Modelos numéricos para redes fluviales

Jacovkis, Pablo Miguel

1988

Tesis presentada para obtener el grado de Doctor en Ciencias Matemáticas de la Universidad de Buenos Aires

MODELOS NUMERICOS PARA REDES FLUVIALES

Trabajo de tesis presentado por

Pablo Miguel Jacovkis

para optar al grado de
Doctor en Matemáticas de la
Facultad de Ciencias Exactas y Naturales de la
Universidad de Buenos Aires

Director de Tesis: Prof. Dr. Juan Pedro Milaszewicz

1988

Lugar de trabajo:
Departamento de Matemática, Facultad de Ingeniería, UBA
Departamento de Computación, Facultad de Ciencias Exactas y Naturales, UBA

RESUMEN

En este trabajo se resuelven numéricamente por métodos implícitos en diferencias finitas las ecuaciones hiperbólicas casilineales de Saint-Venant del flujo no estacionario unidimensional gradualmente variado de aguas poco profundas sobre superficie libre y fondo fijo para sistemas fluviales de tipo arborescente (cuencas) y deltaico, en que se tienen que considerar las condiciones de compatibilidad en los puntos de confluencia (condiciones de Stoker). Linealizando y discretizando según el método de Preissmann demostraremos que el problema se puede reducir, en cada intervalo temporal de cálculo, a la resolución de un sistema lineal de la forma **Ax=b**, donde la matriz **A** tiene una estructura especial rala que facilita su solución; **A** y **b** dependen del intervalo temporal, no así la estructura de ceros de **A**. Dicha estructura es más complicada en el caso deltaico que en el arborescente, por lo cual es conveniente tratarlos por separado. Se describen además diversos experimentos numéricos realizados, tanto para el caso de sistemas fluviales arborescentes como de redes complejas deltaicas, incluyendo comparaciones con soluciones analíticas conocidas.

ABSTRACT

In this work we solve by implicit finite difference numerical methods the Saint-Venant quasilinear hyperbolic equations representing the one-dimensional, non-steady, gradually varied, open channel flow of shallow waters with fixed beds on arborescent fluvial systems (basins) and on deltaic fluvial systems, where compatibility conditions (Stoker conditions) at the junctions must be considered. If we apply the Preissmann method in order to linearize and discretize the equations, we prove that, for each time step, the problem can be reduced to solving a linear system **Ax=b**, and the matrix **A** has a special sparse structure that simplifies its solution; **A** and **b** depend on the time step, but the zero/nonzero structure of **A** does not. This structure is more complicated in the deltaic case than in the arborescent one; thus, it is convenient to treat the two cases separately. We describe several numerical experiments for both the arborescent and the deltaic case. Comparisons with known analytical solutions are included.

Agradecimientos

Para llevar a cabo esta tesis conté con el apoyo de Pedro Milaszewicz, que permanentemente me prestó toda su colaboración, y, en la fase final, una muy considerable cantidad de su tiempo, y cuyas observaciones, críticas y sugerencias, tanto respecto del contenido como de la forma, me fueron de gran utilidad; y con el de Rosita Wachenchauzer, cuyo constante estímulo afectivo fue fundamental para que emprendiera (y terminara) esta tarea, y con quien, por otra parte, discutí fructíferamente varios puntos concretos. Rosita leyó el trabajo varias veces haciéndome siempre comentarios valiosos.

También debo mencionar el apoyo de Leonardo Moledo y Rubén Rosales, que me alentaron insistentemente y con regularidad implacable, uno en forma personal, el otro epistolarmente.

Con Alberto Grunbaum, Luis Reyna, Cora Sadosky, Héctor Sussmann y Daniel Szyld tuve varias conversaciones tremendamente importantes, ya sea en lo profesional o por la confianza que me manifestaron.

Finalmente debo mencionar a Mario Gradowczyk, que hace ya bastantes años me enseñó a trabajar en modelos matemáticos en hidrodinámica y a entusiasmarme con ellos, y con quien desarrollé muchos modelos, cada uno con una dificultad distinta.

A todos ellos, muchas gracias. Y también a Lucas Potenze, quien dibujó todas las figuras.

INDICE

1. INTRODUCCION

El flujo no estacionario gradualmente variado unidimensional de aguas poco profundas sobre superficie libre y con fondo fijo (flujo en canales abiertos, o en ríos, considerando solamente la coordenada espacial en el sentido del eje longitudinal del canal o río) se modeliza mediante un sistema de ecuaciones diferenciales hiperbólicos casilineales, llamadas ecuaciones de Saint-Venant de la hidráulica fluvial, planteado en forma de problema mixto con condiciones iniciales y de contorno. Estas ecuaciones deben resolverse numéricamente; pues no ha podido determinarse en general una solución analítica.

Para tramos simples unidimensionales de ríos o cauces fluviales, los métodos en diferencias finitas implícitos conducen usualmente a la resolución,para cada intervalo temporal de cálculo, de un sistema lineal $Ax=b$, con A matriz banda no simétrica. Por consiguiente, se pueden usar los métodos directos de solución de sistemas lineales con matrices banda, que requieren un número de operaciones igual a $O(n)$, siendo n el orden de la matriz, en lugar de ser igual a $O(n^3)$ -lo cual sería el caso si la matriz fuera llena -, y usan escaso espacio de memoria de computador. Por ese motivo estos métodos están muy difundidos en modelizaciones concretas. En particular, el método más utilizado en hidráulica fluvial unidimensional para ríos y canales con régimen subcrítico es el método de diferencias finitas implícito de Preissmann.

Pero si el sistema modelizado que se quiere resolver no es un tramo simple, sino una red fluvial con afluentes, efluentes, bifurcaciones (debido a la existencia de islas u otros obstáculos en el curso de agua), la situación se complica: la matriz A será rala, pero ya no banda. Interesa por consiguiente tratar de encontrar estructuras generales de la matriz A (es decir, que no dependan de la red fluvial en particular que se modeliza) para poder utilizar métodos directos rápidos y con poco uso de memoria de computador, o tratar de usar métodos iterativos que, si bien son más lentos para este tipo de problemas, aprovechan totalmente la ralidad de la matriz. El problema tiene aplicación práctica inmediata: poder resolverlo en forma eficiente significa

poder implementar algoritmos complejos en computadoras personales (e incluso en "computadoras hogareñas"), que pueden ser muy onerosos (o directamente no implementables) si es necesario resolver sistemas con matrices A de estructura desconocida. Y se evita así también tener que formular el problema en más de una dimensión espacial, caso para el que los problemas antes mencionados se multiplican, además de que la información necesaria complementaria suele no estar disponible.

El objetivo de este trabajo es resolver el problema mencionado: se demuestra aquí que para el método de Preissmann se puede llevar la matriz A a una forma con una estructura especial que facilita enormemente la resolución del sistema inicial Ax=b en cada paso de tiempo. Además, a partir de la teoría desarrollada, se implementan los correspondientes modelos computacionales, tanto en forma simplificada, con lo cual se realizan comparaciones con soluciones analíticas, como en forma completa, con lo cual se realizan experimentos numéricos de casos análogos a casos reales.

Los modelos fluviales unidimensionales analizados se clasifican en dos tipos: los de tipo *arborescente* (cauce principal, afluentes, afluentes de afluentes, etc; la topología de estos modelos puede describirse a través de la noción de arborescencia o árbol dirigido en teoría de grafos), y los de tipo *deltaico* (puede haber bifurcaciones que se cierran; la topología puede describirse a través de grafos dirigidos débilmente conexos sin ciclos). Si bien el caso arborescente es un caso particular del deltaico, enfocamos ambos casos de modo distinto pues el caso arborescente permite una solución especial más eficiente. En particular, los grafos que usaremos para arborescencias estarán dirigidos con orientación inversa a los que usaremos para redes deltaicas; la orientación de éstos será "desde aguas arriba hacia aguas abajo", y la de aquéllos "desde aguas abajo hacia aguas arriba".

En el caso arborescente reducimos el problema a uno que involucra una matriz A triangular superior con a lo sumo dos elementos no nulos por fila de las mismas características que para tramos unidimensionales, independientemente de la complejidad de

la red; en el caso deltaico, aparentemente la matriz no puede llevarse a ninguna matriz banda cuyo ancho no dependa de la red fluvial considerada. En este caso, demostramos que la matriz A se puede particionar en submatrices con estructuras que permiten resolver el sistema por un método directo que usa una cantidad reducida de tiempo y memoria de computadora, siendo por consiguiente muy eficiente.

Los modelos obtenidos fueron implementados en computadora, usando como método numérico en diferencias finitas el método de Preissmann, como ya mencionamos, pero las demostraciones de los teoremas en que se basan los algoritmos utilizados son generales para los esquemas numéricos implícitos de tipo "caja de cuatro puntos", y se pueden extender, en forma natural, a modelos de flujo a presión (cañerías) en que el diseño de redes deltaicas es imprescindible y, con modificaciones, a otros esquemas implícitos.

Calculamos estimaciones de velocidad de procedimientos, y exhibimos diversos experimentos numéricos realizados, tanto para comparar los resultados con soluciones teóricas simplificadas, como para comparar la consistencia de los modelos ante distintas condiciones de contorno y para analizar situaciones interesantes en hidráulica fluvial. El método directo diseñado se comparó con algunos métodos iterativos que también se describen (el método de las proyecciones de Kacmarcz y un método de las proyecciones modificado) que son desechados por ser completamente ineficientes.

El plan de este trabajo es el siguiente: la primera parte consiste en el planteo de la teoría matemática unidimensional de aguas poco profundas, y su extensión a confluencias de cauces fluviales, y métodos numéricos para su resolución en tramos fluviales simples.

En la sección 2 se formula el modelo matemático que representa el flujo unidimensional gradualmente variado, no estacionario, con superficie libre y con fondo fijo, de aguas poco profundas a través de canales o cauces fluviales de sección transversal arbitraria (pero razonablemente "suave"). Este es un modelo de ecuaciones diferenciales en derivadas parciales hiperbólicas

casilineales (ecuaciones de Saint-Venant), que además, para modelizaciones matemáticas de aplicación práctica debe formularse con condiciones iniciales y de contorno, es decir, no como problema de Cauchy sino como problema mixto. Así se modelizan tramos simples unidimensionales.

En la sección 3 se plantean las ecuaciones de compatibilidad (de Stoker) de los puntos de confluencia cuando se tienen tramos afluentes o efluentes.

En la sección 4 se indican los métodos numéricos más usuales para resolver prácticamente las ecuaciones de Saint-Venant.

En la sección 5 se describe en detalle el método de Preissmann, que usamos en el resto del trabajo.

En la sección 6 se analiza la consistencia, convergencia y estabilidad del esquema de Preissmann en el caso de sistemas hiperbólicos lineales homogéneos con coeficientes constantes y con condiciones puramente iniciales, usando la teoría de Lax y Richtmyer, así como el criterio de estabilidad de von Neumann.

