error\_handler

For j = 1 To secciones

nSigma2(j) = csigma2(nalfa(j), nFactor(j), nArea(j)): Sigma2Total = Sigma2Total + nSigma2(j)

Next

Exit Function

error\_handler:

'en caso de error

Exit Function

End Function

Function Sigma3Total(secciones As Integer) As Double

'esta función calcula el valor de sigma3 de todas las secciones para un tirante determinado

On Error GoTo error\_handler

For j = 1 To secciones

nSigma3(j) = csigma3(nFactor(j), nArea(j), nAncho(j), nRadio(j), nDPerimetro(j), ndn(j))

Sigma3Total = Sigma3Total + nSigma3(j)

Next

Exit Function

error\_handler:

'en caso de error

Exit Function

End Function

Function coriolis(secciones As Integer) As Double

'esta función calcula el coeficiente de coriolis para la sección completa

On Error GoTo error\_handler

For j = 1 To secciones

alfa = nalfa(j) \* nFactor(j) ^ 3 / nArea(j) ^ 2: coriolis = coriolis + alfa

Next

coriolis = coriolis \* nAreaT ^ 2 / nFactorT ^ 3

Exit Function

error\_handler:

'en caso de error

Exit Function

End Function

Function verificaresultados() As Boolean

'esta función determina si el valor obtenido de tirante crítico para una iteración existe

On Error GoTo error\_handler

If existe = False Then nyc(i) = 0

If i < procesos And existe = True Then

If nTirante > nymdif(i + 1) Then

nyc(i) = 0

Else

```

        nyc(i) = nTirante
    End If
End If
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function Llenagridresult(grid As MSFlexGrid, procesos As Integer) As Boolean
'esta función presenta los resultados en el grid de la forma de resultados
    On Error GoTo error_handler
    With grid
        .Enabled = True: .Visible = True: .Rows = procesos + 1: .Cols = 2: .Row = 0: .Col = 0: .Text = "Intervalo"
        .Col = 1: .Text = "Tirante crítico (m)": .ColAlignment(0) = 2: .ColAlignment(1) = 0
        For i = 1 To .Rows - 1
            .Row = i: .Col = 0: .Text = Str(i): .Col = 1: .Text = Str(redondea(nyc(i)))
        Next
    End With
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function redondea(valor As Double) As Double
'esta función redondea los resultados hasta el diezmilésimo
    On Error GoTo error_handler
    If Val(Mid(Str(valor - Int(valor)), 7, 1)) > 4 Then
        redondea = Int(valor * 10000) / 10000 + 0.0001
    Else
        redondea = Int(valor * 10000) / 10000
    End If
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function determinapuntos() As Boolean
'esta función determina la cantidad de puntos a graficar en cada intervalo y la separación de los puntos el ultimo intervalo se
'obtiene a partir de la última berma más el valor máximo entre :el doble de la diferencia la última berma y el tirante crítico
'respectivo, si existe, y el valor del intervalo máximo se considera que la separación debe de ser la menor entre 5 cm y la
'décima parte del intervalo
    On Error GoTo error_handler
    ReDim puntos(procesos), diferencia(procesos)
    diferenciamax = 0: puntosT = 0
    For i = 1 To procesos
        If i < procesos Then
            diferencia(i) = Abs(nymdif(i + 1) - nymdif(i))
            If diferencia(i) > diferenciamax Then diferenciamax = diferencia(i)
        Else
            diferencia(i) = Abs((nymdif(i) - nyc(i)) * 2)
            If (diferencia(i) < diferenciamax And nymdif(i) <> 0) Or (nymdif(i) = 0 And i <> 1) Then
                diferencia(i) = diferenciamax
            End If
        End If
        puntos(i) = diferencia(i) / 0.05
        If Not (puntos(i) = Int(puntos(i))) Then puntos(i) = Int(puntos(i)) + 1
        If puntos(i) < 10 Then puntos(i) = 10
        If Not (nyc(i) = 0) Then puntos(i) = puntos(i) + 1
    Next
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

    puntosT = puntosT + puntos(i)
Next
If Chaudhry Then
    If Not (nyc(3) = 0) Then puntos(2) = puntos(2) + 1: puntosT = puntosT + 1
End If
'aquí se meten los puntos de Chaudhry
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function llenapuntos() As Boolean
'esta función genera los puntos a graficar, a partir de cada tirante obtiene un valor de energía específica
On Error GoTo error_handler
i = 0: tirante = 0: nTirante = 0: l = 2
ReDim XEnergiam1(puntosT), YTirantetem1(puntosT)
ReDim XEnergiam(puntosT), YTirantetem(puntosT)
ReDim XEnergia(puntosT), YTirante(puntosT)
For j = 1 To procesos
    If nyc(j) = 0 Then
        incremento = diferencia(j) / puntos(j)
    Else
        incremento = diferencia(j) / (puntos(j) - 1)
    End If
    seccionesproceso = encuentraseccionesproceso(nymdif(j))
    For K = 1 To puntos(j)
        If nyc(j) = 0 Or nTirante >= nyc(j) Or (nyc(j) <> 0 And Abs(nyc(j) - tirante) > incremento And nTirante < nyc(j)) Then
            tirante = tirante + incremento: nTirante = tirante
        Else
            nTirante = nyc(j)
            If Chaudhry And j = 2 Then j = 3
        End If
        i = i + 1
        nAreaT = AreaTotal(seccionesproceso, nTirante)
        nPerimetroT = PerimetroTotal(seccionesproceso, nTirante)
        RadioTotal seccionesproceso
        nRadioT = nAreaT / nPerimetroT
        nTotal seccionesproceso
        nFactorT = FactorTotal(seccionesproceso)
        nAlfaT = coriolis(seccionesproceso)
        nEnergia = nTirante + nAlfaT * Q ^ 2 / (2 * g * nAreaT ^ 2)
        XEnergiam1(i) = nEnergia: YTirantetem1(i) = nTirante
        XEnergiam(i) = nEnergia: YTirantetem(i) = nTirante
        XEnergia(i) = nEnergia: YTirante(i) = nTirante
    Next
Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function promedio(lista() As Double) As Double
'esta función obtiene el promedio de los valores de una lista dada
On Error GoTo error_handler
suma = 0
For i = 1 To UBound(lista)
    suma = suma + lista(i)
Next