En la sección 7 se indica cómo, usando el método de Preissmann en tramos simples unidimensionales, se llega a un sistema de la forma $Ax=b$ en cada intervalo de tiempo de cálculo, donde A es una matriz banda, y cómo se tridiagonaliza el sistema para luego resolverlo mediante un algoritmo de "barrido doble".

En la sección 8 se estudia la matriz A para evaluar heurísticamente su condicionamiento numérico.

La segunda parte del trabajo consiste en el estudio de las redes fluviales arborescentes:

En la sección 9 se definen con precisión algunos conceptos necesarios para generalizar la teoría anterior a estructuras fluviales arborescentes, en el marco de los conceptos de árboles dirigidos o arborescencias. Estos conceptos se utilizarán para establecer una numeración conveniente de los puntos de discretización de la red.

En la sección 10 se demuestra cómo se puede llevar el sistema lineal a un sistema triangular (con a lo sumo dos elementos no nulos sobre cada fila), que se resuelve, por consiguiente, mediante el barrido ascendente del barrido doble. Para ello se hará uso de la numeración obtenida en la sección 9.

En la sección 11, siempre en el caso lineal simplificado, se comparan soluciones teóricas del modelo con estructura arborescente diseñado con los resultados numéricos obtenidos con nuestro algoritmo, y se observa que se producen los mismos fenómenos ya señalados por otros autores para el caso de tramos simples.

En la sección 12 se realizan diversos experimentos numéricos con canales de estructura arborescente, y se extraen las correspondientes conclusiones.

La tercera parte de este trabajo consiste en el estudio del modelo deltaico.

En la sección 13 se describen las redes fluviales complejas con estructura deltaica.

En la sección 14 se formaliza la discretización para redes fluviales deltaicas, usando herramientas de la teoría de grafos dirigidos. Esta teoría se usará para establecer una numeración conveniente de los puntos de discretización de la red deltaica.

En la sección 15 se demuestra que es posible plantear el sistema $Ax=b$ en cada intervalo temporal por medio de una matriz A particionada con una determinada estructura que nos servirá para la resolución eficiente del sistema. Esa estructura permite un significativo ahorro en memoria para el almacenamiento de la matriz. Para obtener esta estructura se hará uso de la numeración obtenida en la sección 14.

En la sección 16 se describe el algoritmo de resolución del sistema, y se dan cotas para la memoria involucrada y el número de operaciones necesarios, que resultan muy inferiores a los requeridos si se aplica un método de Gauss directamente.

En la sección 17 se lleva a cabo un análisis de la consistencia, como modelo, del modelo deltaico implementado, incluyendo en dicho análisis la comparación de los resultados numéricos con resultados analíticos obtenidos usando un modelo lineal simplificado.

En la sección 18 se detallan los experimentos numéricos realizados con un modelo deltaico y las conclusiones resultantes.

La sección 19 consiste en las conclusiones del trabajo.

Este trabajo incluye tres apéndices:

En el primer apéndice se aplica un método iterativo de resolución de sistemas lineales al modelo deltaico, y luego una variante del mismo, para comprobar su impractibilidad. Los métodos iterativos usados son el método de las proyecciones de Kacmarz y una modificación del mismo, de convergencia teóricamente asegurada. En la sección A1 se describe el método de Kacmarz, y su implementación en nuestro sistema lineal para redes deltaicas, almacenando la matriz rala A con un sistema de punteros eficiente. En la sección A2 se detallan los experimentos numéricos realizados y sus tiempos de cálculo, que aconsejan desechar este método. En la sección A3 se implementa una modificación del método de las proyecciones, con el mismo sistema de punteros. En la sección A4 se describen los experimentos numéricos realizados, que no indican ninguna mejoría respecto de los del método de Kacmarz original.

En el segundo apéndice se analizan las propiedades conservativas del método de Preissmann en general, y se compara la "conservación numérica" con la conservación teórica. Además, se exhibe un ejemplo de inestabilidad que no se presenta en sistemas lineales y se efectúan diversas pruebas de consistencia de los modelos.

En el tercer apéndice se comentan algunos problemas computacionales y de modelización encontrados.

Las llamadas se refieren a notas que están al final del trabajo, antes de la bibliografía. Se incluyen también, después de las figuras, los listados de los programas usados, escritos en PASCAL.

PRIMERA PARTE

2. LAS ECUACIONES DE SAINT-VENANT DE LA HIDRODINAMICA

El flujo impermanente (no estacionario) unidimensional de un fluido con superficie libre a lo largo de un canal o cauce fluvial poco profundo no prismático de sección transversal de forma arbitraria sobre fondo fijo se describe aproximadamente mediante el sistema de ecuaciones diferenciales hiperbólicas casilineales de Saint-Venant:

$$B \frac{\partial Z}{\partial t} + \frac{\partial Q}{\partial x} = 0 \qquad (1.1)$$

(ecuación de conservación de la masa)

$$\frac{1}{g} \frac{\partial}{\partial t} \left(\frac{Q}{S}\right) + \frac{\partial}{\partial x} \left(\frac{Q^2}{2gS^2}\right) + \frac{\partial Z}{\partial x} + \frac{Q|Q|}{D^2} = 0 \qquad (1.2)$$

(ecuación de cantidad de movimiento)

a las cuales hay que agregar las condiciones iniciales

$$Z(x,t_o)=Z_o(t) \qquad Q(x,t_o)=Q_o(t) \qquad (2)$$

donde t indica la variable temporal a partir de un instante inicial t_o, x indica la variable espacial a lo largo del eje longitudinal del cauce, g la aceleración de la gravedad, Q = Q(x,t) el caudal, Z = Z(x,t) la cota o nivel de la superficie del flujo respecto de un plano de referencia, S = S(Z(x,t),x) el área mojada de la sección transversal al flujo, B = B(Z(x,t),x) = dS/dZ el ancho superficial de la superficie mojada y D =D(Z(x,t),x) el coeficiente de conducción que mide la rugosidad del lecho del cauce e indica la resistencia por fricción (Ver figuras 1 y 2). Por comodidad, supondremos de ahora en adelante t_o=0.

Si se considera que el río o cauce fluvial recibe aportes laterales de caudal, por ingreso de un afluente o canal (o pierde caudal, por derivación lateral para riego o agua potable, por ejemplo) la ecuación (1.1) de conservación de la masa toma la forma

$$B\,\frac{\partial Z}{\partial t} + \frac{\partial Q}{\partial x} = q \qquad (1.1')$$

donde q = q(x,t) es el aporte lateral de caudal - conocido - por unidad de longitud x. El signo de q indica si se trata de una inyección (signo positivo) o de una extracción de caudal (signo negativo).

Estas ecuaciones son la generalización, para cauces fluviales o canales no prismáticos de sección transversal S irregular (si el canal es prismático valdrá S=S(Z(x,t))) del caso más sencillo de las ecuaciones unidimensionales de aguas poco profundas, en el cual la sección transversal es rectangular de ancho constante unitario: En este caso la ecuación (1.1) se escribe directamente

$$\frac{\partial h}{\partial t} + \frac{\partial Q}{\partial x} = 0 \qquad (3.1)$$

o, teniendo en cuenta que Q=hv

$$\frac{\partial h}{\partial t} + v\frac{\partial h}{\partial x} + h\frac{\partial v}{\partial x} = 0 \qquad (3.1')$$

siendo h = h(x,t) la altura desde el lecho del canal a la superficie libre y v=v(x,t) la velocidad del fluido. Si y = y(x) es la cota o nivel desde el mismo plano de referencia antes mencionado (ver nuevamente figuras 1 y 2) al lecho del canal, será Z(x,t) = h(x,t) + y(x) (si el lecho fuera móvil, es decir, variable en el tiempo, sería y = y(x,t)) o sea

$$\frac{\partial Z}{\partial x} = \frac{\partial h}{\partial x} + \frac{\partial y}{\partial x}$$

y si denominamos S_c a la pendiente de fondo del lecho será

$$\frac{\partial y}{\partial x} = - S_c$$

y, por consiguiente,

$$\frac{1}{g}\frac{\partial}{\partial t}\left(\frac{Q}{h}\right) + \frac{\partial}{\partial x}\left(\frac{Q^2}{2gh^2}\right) + \frac{\partial h}{\partial x} + \frac{\partial y}{\partial x} = -\frac{Q|Q|}{D^2}$$

Multiplicando por g y derivando

$$\frac{1}{h}\frac{\partial Q}{\partial t} - \frac{Q}{h^2}\frac{\partial h}{\partial t} + \frac{Q}{h}\left[\frac{\partial Q}{\partial x}/h - \frac{Q}{h^2}\frac{\partial h}{\partial x}\right] + g\frac{\partial h}{\partial x} = g\left(S_c - \frac{Q|Q|}{D^2}\right)$$

Multiplicando por h y usando que $\partial h/\partial t + \partial Q/\partial x = 0$

$$\frac{\partial Q}{\partial t} + 2\frac{Q}{h}\frac{\partial Q}{\partial x} - \frac{Q^2}{h^2}\frac{\partial h}{\partial x} + \frac{\partial}{\partial x}\left(\frac{gh^2}{2}\right) = gh\left(S_c - \frac{Q|Q|}{D^2}\right)$$

y en definitiva

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}(Q^2/h + gh^2/2) = gh(S_c - Q|Q|/D^2) \qquad (3.2)$$

o

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + g\frac{(h+y)}{\partial x} = -gv|v|h^2/D^2) \qquad (3.2')$$

Tanto para el sistema (1) como para el sistema más simple (3) es necesario suponer que la presión p varía hidrostáticamente sobre la vertical, es decir

$$p(z) = g\rho(h-z)$$

donde z es la coordenada vertical medida desde el lecho, y ρ es la densidad del líquido (Ver figura 1). Además, se considerará que el ángulo ψ que el eje x hace con la horizontal (ver figura 2) es lo suficientemente pequeño como para aproximar el seno de dicho ángulo por la pendiente S_c. O sea este modelo no vale para ríos y cauces de mucha pendiente, hecho que a veces no se tiene en cuenta. Naturalmente, dado que consideramos un flujo unidimensional, suponemos también que la velocidad del fluido es constante sobre la vertical (y sobre el ancho) en cada punto x.

La deducción detallada de estas fórmulas, y las condiciones bajo las cuales son efectivamente una aproximación al modelo físico considerado pueden verse en Stoker [1957], capítulo 2 (especialmente sección 2) o capítulo 10, sección 1 (para las ecuaciones (3.1) y (3.2)) y capítulo 11, sección 1 (para las ecuaciones (1.1) y (1.2)).

Con respecto a las ecuaciones (3), Stoker usa las variables η y h y esta última tiene un significado distinto al nuestro (η es nuestra Z y h es nuestro -y). Las ecuaciones (3.1´) y 3.2´) corresponden a las 2.2.12 y 2.2.11 de Stoker (que desprecia el término $Q|Q|/D^2$). Con respecto a las ecuaciones (1), Stoker cambia de notación en el capítulo 11: su variable y tiene el significado de nuestra h y h el significado de nuestra Z; sus ecuaciones 11.1.8 y 11.1.6 equivalen, con la salvedad de que en ellas está incluido un aporte lateral de caudal q (como en nuestra ecuación (1.1´), a nuestras ecuaciones (1.1) y (1.2).

En Chow [1959], capítulos 5 y 6, se analizan, además del coeficiente de conducción D, otros coeficientes usados para la estimación del término de resistencia. En Mahmood y Yevjevich [1975] se presenta, comentada, una completísima reseña bibliográfica actualizada a la fecha de dicha publicación. También podemos mencionar especialmente el clásico libro de Courant y Friedrichs [1948], el claro y actualizado, aunque no tan detallado, texto de Ockendon y Tayler [1983], capítulo 3, sección 1, y el artículo "fundacional" de Stoker [1948], que contiene parte del material que después figuraría en su ya citado libro, y contiene además como apéndice la deducción de Friedrichs [1948] de las ecuaciones de aguas poco profundas que Stoker adopta.

El sistema (3) con condiciones iniciales

$$h(x,0)=h_o(x) \qquad Q(x,0)=Q_o(x) \qquad (4)$$

está escrito en forma conservativa no homogénea, según la terminología de Rozdestvenskii y Janenko [1983], capítulo 1, sección 5 (los autores norteamericanos llaman "forma conservativa" solamente a la homogénea). En forma vectorial equivale a

$$\frac{\partial\, w(x,t)}{\partial t} + \frac{\partial\, F(w(x,t))}{\partial x} = c(w(x,t),x) \qquad (5)$$

con condiciones iniciales $w(x,0)=w_o(x)$ (6)

siendo

$$w = \begin{bmatrix} h \\ Q \end{bmatrix} \qquad F = \begin{bmatrix} Q \\ Q^2/h + gh^2/2 \end{bmatrix} \qquad c = \begin{bmatrix} 0 \\ gh(S_c - Q|Q|/D^2) \end{bmatrix}$$

La matriz A=dF/dw dada por

$$A = \begin{bmatrix} 0 & 1 \\ -Q^2/h^2+gh & 2Q/h \end{bmatrix}$$

tiene autovalores $\lambda_{1,2} = Q/h \pm \sqrt{gh}$, o sea, son reales y distintos para cualquier valor de la solución w con segunda coordenada estrictamente positiva (h > 0). Por consiguiente, el sistema diferencial (5) con condiciones iniciales (6) es, si h es estrictamente positivo, un sistema de dos ecuaciones diferenciales en derivadas parciales hiperbólico casilineal de primer orden. Más aun: si se desprecia el término de la derecha de (3.1), el sistema vectorial (5) queda homogéneo, y se ve fácilmente que es genuinamente no lineal en el sentido de Lax (ver Lax [1973], sección 5), es decir, vale que los autovalores no son funciones constantes de w:

$$\text{grad}\,(Q/h \pm \sqrt{gh}\,) = ((-Q/h^2 \pm (1/2)\sqrt{g/h}\,),\ 1/h\,) \neq 0$$

y los gradientes de los autovalores no son ortogonales a sus autovectores a derecha. En efecto, los autovectores a derecha son

$$(1,\ Q/h \pm \sqrt{gh}\,)$$

y vale, si $\langle\ ,\ \rangle$ es el producto escalar

$$\langle (1,\ Q/h + \sqrt{gh}\,)\ ,\ (-Q/h^2 + (1/2)\sqrt{g/h}\,,\ 1/h)\ \rangle =$$

$$- Q/h^2 + (1/2)\sqrt{g/h} + Q/h^2 + \sqrt{g/h} = 3/2\sqrt{g/h} \neq 0$$

y análogamente

$$\langle (1,\ Q/h - \sqrt{gh}\ )\ ,\ (-Q/h^2 - (1/2)\sqrt{g/h}\ ,\ 1/h)\ \rangle =$$

$$-Q/h^2 - (1/2)\sqrt{g/h} + Q/h^2 - \sqrt{g/h} = -(3/2)\sqrt{g/h} \neq 0$$

Del mismo modo, calculando las derivadas respecto de t de los cocientes Q/S y Q^2/S^2 en (1.2) y usando (1.1) se observa que la condición (1.2) se puede reescribir (multiplicando convenientemente por S y por g en forma similar a lo efectuado para (3.1) y (3.2))

$$\frac{\partial Q}{\partial t} + \frac{2Q\partial Q}{S\partial x} + [gS - Q^2B/S^2]\frac{\partial Z}{\partial x} - Q^2S_x/S^2 + gSQ|Q|/D^2 = 0 \qquad (1.2'')$$

donde S_x es la derivada parcial de $S(Z(x,t),x)$ respecto de x para Z fijo.

El sistema (1) con condiciones iniciales (2) se puede escribir en la forma vectorial

$$\frac{\partial w}{\partial t} + A\frac{\partial w}{\partial x} = c \qquad (1'')$$

con condiciones iniciales

$$w(x,t_o) = w_o(x) \qquad (2'')$$

donde

$$w(x,t) = \begin{bmatrix} Z(x,t) \\ Q(x,t) \end{bmatrix}$$

$$A(w) = \begin{bmatrix} 0 & 1/B \\ gS - Q^2B/S^2 & 2Q/S \end{bmatrix}$$

$$c(w)= \begin{bmatrix} 0 \\ Q^2 S_x/S^2 \quad - gSQ|Q|/D^2 \end{bmatrix}$$

Se ve como antes que el sistema (1) con condiciones iniciales (2) tiene autovalores $\lambda_{1,2}= Q/S \pm \sqrt{gS/B}$ también reales y distintos, para $B > 0$, y por consiguiente interesa analizar las condiciones bajo las cuales existe una solución del sistema (1), (2) y del sistema (3), (4), y en qué sentido existe esta solución, tanto en el caso en que se tenga un problema de Cauchy exclusivamente a valores iniciales como en el caso en que se tenga un problema mixto con condiciones iniciales y de contorno (que es el caso de mayor importancia práctica, pues los tramos de río analizados normalmente tienen un extremo "aguas arriba" y otro "aguas abajo"; incluso si se analiza un río en toda su extensión, no tiene longitud infinita).

Existen diversas formas de plantear las ecuaciones (1), (2) en forma conservativa (no homogénea). Podemos escribir, por ejemplo, tomando como variables al area $S(Z(x,t),x)$, y a la velocidad $V(x,t)=Q/S$ (y suponiendo una relación unívoca entre S y Z en cada punto x)

$$\frac{\partial S}{\partial t} + \frac{\partial Q(V,S)}{\partial x} = 0 \qquad (7.1)$$

$$\frac{\partial V}{\partial t} + \frac{\partial}{\partial x}[\, V^2/2 + gZ(S)] + g\, V|V|S^2/D^2 = 0 \qquad (7.2)$$

y queda planteado un sistema diferencial conservativo no homogéneo de tipo

$$\frac{\partial w}{\partial t} + \frac{\partial}{\partial x} F(w,x) = c(w,x) \qquad (8)$$

con condición inicial $w(x,0)=w_o(x)$, donde (8) indica, en este caso particular, un par de ecuaciones:

$$w = \begin{bmatrix} S \\ V \end{bmatrix} \qquad F = \begin{bmatrix} Q(S,V) \\ V^2/2 + gZ(S) \end{bmatrix} \qquad c = \begin{bmatrix} 0 \\ -gV|V|S^2/D^2 \end{bmatrix}$$

Debido a la no linealidad de las ecuaciones de aguas poco profundas, para investigar la existencia de solución teórica *global* de las mismas se requiere la introducción del concepto de solución débil. Es decir, no se puede asegurar, por más regulares que sean las condiciones iniciales, que existe solución suficientemente derivable El contraejemplo clásico a la suposición de existencia de solución con derivada continua se obtiene sin necesidad de analizar sistemas diferenciales: basta estudiar (como puede verse en Lax [1972] o Lax [1973]) la ecuación diferencial (escalar)

$$\frac{\partial f}{\partial t} + a(f)\,\frac{\partial f}{\partial x} = 0$$

con condición inicial $f(x,0)=f_o(x)$. $a(f)$ es una función C^1 genuinamente no lineal, es decir, $da/df \neq 0$ siempre. Por ejemplo, $da/df > 0$ en lo que sigue.

Sobre cada curva

$$\frac{dx}{dt} = a(f(x,t)) \qquad (9)$$

se cumplirá la ecuación

$$\frac{\partial f}{\partial t} + \frac{dx}{dt}\,\frac{\partial f}{\partial x} = 0$$

o sea, sobre cada curva (9) la derivada total D/Dt de f con respecto a t, o sea $Df(x(t),t)/Dt$, será nula. Es decir, f será constante sobre cada curva, o sea a(f) será constante, y por consiguiente las curvas serán rectas, y el valor de f en un punto (x,t) será el mismo que en el punto en que la recta que pasa por (x,t) cruce la línea t=0 (que es conocido), o sea

$$f(x,t)=f_o(x - ta(f))$$

y todo estaría bien si hubiera *un solo* punto x-ta(f) sobre la recta (x,0) que pasara por x. Lamentablemente (o no, porque esto es lo que hace a la teoría de ecuaciones hiperbólicas no lineales tan fascinante), si f_o es estrictamente decreciente dos rectas de pendientes

$$\frac{dx}{dt} = f_o(x_1) \qquad y \qquad \frac{dx}{dt} = f_o(x_2)$$

con $x_1 < x_2$, terminan cortándose, o sea en el punto de intersección (x,t) deberá ser f(x,t) igual por un lado a $f_o(x_1)$ y por otro lado igual a $f_o(x_2)$, lo cual indica que f es cuanto menos discontinua en (x,t).

Este es un ejemplo clásico de que una propiedad de la condición inicial - en este caso, que f_o sea estrictamente decreciente - puede provocar problemas. Más específicamente, para sistemas de leyes de conservación de 2 x 2 el mismo Lax [1964] probó que las soluciones se hacen discontinuas después de un tiempo finito, a menos que las condiciones iniciales satisfagan una determinada condición de monotonicidad. Por eso se introduce la noción de *solución débil* del sistema de conservación (8).