```

```

    promedio = suma / UBound(lista)
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function desviacion(lista() As Double, promedio As Double) As Double
'esta función obtiene la desviación estándar de una serie de valores dados
    On Error GoTo error_handler
    suma = 0
    For i = 1 To UBound(lista)
        suma = suma + (Abs(promedio - lista(i))) ^ 2
    Next
    desviacion = Sqr(suma / (UBound(lista) - 1))
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function limpiamayores(promedio As Double, lista() As Double, lista2() As Double, lista3() As Double, lista4() As Double) _
As Boolean
'lista es el temporal de energia, lista2 es la lista definitiva de energía lista3 es el temporal de tirante, lista4 es la lista definitiva de
'tirante
    On Error GoTo error_handler
    menores = 0
    For i = 1 To UBound(lista)
        If lista(i) < promedio Then menores = menores + 1
    Next
    ReDim lista2(menores), lista4(menores)
    j = 0
    For i = 1 To UBound(lista)
        If lista(i) < promedio Then j = j + 1: lista2(j) = lista(i): lista4(j) = lista3(i)
    Next
    ReDim lista(menores), lista3(menores)
    For i = 1 To UBound(lista)
        lista(i) = lista2(i): lista3(i) = lista4(i):
    Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function mayorlista(lista() As Double) As Double
'esta función obtiene el valor más alto de una lista dada
    On Error GoTo error_handler
    mayorlista = 0
    For i = 1 To UBound(lista)
        If lista(i) > mayorlista Then mayorlista = lista(i)
    Next
Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

```

```

Function menorlista(lista() As Double) As Double
'esta función obtiene el valor menor de una lista dada
    On Error GoTo error_handler