Una solución débil del sistema (8) -para el caso general no homogéneo se puede ver por ejemplo Liu [1979]- estará dada por una función w acotada y medible tal que cumple la ley de conservación integral

$$\iint_{t\geq o} (w\partial\varphi/\partial t + F(w)\partial\varphi/\partial x - c(w,x)\varphi)\, dxdt + \int_{-\infty}^{\infty} w_o(x)\varphi\, dx = 0$$

para toda función regular $\varphi(x,t)$ con soporte compacto en $t\geq 0$. w_o deberá también ser medible y acotada.

Los métodos numéricos a usarse, por consiguiente, deberán considerar si detectan o no y representan o no las correspondientes discontinuidades.

Si no se dan condiciones de contorno, el problema que se tiene (de Cauchy) es tratar de resolver las ecuaciones en una región $\Omega=\{(x,t) / t \geq 0\}$. El sistema que se ha estudiado usualmente como

problema de Cauchy es el (5), (6) homogéneo, es decir, suponiendo que c=0 en (5). Se ha estudiado de dos maneras: por un lado, como caso particular al cual se le aplica la teoría que se va desarrollando para leyes de conservación, es decir, para sistemas (5), (6) generales en que $w(x,t)=(w_1(x,t),\ldots,w_n(x,t)$ y en que $F=(f_1(w_1,\ldots,w_n),\ldots,f_n(w_1,\ldots,w_n))$ es suficientemente regular; y por otro lado, mediante el análisis específico de las ecuaciones (3) o similares.

La teoría general de existencia se desarrolló en la década del 50 para ecuaciones escalares de conservación, es decir, con n=1, fundamentalmente a partir de varios trabajos de Oleinik en la Unión Soviética y Lax en Estados Unidos; mencionaremos Oleinik [1963], que contiene la traducción al inglés de una detalladísima y muy importante conferencia pronunciada por la autora el 27 de junio de 1956 en la tercera conferencia de matemática de la Unión Soviética, y Lax [1957]; la unión de las bibliografías de ambos trabajos ofrece una visión muy completa del desarrollo original de la teoría de las ecuaciones escalares hiperbólicas no lineales y de algunos métodos numéricos usados en la época. Por supuesto, esta teoría no se puede aplicar a nuestras ecuaciones pues en ellas n=2.

En esencia, se prueba que existe solución débil para ecuaciones de conservación de la forma

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \qquad x \in \mathbb{R} \qquad t > 0$$

con condición inicial $u(x,0) = u_o(x)$ si $u_o \in L^\infty(\mathbb{R})$ y $f'' > 0$ en el rango de u_o; para garantizar unicidad se requiere adicionalmente una condición llamada *de entropía* por razones físicas, que también se generalizará a sistemas (ver por ejemplo Lax [1973]). La demostración (adaptada de la de Oleinik, a quien se debe el teorema) de esta afirmación, y el estudio del comportamiento asintótico de la solución para t tendiendo a infinito puede verse en el clarísimo capítulo 16 del libro de Smoller [1983]; en las notas del mismo capítulo se indican algunas generalizaciones y la correspondiente bibliografía.

La teoría general de existencia de solución de un sistema de leyes de conservación, para n > 1, está basada en el trabajo de Glimm [1965], que establece la existencia de solución para sistemas estrictamente hiperbólicos y genuinamente no lineales (pero no la unicidad) si la variación total de u_o es suficientemente pequeña. El capítulo 19 de Smoller [1983] contiene una demostración del teorema de Glimm tomado casi íntegramente del trabajo original de éste, así como, en las notas, una indicación de avances en las generalizaciones (por ejemplo, qué pasa con datos iniciales no tan restringidos, comportamientos asintóticos, etc.). Una síntesis muy clara y concreta de los problemas en ecuaciones en derivadas parciales hiperbólicas casilineales con indicaciones sobre los rasgos más importantes de la teoría y de las líneas de investigación en curso puede verse en Liu [1983].

Pero la teoría general no soluciona todos los interrogantes del problema de Cauchy para las ecuaciones de aguas poco profundas. En primer lugar, para poder aplicar la teoría general hay que estar seguro (entre otras cosas) de que el sistema (3) es estrictamente hiperbólico, o sea los autovalores son reales y distintos. Pero los autovalores son $u + \sqrt{gh}$ y $u - \sqrt{gh}$, y hay que tomarse el trabajo de verificar que $h > 0$. (nada impide que, fisicamente, $h=0$, o sea el cauce "se seque" si se dan las condiciones para ello). Además, uno quiere ver si para estas ecuaciones, trabajando directamente sobre ellas, se pueden obtener resultados más generales que en la teoría general. Para ello se aprovecha la equivalencia entre las ecuaciones de aguas poco profundas y las ecuaciones isentrópicas de dinámica de gases politrópicos[1], y se adaptan los resultados que se obtienen para éstas últimas, sobre las que se ha trabajado más (probablemente por el impulso inicial en la década del cuarenta debido a motivos bélicos). En particular, DiPerna [1973], demuestra, como aplicación de los resultados que obtiene para cierta clase de sistemas de 2 x 2, la existencia de solución del problema de Cauchy para las mencionadas ecuaciones isentrópicas de gases politrópicos, para condiciones que, traducidas a las ecuaciones de aguas poco profundas, consisten en que v_o y h_o tengan variación total acotada y que

$$\inf v_o(x) + 2\sqrt{gh_o(x)} > \sup v_o(x) - 2\sqrt{gh_o(x)} \qquad x \in \mathbb{R}$$

Si se quiere estudiar un cauce fluvial, parte del cual está seco, se puede utilizar esta relación entre ecuaciones de la dinámica de fluidos incompresibles y ecuaciones simplificadas de aguas poco profundas, investigando según la línea de Liu y Smoller [1980].

Ahora bien, en modelización fluvial concreta se requerirá dar condiciones de contorno "aguas arriba" y "aguas abajo" (los tramos de ríos, por más extendidos que sean tienen comienzo y fin, como ya comentamos), y por consiguiente la franja que interesa será un rectángulo $\Omega=\{(x,t)/x_o\leq x\leq x_L,\ 0\leq t\leq T\}$, y sobre alguna de las rectas $x=x_o$, o $x=x_L$, o sobre ambas (dependiendo del signo de los autovalores) se dan condiciones de contorno del tipo

$$F_1(Q(x_i,t),Z(x_i,t))=f_1(t) \qquad (10.1)$$

$$F_2(Q(x_i,t),Z(x_i,t))=f_2(t) \qquad (10.2)$$

donde x_i es x_o o x_L; lo que se tiene en este caso es un problema con condiciones iniciales y de contorno (mixto). T es finito pues la solución interesa durante un intervalo temporal determinado. La teoría respectiva tiene un desarrollo menor que la teoría relativa al problema de Cauchy exclusivamente.

Para un sistema diferencial hiperbólico lineal, bajo ciertas condiciones bastante generales se puede asegurar la existencia de una solución del problema de Cauchy, continuamente dependiente de la solución inicial, y la existencia de una solución del problema mixto dando tantas condiciones de contorno en la frontera $x=x_o$ como autovalores positivos tenga la matriz del sistema, y tantas condiciones de contorno en la frontera $x=x_L$ como autovalores negativos tenga dicha matriz (en Godunov [1978], capítulo 1, secciones 13 a 17, puede verse una descripción muy completa y rigurosa del problema mixto en el caso lineal).

En nuestro caso no lineal, supondremos la existencia de una solución generalizada, con discontinuidades, valiendo también la condición sobre el signo de los autovalores (ver también Rozdestvenskiĭ y Janenko [1983], capítulo 1, sección 11, o el detalladísimo y clásico análisis "a pulmón" de Courant y Hilbert [1962], capítulo 5, parágrafo 2). Cabe acotar que deben darse condiciones de contorno en forma cuidadosa. Por ejemplo, si las condiciones de contorno son caudal aguas arriba $Q(x_o,t)$ constante y caudal aguas abajo $Q(x_L,t)$ constante tal que $Q(x_o,t) << Q(x_L,t)$ es evidente que la solución no puede existir durante un lapso prolongado pues el cauce se "vacía", y no puede seguir saliendo más caudal del que entra.

Por otra parte, el hecho de que la solución tiene discontinuidades no es en general mencionado (salvo para analizar saltos hidráulicos - resaltos, en la terminología de ingeniería hidráulica- o roturas bruscas de diques), pues normalmente los métodos numéricos más populares ignoran las discontinuidades y éstas no son relevantes para las aproximaciones usualmente buscadas en modelización fluvial (con las importantes excepciones, ya indicadas, de resalto y rotura brusca de diques).

La condición sobre el signo de los autovalores implica que si

$$Q/S > \sqrt{gS/B}$$

los dos autovalores son positivos, y es necesario dar dos condiciones de contorno "aguas arriba" en $x=x_o$ (éste es el caso de régimen supercrítico; si

$$Q/S < \sqrt{gS/B}$$

será necesario dar una condición de contorno "aguas arriba" y otra "aguas abajo" en $x=x_L$ (régimen subcrítico). Los regímenes subcrítico y supercrítico equivalen, según la analogía ya mencionada, a los regímenes subsónico y supersónico en dinámica de gases.

Si bien las condiciones de contorno pueden ser dadas por funciones F_i bastante generales, a los efectos prácticos es usual

que $F_i(Q,Z)=Q$ o $F_i(Q,Z)= Z$ (merece atención también el caso de $F_i(Q,Z)=0$, indicando una relación funcional entre caudal y nivel). En lo que sigue de este trabajo se analizará exclusivamente el caso subcrítico, que es el que se presenta en ríos de llanura o, en general, en ríos de pendiente poco pronunciada.

Sea ahora una grilla en la región Ω en que se quiere resolver numéricamente por diferencias finitas el problema (1), (2) , (10) de paso espacial Δx y de paso temporal Δt, o sea la grilla incluye los puntos de discretización (x_i, t^n), con $x_i=i\Delta x$, $t^n=n\Delta t$ (consideramos la grilla igualmente espaciada, tanto respecto del eje temporal como del espacial, exclusivamente por razones de simplicidad de notación, puesto que los esquemas numéricos usuales permiten intervalos Δx y Δt variables). Una aproximación linealizada en diferencias finitas al sistema (1) o (3) con dos niveles temporales puede escribirse vectorialmente (véase por ejemplo Richtmyer y Morton [1967], capítulo 3) como

$$B_1^{n+1} w^{n+1} = B_0^n w^n + c^n$$

donde $B_1^{n+1}=B_1^{n+1}(\Delta x,\Delta t)$ y $B_0^n=B_0^n(\Delta x,\Delta t)$ indican operadores lineales en diferencias finitas que reemplazan las derivadas en los tiempos de cálculo t^{n+1} y t^n. w^{n+1} indica el vector de valores desconocidos, por ejemplo

$$w^{n+1} = \{ Z_0^{n+1}, Q_1^{n+1}, Z_1^{n+1}, \ldots\ldots, Q_{L-1}^{n+1}, Z_{L-1}^{n+1}, Q_L^{n+1} \}$$

(esto si las condiciones de contorno son $g_1(t)=Q(x_0,t)$ y $g_2(t)=Z(x_L,t)$). w^n indica el vector de valores conocidos en el tiempo anterior

$$w^n = \{ Z_0^n, Q_1^n, Z_1^n, \ldots., Q_{L-1}^n, Z_{L-1}^n, Q_L^n \}$$

y c^n es un vector conocido obtenido a partir de las condiciones de contorno y del término no homogéneo. La solución en un paso determinado n se obtiene recurrentemente a partir de las condiciones iniciales. Si $B^{n+1}(\Delta t)=I$ (matriz unidad) para todos los intervalos de tiempo n, se tiene un esquema numérico explícito; de lo contrario se tiene un esquema implícito, y en cada intervalo de tiempo es necesario resolver un sistema lineal

$$B_1^{n+1} w^{n+1} = d^n \qquad (11)$$

con d^n conocido.

Dado que se conoce w_o (condición inicial) se tendrá la solución, a partir de ese valor, en forma recurrente, para todo n, en forma directa, si el esquema es explícito, o resolviendo un sistema lineal, si el esquema es implícito, procediendo, para obtener el resultado en el paso de tiempo m, a resolver el problema para n=1,2,...,m.

Dado que usualmente los esquemas implícitos garantizan estabilidad numérica usando intervalos temporales Δt significativamente mayores que los esquemas explícitos (en relación con un mismo Δx), pues no están sujetos a la condición de Courant-Friedrich-Lewy (CFL), son preferidos a éstos pese a que requieren programas computacionales más complicados y más memoria de computador para almacenar -de alguna manera- la matriz B^{n+1}.

Cabe observar que este análisis es válido para resolver mediante métodos de diferencias finitas sistemas diferenciales lineales en derivadas parciales; para sistemas no lineales se puede proceder - siempre trabajando en diferencias finitas - por linealización, como haremos en este trabajo, o usando técnicas para hallar soluciones de ecuaciones no lineales, que son más complicadas. En la Sección 5 se muestra cómo se forman las ecuaciones (11) a través de un ejemplo, por medio del método (linealizado) de Preissmann; antes, en la Sección 4, se indica bibliografía general para solución numérica de problemas en dinámica de fluidos, incluyendo técnicas no lineales.

Usualmente, cuando se analizan tramos unidimensionales de ríos, los esquemas numéricos implícitos para resolver (11) conducen a una matriz banda, donde el ancho de la banda depende del número de puntos que se toman para la discretización numérica. Esto permite resolver el sistema (11) en cada paso de tiempo usando un algoritmo de resolución de sistemas para matrices banda, con el consiguiente ahorro en tiempo de computadora y memoria central.

Esto se debe a que para matrices banda tanto la memoria necesaria como el número de operaciones aumentan linealmente con el número de puntos de discretización sobre el eje x con una constante de proporcionalidad dependiente del ancho de la banda.

Así, se han podido modelizar matemáticamente en forma precisa y eficiente numerosos tramos de ríos y canales. En lo que sigue se planteará y resolverá la generalización al caso de modelización de redes fluviales más complejas, esencialmente cuencas fluviales y deltas.

3. CONFLUENCIAS DE DOS TRAMOS FLUVIALES

Para generalizar el modelo hidrodinámico descripto por las ecuaciones (1) con condiciones iniciales (2) y condiciones de contorno (10) a cuencas fluviales o deltas es necesario analizar las ecuaciones que reflejan la modelización de la unión de dos cauces fluviales o la bifurcación en dos cauces fluviales. Desde el punto de vista físico, esta es la única diferencia con tramos unidimensionales simples; desde el punto de vista numérico, cuencas fluviales y deltas tienen diferencias topológicas que hacen necesario analizarlas por separado. La unión de dos cauces fue modelizada por Stoker [1957], capítulo 11, sección 2, dividiendo dicha unión en tres secciones transversales: dos aguas arriba de la confluencia (una sobre cada tramo que afluye) y una aguas abajo (ver figura 3). Las ecuaciones de compatibilidad de Stoker son la ecuación de conservación de la masa

$$Q_i + Q_j = Q_k \qquad (12.1)$$

y las ecuaciones que relaciona los niveles en las secciones transversales

$$Z_i = Z_j = Z_k \qquad (12.2)$$

Los puntos i,j,k representan los extremos aguas abajo de los tramos que finalizan en la confluencia y el extremo aguas arriba del tramo resultante, respectivamente. El mismo análisis vale para una bifurcación, cambiando el signo de los caudales, como también se ve en la figura 3. El propio Stoker propuso, en la referencia mencionada, un esquema explícito de resolución del modelo, que aplicó a la desembocadura del río Ohio en el Mississipi. La condición de compatibilidad de Stoker (12.1) indica simplemente que el caudal que fluye por la confluencia de dos cauces o canales es la suma de los que llegan a la confluencia; la condición (12.2) es una aproximación a las condiciones más precisas

$$Z_k - Z_i = \frac{U_k + U_i}{2g} (U_i - U_k) \qquad (13.1)$$

$$Z_k - Z_j = \frac{U_k + U_j}{2g} (U_j - U_k) \qquad (13.2)$$

donde $U_k = Q_k / S_k$, $U_i = Q_i / S_i$, $U_j = Q_j / S_j$ indican las velocidades medias del flujo en las secciones transversales k,i,j, respectivamente. Estas condiciones se deducen directamente de la ecuación de conservación de la cantidad de movimiento (1.1), como puede verse en Li *et al.* [1983]. Usualmente los términos de la derecha en (13.1) y (13.2) son despreciables, y se obtiene así la condición de compatibilidad de Stoker (12.2).

Otro enfoque para tratar el problema de las confluencias es considerar el modelo *bidimensional*, es decir, el modelo que representa el flujo no estacionario de aguas poco profundas sobre superficie libre en dos dimensiones espaciales: se toma en cuenta no solamente la dirección de las velocidades (o caudales) según el eje longitudinal x del cauce fluvial sino también según el eje transversal y. En este caso se tendrá un sistema diferencial de 3 ecuaciones de tipo

$$\frac{\partial w}{\partial t} + A_1 \frac{\partial w}{\partial x} + A_2 \frac{\partial w}{\partial y} = B$$

con $w=w(x,y,t)=(U(x,y,t),V(x,y,t),h(x,y,t))^t$

$$A_1 = A_1(w) = \begin{bmatrix} U & 0 & g \\ 0 & U & 0 \\ h & 0 & U \end{bmatrix} \quad A_2=A_2(w)=\begin{bmatrix} V & 0 & 0 \\ 0 & V & g \\ 0 & h & V \end{bmatrix} \quad B=B(w,x)=g\begin{bmatrix} Px-Sfx \\ Py-Sfy \\ 0 \end{bmatrix}$$

donde U=U(x,u,t) es la velocidad en el sentido del eje x, V=V(x,y,t) es la velocidad en el sentido del eje y, Px y Py son las pendientes del fondo en las direcciones x e y, y Sfx y Sfy son las pendientes de fricción en las direcciones x e y, dadas por

$$Sfx = U|U|h^2/D^2 \qquad Sfy = V|V|h^2/D^2$$

y donde ahora D=D(x,y,t).

Usando este modelo, una confluencia se puede modelizar numéricamente con una discretización en diferencias finitas como la que se indica en la figura 4. Existen numerosos modelos numéricos bidimensionales en diferencias finitas, entre los cuales citaremos el de Leendertse [1967], donde también puede verse la deducción de las ecuaciones diferenciales. Estos modelos tienen muchas más dificultades numéricas que los modelos unidimensionales, las cuales pueden comprobarse en el informe de Leendertse (en el cual se indica la modelización bidimensional de la zona de Haringvliet en el estuario del Rin en Holanda y los efectos del tsunami en la bahía de Tokio), sin contar las dificultades teóricas, para las cuales puede consultarse Majda [1984].

4. METODOS NUMERICOS EN HIDRAULICA FLUVIAL UNIDIMENSIONAL

Se han aplicado numerosos métodos numéricos en hidráulica fluvial unidimensional para flujos en períodos prolongados. Los primeros esquemas numéricos modernos en hidráulica fluvial fueron dos propuestos por Stoker [1957], capítulo 11, sección 5, usando en uno de ellos la formulación de las ecuaciones simplificadas (3) a lo largo de las características, es decir

$$2\{(u+c)\partial c/\partial x + \partial c/\partial t\} + \{(u+c)\partial u/\partial x + \partial u/\partial t\} = g(S_c - S_f)$$
$$-2\{(u-c)\partial c/\partial x + \partial c/\partial t\} + \{(u-c)\partial u/\partial x + \partial u/\partial t\} = g(S_c - S_f)$$

con $gh=c^2$ y resolviéndolas explícitamente, y usando en el otro (que recomienda) las ecuaciones originales en una grilla de puntos alternados, también explícitamente. (Estos *no* fueron los primeros métodos numéricos en dinámica de fluidos general; los fluidos compresibles, dada su importancia bélica, ya habían sido objeto de análisis numéricos en Los Alamos desde antes de finalizar la segunda guerra mundial, como indica Richtmyer [1957] en el prefacio a la primera edición de su libro).

Stoker aplicó su segundo método modelizando, en forma simplificada, el río Ohio y su confluencia con el Mississippi. Este fue un trabajo pionero de incalculable valor, pues permitió comprender la magnitud de los problemas (muchos de ellos no matemáticos) en modelización fluvial, además de poner de manifiesto la extraordinaria versatilidad de los matemáticos de la escuela de Nueva York. Ahora bien, como estos métodos eran explícitos, Stoker debió mantener un Δt incómodamente pequeño (conviene recordar que es común simular el comportamiento de un río durante varios meses) para asegurar la estabilidad de su esquema, tal como lo requiere la condición CFL.

Los métodos implícitos de tipo "caja de cuatro puntos" (en los que cada ecuación discretizada relaciona variables sobre los puntos en que las rectas $x=x_i$ y $x= x_{i+1}$, intersecan a las rectas $t=t^n$ y $t=t^{n+1}$) comenzaron a usarse en la década del 60. Al respecto, cabe mencionar a Amein y Fang [1970], que usan en esencia el método de Preissmann (descripto en la próxima sección)

con coeficiente de ponderación en el tiempo $\theta=1/2$ y sin linealizar totalmente las ecuaciones, luego de lo cual emplean el método de Newton-Raphson; estos autores aplican su método a la modelización del río Neuse, en Carolina del Norte. Amein y Chu [1975] usan un esquema análogo (también sin linealizar) pero con $\theta=1$, y modelizan un canal artificial en el sistema del valle de Tennessee.