```

```

    menorlista = mayorlista(lista())
    For i = 1 To UBound(lista)
        If lista(i) < menorlista Then menorlista = lista(i)
    Next
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function encuentra(lista() As Double, coord As Double) As Integer
'esta función encuentra el índice de un arreglo para un valor dado
    On Error GoTo error_handler
    encuentra = 0
    For i = 1 To UBound(lista)
        If coord > 0.1 Then
            If Abs(lista(i) - coord) / coord < 0.0001 Then encuentra = i: Exit Function
        Else
            If lista(i) = coord Then encuentra = i: Exit Function
        End If
    Next
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function verificacolor(j As Integer) As Double
'esta función determina si el valor dado corresponde a la altura de berma o a un tirante crítico
    On Error GoTo error_handler
    If encuentra(nyc, YTirante(j)) <> 0 Then
        verificacolor = vbBlue: YTirante(j) = redondea(YTirante(j))
    ElseIf encuentra(nymdif, YTirante(j)) <> 0 Then
        verificacolor = vbRed: YTirante(j) = redondea(YTirante(j))
    Else
        verificacolor = vbBlack
    End If
    Exit Function
error_handler:
'en caso de error
    Exit Function
End Function

Function minimo(uno As Double, dos As Double) As Double
'esta función obtiene el mínimo entre dos valores
    On Error GoTo error_handler
    minimo = 0
    If uno < dos And uno <> 0 Then
        minimo = uno
    ElseIf uno < dos And uno = 0 Then
        minimo = dos
    End If
    If dos < uno And dos <> 0 Then
        minimo = dos
    ElseIf dos < uno And dos = 0 Then
        minimo = uno
    End If
    If uno = 0 And dos = 0 Then minimo = 0
    Exit Function
error_handler:
'en caso de error

```

```
Exit Function
End Function
```

```
Function maximo(uno As Double, dos As Double) As Double
```

```
'esta función obtiene el mayor entre dos valores
```

```
On Error GoTo error_handler
```

```
maximo = 0
```

```
If uno > dos Then maximo = uno
```

```
If dos > uno Then maximo = dos
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function ajuste(inicio As Integer, fin As Integer) As Boolean
```

```
'esta función ajusta los puntos a graficar
```

```
On Error GoTo error_handler
```

```
totalptos = fin - inicio + 1
```

```
ReDim YTirantetem(totalptos), XEnergiamem(totalptos)
```

```
j = 0
```

```
For i = inicio To fin
```

```
    j = j + 1: YTirantetem(j) = YTirante(i): XEnergiamem(j) = XEnergia(i)
```

```
Next
```

```
ReDim YTirante(totalptos), XEnergia(totalptos)
```

```
For i = 1 To UBound(YTirante)
```

```
    YTirante(i) = YTirantetem(i): XEnergia(i) = XEnergiamem(i)
```

```
Next
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function llenagridfactor(grid As MSFlexGrid) As Boolean
```

```
'esta función vacía los valores de las iteraciones en una malla
```

```
On Error GoTo error_handler
```

```
With grid
```

```
    .ColWidth(1) = 600: .ColWidth(0) = 750: .Rows = UBound(nseccion) + 1: .Row = 0
```

```
    .Col = 0: .Text = "Sección": .Col = 1: .Text = "Factor":
```

```
    For i = 1 To UBound(nseccion)
```

```
        .Row = i: .Col = 0: .Text = Str(nseccion(i)): .Col = 1: .Text = Str(nFactor(i))
```

```
    Next
```

```
End With
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

```
Function encuentramayor(tipo As String) As Integer
```

```
'esta función encuentra el valor más alto para un tipo de sección, sea derecha o izquierda
```

```
On Error GoTo error_handler
```

```
For j = 2 To total
```

```
    If tipo = ntipo(j) Then encuentramayor = j
```

```
Next
```

```
Exit Function
```

```
error_handler:
```

```
'en caso de error
```

```
Exit Function
```

```
End Function
```

Function llenacombo(combo As ComboBox) As Boolean

'esta función llena un combo

```
On Error GoTo error_handler
Dim j As Integer
combo.Clear
For j = 1 To total
    combo.AddItem Str(nseccion(j))
Next
```

```
Exit Function
```

error\_handler:

'en caso de error

```
Exit Function
```

End Function

Function Chaudhrys() As Boolean

'esta función determina si el ejemplo analizar cumple con los requisitos para emplear el método de Chaudhry

```
On Error GoTo error_handler
Chaudhrys = False
If total = 3 And procesos = 2 Then
    If nki(1) = nkd(1) And ntipo(2) <> ntipo(3) And nB(2) = nB(3) And nym(2) = nym(3) And nki(2) = nkd(3) And _
        nki(3) = nkd(2) And nalfa(2) = nalfa(3) And nc(2) = nc(3) And nalfan(2) = nalfan(3) And nalfa(2) = 1 And nalfa(1) _
            = 1 And nks(2) = nks(3) And nnm(2) = nnm(3) Then Chaudhrys = True
```