En cuanto a los métodos en general, es muy útil la extensa reseña de Liggett y Cunge [1975], que incluye también los métodos de la escuela soviética, y por último el método aplicado por los modelizadores chinos, detallado en Li *et al.* [1983], que es, en esencia, una modificación del método de Godunov [1959], que puede consultarse en inglés en Holt [1977]. Además, están los métodos generales en dinámica de fluidos, muchos de los cuales pueden verse en Roache [1976], en Holt [1977], este último con particular énfasis en la escuela soviética, y en Peyret y Taylor [1983] (acotemos que estos tres textos mencionan sólo marginalmente métodos especialmente aplicables a aguas poco profundas). Algunos autores centroeuropeos utilizan el "método implícito de las características" (IMOC), con el cual modelizaron el río Danubio en Baviera (ver Schmitz y Edenhofer [1980]).

En lo que se refiere específicamente a sistemas con afluentes o estructuras deltaicas, para los cuales hay que incluir las ecuaciones de compatibilidad de Stoker, podemos citar varios modelos. Quinn y Wylie [1972] modelizaron, en forma rudimentaria - con apenas 6 puntos de cálculo - el río Detroit entre su nacimiento en el lago Erie y su desembocadura en el lago Saint Clair. Es el caso simple de río con afluente (esquema Y simple) porque la isla Grosse separa las dos ramas del río al nacer. Usaron un esquema de Preissmann combinado con el método de Newton Raphson para tratar los términos no lineales . Fread [1973] usó un método que termina siendo el método de Preissmann con $\theta=1$, también combinado con un Newton-Raphson, para un esquema Y simple, y mediante aproximaciones sucesivas a las condiciones de confluencia: en cada paso Δt se obtienen resultados para el río principal (considerado como tramo único) con aporte lateral dado en la confluencia (primera aproximación); con los niveles

obtenidos en la confluencia, modeliza el afluente usando esos niveles como condición de contorno aguas abajo; vuelve a calcular resultados sobre el río principal usando como aporte en la confluencia el obtenido para el afluente, y así sucesivamente hasta obtener aproximaciones satisfactorias y pasar al Δt siguiente. Esto lo aplicó a un sistema teórico.

Gradowczyk y Jacovkis [1974] modelizaron una red fluvial compleja (el delta del río Paraná, con 65 puntos de discretización) resolviendo el sistema lineal resultante por el método de Gauss con pivote maximal por columnas, sin optimizar ni la memoria ni el número de operaciones. No tenemos noticias de otras modelizaciones de este grado de complejidad para esa fecha.

Wood, Harley y Perkins [1975] consideraron una matriz de coeficientes externos para los bordes y las confluencias y modelizaron el río James en Estados Unidos, el estuario de Puerto Cork, en Irlanda, y la cuenca del río Bayamón, en Puerto Rico. Joliffe [1984] propuso una adaptación del método generalizado de Newton Raphson, y usó redes teóricas de canales. Li *et al.* [1983] modelizaron un esquema en Y del río Yangtse (en China) por medio del método modificado de Godunov con resolución implícita de las ecuaciones de la confluencia.

Cabe citar finalmente el uso del método de elección aleatoria, basado en el teorema de Glimm [1965] de existencia de solución débil de un sistema diferencial hiperbólico casilineal si la condición inicial es de variación "poco" acotada, cuando las discontinuidades de las ondas son importantes: Marshall y Méndez [1981] lo aplicaron a la solución de un sistema simplificado en forma conservativa, y posteriormente Marshall y Menéndez [1981] lo extendieron al caso en que el sistema es no homogéneo al introducir los efectos de fricción. En ambos casos las ecuaciones son las simplificadas para canales prismáticos de ancho unitario. El método de Glimm es explícito y de primer orden (esto último no es grave en hidráulica fluvial aplicada pues los errores de medición de los datos hidrométricos hacen irreal cualquier aproximación excesivamente precisa); es muy útil para seguir

discontinuidades, pero es excesivamente lento, o sea no es muy recomendable para modelizaciones de regímenes subcríticos en los que las discontinuidades no interesan particularmente.

Pero en resumen, para modelizaciones fluviales de flujos durante períodos prolongados (es usual modelizar un año hidrológico, por lo cual el intervalo temporal no puede ser excesivamente pequeño, lo cual induce a desechar los esquemas explícitos), de ríos extensos (lo cual implica gran número de puntos de discretización en general no igualmente espaciados) y de secciones transversales de geometría muy variable, el método de Preissmann, originado en 1960 (ver Preissmann [1961], Cunge y Wegner [1964], y, con mucho detalle, la tesis de Cunge [1966]) parece ser el más utilizado y aceptado, pese a ser también de primer orden. Dado que es el método con que se ejemplificará la teoría expuesta en lo sucesivo, lo examinaremos con atención en la próxima sección.

En todos los casos mencionados parte del problema consiste en resolver un sistema lineal donde la matriz, a priori, es rala pero no banda. Por consiguiente, resulta muy deseable, por razones no solamente de tiempo y memoria de computadora sino también de programación, poder modificar el sistema lineal de tal manera que el sistema equivalente resultante tenga asociada una matriz banda lo más angosta posible.

5. EL METODO DE PREISSMANN

. En esencia, el método de Preissmann es un esquema de cuatro puntos (ver figura 5). Las aproximaciones numéricas de una función f(x,t) y de sus derivadas están dadas por

$$f(x,t) \approx \theta\,(f_{i+1}^{n+1}+f_i^{n+1})/2 + (1-\theta)(f_{i+1}^{n}+f_i^{n})/2 \qquad (14.1)$$

$$\frac{\partial f}{\partial t} \approx \frac{1}{2\Delta t^n}\,(f_{i+1}^{n+1} + f_i^{n+1} - f_{i+1}^{n} - f_i^{n}) \qquad (14.2)$$

$$\frac{\partial f}{\partial x} \approx \frac{\theta}{\Delta x_i}\,(f_{i+1}^{n+1} - f_i^{n+1}) + \frac{(1-\theta)}{\Delta x_i}(f_{i+1}^{n} - f_i^{n}) \qquad (14.3)$$

donde f_k^m indica $f(x_k,t^m)$, $\Delta x_i=|x_{i+1}-x_i|$, $\Delta t^n= t^{n+1}-t^n$. θ es un coeficiente de ponderación en el tiempo (el coeficiente de "implicitud"), tal que $0 \le \theta \le 1$.

El esquema se utiliza usualmente linealizando en los incrementos de las funciones, es decir, se reemplaza f^{n+1} por $f^n+\Delta f^n$, y la incógnita es Δf^n. Al ser de tipo "caja de cuatro puntos", para cada discretización en diferencias finitas se usan los vértices (x_i,t^n), (x_i,t^{n+1}), (x_{i+1},t^n) y (x_{i+1},t^{n+1}). Todo el análisis que se realizará a continuación vale también, sin modificaciones, para cualquier esquema de "dos puntos contiguos implícitos", es decir, cualquier esquema en que en la capa t^{n+1} sobre la que se desconocen los valores de f hay dos puntos x_i contiguos. Sobre la capa t^n no interesa cuántos puntos se tomen. Es importante recordar, por otra parte, que el esquema de Preissmann, como todo método en diferencias finitas que no trata especialmente las características, ignora las discontinuidades; en esencia, las suaviza (las "borronea", traduciendo literalmente "to smear").

Para un esquema de caja de cuatro puntos, si la discretización en diferencias finitas corresponde a un tramo unidimensional de puntos de discretización $x_0,x_1,\ldots,x_L$, se tendrá un conjunto de ecuaciones que en el intervalo entre los puntos de discretización x_i y x_{i+1} son

$$A_{1i}\Delta Q_i+A_{2i}\Delta Z_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1}=A_{5i} \qquad (15.i)$$

$$B_{1i}\Delta Q_i+B_{2i}\Delta Z_i+B_{3i}\Delta Q_{i+1}+B_{4i}\Delta Z_{i+1}=B_{5i} \qquad (16.i)$$

donde los A_{ji} y B_{ji}, $j=1,\ldots,5$ dependen de la discretización utilizada. Para i=0 se debe conocer ΔQ_o , ΔZ_o, o una relación funcional entre Q_o y Z_o que se linealiza en el incremento, y análogamente para Q_L y Z_L (para simplificar la notación se suprimieron los superíndices n de los incrementos ΔQ^n y ΔZ^n). En particular, los valores de las variables A_{ji} y B_{ji} con la discretización de Preissmann se deducen del siguiente modo (escribiremos Δx por Δx_i y Δt por Δt^n y suprimiremos además los superíndices n):

La ecuación (1.1) en cada intervalo de discretización (i,i+1) se aproxima como

$$\frac{S_{i+1}^{n+1}- S_{i+1}+S_i^{n+1}- S_i}{2\Delta t} + \theta\frac{(Q_{i+1}^{n+1} -Q_i^{n+1})}{\Delta x} + (1-\theta)\frac{(Q_{i+1}-Q_i)}{\Delta x} = 0 \qquad (17)$$

Como $S_i^{n+1} = S_i(Z_i^{n+1}) = S_i(Z_i + \Delta Z_i)$, desarrollando en serie de Taylor respecto del nivel Z_i se tiene

$$S_i^{n+1} = S_i + (\frac{dS}{dZ})_i \Delta Z_i + O((\Delta Z_i)^2)$$

y análogamente para S_{i+1}. Reemplazando en (17) y usando que $\frac{dS}{dZ}=B$ se puede escribir, despreciando términos de orden superior a 1

$$\frac{B_{i+1}\Delta Z_{i+1} + B_i\Delta Z_i}{2\Delta t} + \theta\frac{\Delta Q_{i+1} - \Delta Q_i}{\Delta x} + \frac{Q_{i+1} - Q_i}{\Delta x} = 0$$

que es igual a

$$-\Delta Q_i + \frac{\Delta x}{2\theta\Delta t} B_i\Delta Z_i + \Delta Q_{i+1} + \frac{\Delta x}{2\theta\Delta t} B_{i+1} \Delta Z_{i+1} = \frac{(Q_i -Q_{i+1})}{\theta}$$

con lo cual los coeficientes A_{ki} , k=1,...5, tendrán los valores

$$A_{1i} = -1 \qquad (18.1)$$

$$A_{2i} = \frac{\Delta x\ B_i}{2\theta\Delta t} \qquad (18.2)$$

$$A_{3i} = 1 \qquad (18.3)$$

$$A_{4i} = \frac{\Delta x\ B_{i+1}}{2\theta\Delta t} \qquad (18.4)$$

$$A_{5i} = \frac{Q_i - Q_{i+1}}{\theta} \qquad (18.5)$$

Análogamente, la ecuación (1.2) se aproxima, para cada intervalo de discretización (i,i+1), como

$$\frac{\left(\frac{Q}{S}\right)_{i+1}^{n+1} - \left(\frac{Q}{S}\right)_{i+1} + \left(\frac{Q}{S}\right)_i^{n+1} - \left(\frac{Q}{S}\right)_i}{2g\Delta t} +$$

$$\frac{1}{2g}\left[\frac{\theta\left[\left(\frac{Q^2}{S^2}\right)_{i+1}^{n+1} - \left(\frac{Q^2}{S^2}\right)_i^{n+1}\right]}{\Delta x} + \frac{(1-\theta)\left[\left(\frac{Q^2}{S^2}\right)_{i+1} - \left(\frac{Q^2}{S^2}\right)_i\right]}{\Delta x}\right] +$$

$$-\frac{\theta\,(Z_{i+1}^{n+1} - Z_i^{n+1})}{\Delta x} + \frac{(1-\theta)\,(Z_{i+1} - Z_i)}{\Delta x} +$$

$$\frac{1}{2}\left[\theta\left[\left(\frac{Q|Q|}{D^2}\right)_{i+1}^{n+1} + \left(\frac{Q|Q|}{S^2}\right)_i^{n+1}\right] + (1-\theta)\left[\frac{Q_{i+1}|Q_{i+1}|}{D_{i+1}^2} + \frac{Q_i|Q_i|}{D_i^2}\right]\right] = 0 \qquad (19)$$

y desarrollando en serie de Taylor, como antes, hasta los términos de primer orden (ahora en dos variables, Z y Q), y despreciando los términos de orden superior, (19) se puede escribir

$$\frac{\frac{d}{dQ}\left(\frac{Q}{S}\right)_{i+1}\Delta Q_{i+1} + \frac{d}{dZ}\left(\frac{Q}{S}\right)_{i+1}\Delta Z_{i+1}}{2g\Delta t} +$$

$$\frac{\frac{d}{dQ}\left(\frac{Q}{S}\right)_{i}\Delta Q_{i} + \frac{d}{dZ}\left(\frac{Q}{S}\right)_{i}\Delta Z_{i}}{2g\Delta t} +$$

$$\theta\left[\left(\frac{Q^2}{S^2}\right)_{i+1} + \frac{d}{dQ}\left(\frac{Q^2}{S^2}\right)_{i+1}\Delta Q_{i+1} + \frac{d}{dZ}\left(\frac{Q^2}{S^2}\right)_{i+1}\Delta Z_{i+1}\right.$$

$$\left. - \left(\frac{Q^2}{S^2}\right)_{i} - \frac{d}{dQ}\left(\frac{Q^2}{S^2}\right)_{i}\Delta Q_{i} - \frac{d}{dZ}\left(\frac{Q^2}{S^2}\right)_{i}\Delta Z_{i}\right] / (2g\Delta x) +$$

$$(1-\theta)\left[\left(\frac{Q^2}{S^2}\right)_{i+1} - \left(\frac{Q^2}{S^2}\right)_{i}\right] / (2g\Delta x) +$$

$$\theta\,(Z_{i+1}^{n+1} - Z_{i}^{n+1})/\Delta x + (1-\theta)\,(Z_{i+1} - Z_{i})/\Delta x +$$

$$\frac{\theta}{2}\left[\left(\frac{Q|Q|}{D^2}\right)_{i+1} + \frac{d}{dQ}\left(\frac{Q|Q|}{D^2}\right)_{i+1}\Delta Q_{i+1} + \frac{d}{dZ}\left(\frac{Q|Q|}{D^2}\right)_{i+1}\Delta Z_{i+1}\right.$$

$$\left. + \left(\frac{Q|Q|}{D^2}\right)_{i} + \frac{d}{dQ}\left(\frac{Q|Q|}{D^2}\right)_{i}\Delta Q_{i} + \frac{d}{dZ}\left(\frac{Q|Q|}{D^2}\right)_{i}\Delta Z_{i}\right]$$

$$+ \frac{(1-\theta)}{2}\left[\left(\frac{Q|Q|}{D^2}\right)_{i+1} + \left(\frac{Q|Q|}{D^2}\right)_{i}\right] = 0$$

es decir

$$\left[\frac{\Delta Q_{i+1}}{S_{i+1}} - \frac{Q_{i+1}}{S_{i+1}^2} B_{i+1} \Delta Z_{i+1} + \frac{\Delta Q_i}{S_i} - \frac{Q_i}{S_i^2} B_i \Delta Z_i \right] / (2g\Delta t) +$$

$$\theta \left[\left(\frac{Q^2}{S^2}\right)_{i+1} + \left(\frac{2Q}{S^2}\right)_{i+1} \Delta Q_{i+1} - \left(\frac{2Q^2}{S^3}\right)_{i+1} B_{i+1} \Delta Z_{i+1} \right.$$

$$\left. - \left(\frac{Q^2}{S^2}\right)_i - \left(\frac{2Q}{S^2}\right)_i \Delta Q_i + \left(\frac{2Q^2}{S^3}\right)_i B_i \Delta Z_i \right] / (2g\Delta x) +$$

$$(1-\theta) \left[\left(\frac{Q^2}{S^2}\right)_{i+1} - \left(\frac{Q^2}{S^2}\right)_i \right] / (2g\Delta x) +$$

$$\theta (\Delta Z_{i+1} - \Delta Z_i) / \Delta x + (Z_{i+1} - Z_i) / \Delta x +$$

$$\left[\left(\frac{Q|Q|}{D^2}\right)_{i+1} + \left(\frac{Q|Q|}{D^2}\right)_i + \theta \left[\left(\frac{2|Q|}{D^2}\right)_{i+1} \Delta Q_{i+1} - \right. \right.$$

$$\left. \left. - \left(\frac{2Q|Q|}{D^3}\right)_{i+1} \frac{dD}{dZ}_{i+1} \Delta Z_{i+1} + \left(\frac{2|Q|}{D^2}\right)_i \Delta Q_i - \left(\frac{2Q|Q|}{D^3}\right)_i \left(\frac{dD}{dZ}\right)_i \Delta Z_i \right] \right] / 2 = 0$$

o sea

$$\left[\frac{\Delta x}{2g\Delta t S_i} - \frac{\theta Q_i}{gS_i^2} + \frac{\theta|Q_i|\Delta x}{D_i^2}\right]\Delta Q_i \quad +$$

$$\left[-\frac{Q_i B_i \Delta x}{2g\Delta t S_i^2} + \frac{\theta B_i Q_i^2}{gS_i^3} - \theta - \frac{\Delta x\, \theta\, Q_i|Q_i|}{D_i^3}\left(\frac{dD}{dZ}\right)_i\right]\Delta Z_i +$$

$$\left[\frac{\Delta x}{2g\Delta t S_{i+1}} + \frac{\theta Q_{i+1}}{gS_{i+1}^2} + \frac{\theta|Q_{i+1}|\Delta x}{D_{i+1}^2}\right]\Delta Q_{i+1} +$$

$$\left[-\frac{Q_{i+1}B_{i+1}\Delta x}{2g\Delta t S_{i+1}^2} - \frac{\theta B_{i+1}Q_{i+1}^2}{gS_{i+1}^3} + \theta - \frac{\Delta x\, \theta Q_{i+1}|Q_{i+1}|}{D_{i+1}^3}\left(\frac{dD}{dZ}\right)_{i+1}\right]\Delta Z_{i+1}$$

$$= \left[\frac{Q_i^2}{S_i^2} - \frac{Q_{i+1}^2}{S_{i+1}^2}\right]/(2g) + Z_i - Z_{i+1} - \frac{\Delta x}{2}\left[\frac{Q_i|Q_i|}{D_i^2} + \frac{Q_{i+1}|Q_{i+1}|}{D_{i+1}^2}\right]$$

y los coeficientes B_{ki} , k=1,...,5, tendrán la forma

$$B_{1i} = \frac{\Delta x}{2g\Delta t S_i} - \frac{\theta Q_i}{gS_i^2} + \frac{\theta|Q_i|\Delta x}{D_i^2} \qquad (20.1)$$

$$B_{2i} = -\frac{Q_i B_i \Delta x}{2g\Delta t S_i^2} + \frac{\theta B_i Q_i^2}{gS_i^3} - \theta - \frac{\Delta x\, \theta\, Q_i|Q_i|}{D_i^3}\left(\frac{dD}{dZ}\right)_i \qquad (20.2)$$

$$B_{3i} = \frac{\Delta x}{2g\ \Delta t\ S_{i+1}} + \frac{\theta Q_{i+1}}{gS_{i+1}^2} + \frac{\theta\ |Q_{i+1}|\ \Delta x}{D_{i+1}^2} \qquad (20.3)$$

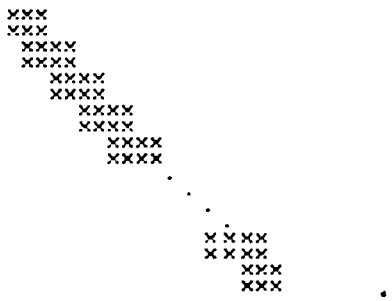
$$B_{4i} = - \frac{Q_{i+1} B_{i+1} \Delta x}{2g \Delta t S_{i+1}^2} - \frac{\Theta B_{i+1} Q_{i+1}^2}{g S_{i+1}^3} + \Theta -$$

$$\frac{\Delta x \Theta Q_{i+1} |Q_{i+1}|}{D_{i+1}^3} \left(\frac{dD}{dZ}\right)_{i+1} \qquad (20.4)$$

$$B_{5i} = \left[\frac{Q_i^2}{S_i^2} - \frac{Q_{i+1}^2}{S_{i+1}^2} \right] /(2g) + Z_i - Z_{i+1}$$

$$- \frac{\Delta x}{2} \left[\frac{Q_i |Q_i|}{D_i^2} + \frac{Q_{i+1} |Q_{i+1}|}{D_{i+1}^2} \right] \qquad (20.5)$$

O sea, la estructura general de la matriz del sistema originalmente planteado es



El primer par (1,2) de filas y el último (2L-1, 2L) tienen tres elementos no nulos en vez de cuatro pues una de las incógnitas es condición de contorno conocida.

En las secciones 7 y 8 volveremos a estudiar más atentamente esta matriz, su tridiagonalización, y su condicionamiento numérico. Veremos ahora la consistencia, convergencia y estabilidad del método de Preissmann, según la teoría de Lax y Richtmyer.