```
Else
```

```
    Chaudhrys = False
```

```
End If
```

```
Exit Function
```

error\_handler:

'en caso de error

```
Exit Function
```

End Function

Function calculaCmax() As Double

'esta función obtiene el valor máximo de la variable C para el método de Chaudhry

```
On Error GoTo error_handler
Dim Tirante0 As Double, Tirante1 As Double
Dim C0 As Double, C1 As Double
calculaCmax = 0: nTirante = nyndif(2) + 0.00001: Tirante0 = nTirante: C0 = calculaC(nTirante)
While calculaCmax = 0
    nTirante = nTirante + 0.00001: Tirante1 = nTirante: C1 = calculaC(nTirante)
    If C1 < C0 Then
        calculaCmax = C0: nyCmax = nTirante
    End If
    C0 = C1: Tirante0 = Tirante1
Wend
```

```
Exit Function
```

error\_handler:

'en caso de error

```
Exit Function
```

End Function

Function calculaC(nTirante As Double) As Double

'esta función obtiene el valor de la variable C para el método de Chaudhry

```
On Error GoTo error_handler
ReDim nBeta(3), nBetap(3), nDFactor(3)
nAm = Area(nB(1), nyndif(2), nki(1), nkd(1), 0)
nTm = Ancho(nB(1), nyndif(2), nki(1), nkd(1), 0)
nAreaT = AreaTotal(3, nTirante)
nAnchoT = AnchoTotal(3, nTirante)
nDAnchoT = DAnchoTotal(3, nTirante)
```



```

nPerimetroT = PerimetroTotal(3, nTirante)
nDPerimetroT = DPerimetroTotal(3, nTirante)
RadioTotal 3
nRadioT = nAreaT / nPerimetroT
nTotal 3
nFactorT = FactorTotal(3)
nDFactorT = DFactorTotal(3)
nBetaT = BetaTotal(3)
nBetapT = nBetaT * nAnchoT / nAreaT - 2 * nBetaT / nFactorT * nDFactorT + nAreaT / (3 * nFactorT ^ 2) * (7 *
^ 2 * nRadio(2) * nDPerimetro(2))
calculaC = nAm ^ 3 / (nAreaT ^ 2 * nTm) * (nBetaT * nAnchoT / nAreaT - nBetapT)
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

Function BetaTotal(secciones As Integer) As Double
'esta función calcula el coeficiente de Boussinesq para la sección completa
On Error GoTo error_handler
For j = 1 To secciones
Beta = nFactor(j) ^ 2 / nArea(j): BetaTotal = BetaTotal + Beta
Next
BetaTotal = BetaTotal * nAreaT / nFactorT ^ 2
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

Function DFactorTotal(secciones As Integer) As Double
'esta función calcula la variación del Factor de conducción con respecto del tirante
On Error GoTo error_handler
For j = 1 To secciones
nDFactor(j) = 5 * nFactor(j) / 3 * nAncho(j) / nArea(j) - 2 / 3 * nFactor(j) / nArea(j) * nRadio(j) * nDPerimetro(j)
FactorTotal = DFactorTotal + nDFactor(j)
Next
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

Function encuentranyc(nTirante As Double, c As Double) As Double
'esta función encuentra un tirante para el valor de C del método de Chaudhry a partir de los datos respecto a un cierto gasto
On Error GoTo error_handler
Dim fin As Boolean
fin = False: tirantetem1 = nTirante: ctem1 = calculaC(nTirante): error1 = Abs(c - ctem1) / c
While fin = False
nTirante = nTirante + 0.00001: tirantetem2 = nTirante: ctem2 = calculaC(nTirante): error2 = Abs(c - ctem2) / c
If error1 < error2 Then encuentranyc = nTirante: fin = True
tirantetem1 = tirantetem2: ctem1 = ctem2: error1 = error2
Wend
Exit Function
error_handler:
'en caso de error
Exit Function
End Function

```

Function gastomax() As Double

'está función determina el gasto máximo con el cual se presentan tirantes críticos múltiples para una sección simétrica con dos llanuras de inundación

```
On Error GoTo error_handler
nAm = Area(nB(1), nymdif(2), nki(1), nkd(1), 0)
nTm = Ancho(nB(1), nymdif(2), nki(1), nkd(1), 0)
gastomax = Sqr(nAm ^ 3 * g / (nalfa(1) * nTm))
error_handler:
'en caso de error
Exit Function
End Function
```

Function calculaFFmmax() As Double

'esta función obtiene el valor máximo de la variable F/Fm

'para el método de Blalock

```
On Error GoTo error_handler
Dim Tirante0 As Double, Tirante1 As Double
Dim ffm0 As Double, ffm1 As Double
calculaFFmmax = 0: nTirante = nymdif(2) * 1.001: Tirante0 = nTirante: ffm0 = calculaF(nTirante)
While calculaFFmmax = 0
nTirante = nTirante + 0.00001: Tirante1 = nTirante: ffm1 = calculaF(nTirante)
If ffm1 < ffm0 Then calculaFFmmax = ffm0
ffm0 = ffm1: Tirante0 = Tirante1
Wend
nTirante = nymdif(2)
nAm = Area(nB(1), nymdif(2), nki(1), nkd(1), 0)
nTm = Ancho(nB(1), nymdif(2), nki(1), nkd(1), 0)
Fm = Q / Sqr(nAm ^ 3 * g / (nalfa(1) * nTm))
calculaFFmmax = calculaFFmmax / Fm
Exit Function
error_handler:
'en caso de error
Exit Function
End Function
```

Function calculaF(nTirante As Double) As Double

'esta función obtiene el valor del número de Froude según Blalock

```
On Error GoTo error_handler
nAreaT = AreaTotal(3, nTirante)
nAnchoT = AnchoTotal(3, nTirante)
nDAnchoT = DAnchoTotal(3, nTirante)
nPerimetroT = PerimetroTotal(3, nTirante)
nDPerimetroT = DPerimetroTotal(3, nTirante)
RadioTotal 3
nRadioT = nAreaT / nPerimetroT
nTotal 3
nFactorT = FactorTotal(3)
nDFactorT = DFactorTotal(3)
dnTotal 3
nSigma1T = Sigma1Total(3)
nSigma2T = Sigma2Total(3)
nSigma3T = Sigma3Total(3)
nAlfaT = coriolis(3)
calculaF = Sqr((nSigma2T * nSigma3T / nFactorT - nSigma1T) * (Q ^ 2 / 2 / g / nFactorT ^ 3))
Exit Function
error_handler:
'en caso de error
Exit Function
End Function
```

## BIBLIOGRAFÍA

- Aldama, Álvaro A. y Ocón Alfredo R., “¿Qué es el flujo crítico?”, XVII Congreso Nacional de Hidráulica, Monterrey, N.L., 2002.
- Blalock, M. E. Y Sturm T. W., 1981, “Minimum specific energy in compound channel”, ASCE J Hydraulics Division, 107 (HY6):699-717
- Chaudhry M., Hanif, “Open channel flow”, Prentice Hall, EUA, 1993.
- Chaudhry M. Hanif y Bhallamudi S. Murty, 1988, “Computation of critical depth in symmetrical compound channels” J. Hydraulic Research, IARH 26(4).
- Comisión Nacional del Agua, “Estadísticas del agua en México”, CNA, México, 2003.
- Conde C., José, “Encauzamiento del río Tijuana”, Facultad de Ingeniería, U.N.A.M., 1962.
- Encyclopedia Americana, Vol. XIV página 541 y Vol. XXVIII página 432, 1995.
- Estrella V., Rafael E., “Diseño y procedimientos de construcción del canal para riego del proyecto Yaque del sur Azua, República Dominicana”, Facultad de Ingeniería, U.N.A.M., 1978.
- Levi, Enzo, “El agua según la ciencia”, CONACYT, México, D.F., 1989.
- Sotelo Ávila, Gilberto, “Hidráulica de canales”, Facultad de Ingeniería, U.N.A.M., 2002.
- Sotelo Ávila, Gilberto, “Algoritmo del método de Blalock Sturm para determinar los tirantes críticos múltiples en canales compuestos”, Ingeniería Hidráulica en México”, Vol. XIII, Núm, 1, páginas 51 a 60, enero-abril de 1998
- Sturm, T. W. y A. Sadiq, 1996, “Water surface profiles in compound channel with multiple critical depths”, ASCE, J. Hydraulic Engineering 122(12):703-709.
- Ven Te Chow, Hidráulica de canales abiertos", Mc Graw Hill, Santa Fe de Bogotá, Colombia, 1994.