6. ANALISIS DE ESTABILIDAD NUMERICA DEL METODO DE PREISSMANN

La demostración teórica de las propiedades de estabilidad del método de Preissmann para las ecuaciones de Saint-Venant ha tropezado siempre con dificultades muy grandes, debido al carácter no lineal de las mismas. La experiencia de innumerables modelizaciones concretas en muchos ríos y canales del mundo (tramos simples) ha sido ampliamente satisfactoria (en particular este autor participó en la modelización de tramos simples del río Paraná, del río Uruguay, de otros ríos argentinos, y del río Amazonas, sin inconvenientes). La base teórica, extendida empíricamente a las ecuaciones casilineales, consiste en aplicar el método de Preissmann a un sistema de ecuaciones hiperbólicas lineales y homogéneas con coeficientes constantes, el sistema clásico de aguas poco profundas linealizado en que se supone $gH>0$

$$\frac{\partial h}{\partial t} + U\frac{\partial h}{\partial x} + H\frac{\partial u}{\partial x} = 0 \qquad (21.1)$$

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} + U\frac{\partial u}{\partial x} = 0 \qquad (21.2)$$

(con U y H constantes) y extrapolar al caso no lineal. Así se hace - con un sistema de ecuaciones diferenciales más sencillo, en que $U=0$, $g=H=1$ - en el análisis más completo realizado, el de la tesis de Cunge [1966][2], y en la mayor parte de la bibliografía existente. Vamos entonces a analizar las propiedades del esquema de Preissmann para sistemas lineales, pero no es necesario restringirse al sistema estudiado por Cunge: se pueden hacer las demostraciones para el sistema hiperbólico lineal general

$$\frac{\partial w}{\partial t} + A\frac{\partial w}{\partial x} = 0 \qquad (22)$$

siendo A una matriz constante de orden n con autovalores reales y distintos, y $w=w(x,t)$ una función vectorial de m componentes, con condiciones iniciales

$$w(x,0)=w_o(x) \qquad (23)$$

donde w_o es de cuadrado integrable en $-\infty < x < \infty$ o en un intervalo finito $x_o \leq x \leq x_L$. (Se tomará entonces como norma $\| \ \|$ la norma $L^2(\mathbb{R})$ o $L^2[0,2\Pi]$). Como veremos en seguida, usaremos en ambos casos subconjuntos densos de los respectivos espacios para nuestras demostraciones; dichos subconjuntos densos serán $C_o^2(\mathbb{R})$, el conjunto de las funciones dos veces continuamente derivables con soporte compacto, si el espacio es $L^2(\mathbb{R})$, y el subconjunto de las funciones f dos veces diferenciales tales que $f(0)=f(2\Pi)$, si el espacio es $L^2[0,2\Pi]$. En este caso, cuando convenga, extenderemos por periodicidad la definición de una función en $[0,2\Pi]$ a todo $\mathbb{R}$, aunque, por supuesto, los límites de integración se mantendrán en el intervalo $[0,2\Pi]$.

Observemos primero que existe una matriz de diagonalización S tal que $S^{-1}AS = \Lambda$, con Λ matriz diagonal de autovalores de A (A se puede diagonalizar pues sus autovalores son distintos). Por consiguiente, si $w = Sv$,

$$\frac{\partial Sv}{\partial t} + A \frac{\partial Sv}{\partial x} = 0 \qquad Sv_o(x) = Sv(x,0)$$

y usando que S es constante y puede salir fuera de la derivación, y multiplicando todo por S^{-1}, se obtiene

$$\frac{\partial v}{\partial t} + \Lambda \frac{\partial v}{\partial x} = 0 \qquad (24)$$

$$v_o(x) = v(x,0) \qquad (25)$$

Tenemos entonces que (22) , (23) equivale a (24) , (25). Esto es muy útil, pues el sistema (24) , (25) esta *desacoplado* , es decir, se puede descomponer en n ecuaciones

$$\frac{\partial v_i}{\partial t} + \lambda_i \frac{\partial v_i}{\partial x} = 0 \qquad 1 \leq i \leq n$$

con $v_i(x,0) = v_{o,i}(x)$.

La fabricación de la solución de (24), (25) para condiciones iniciales dos veces derivables con soporte compacto (en $\mathbb{R}$) o dos veces derivables de período 2Π (en $[0,2\Pi]$), es muy fácil, usando que sobre la recta $dx/dt = \lambda_i$ la derivada total D/Dt $(v_i(x(t),t)$ es nula: entonces sobre esa recta v_i es constante, o sea

$$v_i(x,t) = v_i(x-\lambda_i t,0) = v_{o,i}(x-\lambda_i t)$$

Usando la terminología de Godunov [1978] diremos que el sistema (24) , (25) está escrito en forma canónica. Además, la matriz Λ, al ser diagonal, es simétrica.

Formularemos el problema (22), (23) adecuando a él la teoría de estabilidad de Lax-Richtmyer, formulada por Lax en su seminario de la Universidad de Nueva York (enero de 1954), y publicada por Lax y Richtmyer [1956], y luego reproducida en Richtmyer [1957], Richtmyer y Morton [1967], Meis y Marcowitz [1981], etc.

Definiremos como *solución genuina* del problema (22), (23) a una familia a un parámetro $w(t)$ de elementos de $\mathfrak{B}$ ($\mathfrak{B}$ es el espacio de Banach en que está definida la condición inicial w_o) tal que

$$\|(w(t+\Delta t)-w(t))/\Delta t + A\partial w/\partial x(t)\| \to 0 \text{ si } \Delta t\to 0 \text{ con } 0\le t\le T$$

Sea ahora el subconjunto de los w_o para los cuales existe una solución genuina de (22), (23). Definimos como $E_o(t)$ al operador lineal, definido en ese subconjunto, que a cada condición inicial hace corresponder $w(t)$. En nuestro caso, $\mathfrak{B}$ será $L^2(\mathbb{R})$ o $L^2[0,2\Pi]$ y la norma $\|\ \|$ la norma en ese espacio.

El problema (22), (23) está bien planteado, es decir, como se indica en el capítulo 4, parágrafo 3, de Richtmyer y Morton [1967], el dominio de E_o es denso en $\mathfrak{B}$ y la familia de operadores E_o está uniformemente acotada, o sea existe $K=K(T)$ tal que $\|E_o\|<K$ para $0\le t\le T$ (T fijado arbitrario).

La demostración de que el problema (22), (23) está bien planteado (en $L^2(\mathbb{R})$ o $L^2[0,2\Pi]$) puede verse por ejemplo en Godunov [1978], capítulo 2, sección 10. En realidad Godunov demuestra algo más fuerte: demuestra que si se tiene el sistema

$$E\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} + C\frac{\partial u}{\partial y} + Qu = f(x,y,t) \qquad (26)$$

con condiciones iniciales

$$u(x,y,0) = u_o(x,y) \qquad (27)$$

y con matrices $E=E(x,y,t)$, $A=A(x,y,t)$, $C=C(x,y,t)$, $Q=Q(x,y,t)$ de las cuales E, A y C son simétricas y E es además definida positiva, y con todas las matrices y los vectores u_o y f funciones regulares de las variables de las que dependen, entonces existe una solución regular $u(x,y,t)$ del sistema (26) tal que

$$\|u(t)\| \leq K_1(T)\,\|u_o\| + K_2(T) \max_{0\leq t\leq T} \|f(t)\|$$

donde ahora la norma se toma integrando en las dos variables x e y los cuadrados de las funciones correspondientes . El sistema (26) es t-simétrico en el sentido de Friedrichs. Naturalmente (26) , (27) engloba a nuestro problema (22), (23) - o, equivalentemente, a nuestro problema (24) , (25) - pues basta considerar E = I, C = 0 , Q = 0 , f = 0 y A con coeficientes constantes (o directamente Λ, que es simétrica).

Obsérvese que, por el clásico teorema de extensión de un operador lineal acotado, E_o tiene una extensión a un operador E(t) definido en todo $\mathfrak{B}$, que será la solución generalizada del problema (22), (23).

En primer lugar, recordaremos las definiciones de consistencia, convergencia y estabilidad de aproximaciones en diferencias finitas a sistemas diferenciales, según Richtmyer y Morton [1967], capítulo 3, sección 2 (teoría general) y capítulo 4 (problemas exclusivamente de valores iniciales y con coeficientes constantes), que simplificaremos para aplicarla al tipo de ecuaciones diferenciales (22), (23):

Sea $w^n=w^n(x)$ una aproximación a $w(x,n\Delta t)$ obtenida de las ecuaciones en diferencias

$$B_1 w^{n+1} = B_0 w^n \qquad (28)$$

(28) será una aproximación en diferencias finitas al sistema diferencial (22) si $B_1=B_1(\Delta t,\Delta x)$ y $B_2=B_2(\Delta t,\Delta x)$ son operadores lineales, dependientes de Δt y Δx pero no de t y x, tales que B_1 tiene inversa, es decir, w^n determina unívocamente a w^{n+1} y tales que, para cada x, B_1 y B_0 son combinaciones lineales de w^{n+1} en puntos cercanos $x+d_i\Delta x$, con d_i entero.

En la práctica, lo que se quiere es que w_i^n "esté cerca" (en algún sentido) de $w(i\Delta x,n\Delta t)$, y B_1 y B_0 consisten, para cada i, en combinaciones lineales de los w^{n+1} y w^n, respectivamente, en coordenadas j "cercanas" a i. Una fórmula del tipo de la dada en (26) se denomina "fórmula de dos niveles", pues solamente involucra a dos pasos de tiempo $t^n=n\Delta t$ y $t^{n+1}=(n+1)\Delta t$. Para hacer depender B_1 y B_0 de un solo parámetro que haremos tender a cero, se supondrán, en las definiciones que siguen, relaciones $\Delta x=g(\Delta t)$ tales que $\Delta x \to 0$ cuando $\Delta t \to 0$, y $\Delta x=O(\Delta t)$. Por consiguiente, se tendrá

$$w^{n+1} = C(\Delta t)w^n \qquad (29)$$

donde $C(\Delta t) = B_1^{-1}(\Delta t,g(\Delta t))B_0(\Delta t,g(\Delta t))$. (Por supuesto la igualdad (29) importa en el análisis teórico del método numérico; es altamente improbable que alguien encare la solución de (28) pasando por (29) en vez de resolver directamente el sistema (28)).

Entonces estamos en condiciones de definir consistencia, convergencia y estabilidad del esquema numérico dado por (28).

Consistencia: Se dice que una aproximación en diferencias finitas a un sistema diferencial de la forma (22) es consistente si para todas las soluciones genuinas en un subconjunto denso del espacio en que están definidas las condiciones iniciales (en nuestro caso, tomaremos como subconjunto el de las funciones dos veces continuamente derivables de soporte compacto).

$$\frac{\|w(t+\Delta t)-C(\Delta t)w(t)\|}{\Delta t} = \frac{\|w(t+\Delta t)-B_1^{-1}(\Delta t)B_0(\Delta t)w(t)\|}{\Delta t} = O(\Delta t^p) \qquad (30)$$

con $\rho > 0$. El mayor ρ para el cual se cumple (30) se denomina *orden de aproximación* del esquema numérico.

Convergencia: La familia de operadores en diferencias finitas $C(\Delta t)$ es una aproximación convergente al problema (22), (23) si, para cada t fijo, con $0 < t < T$, cada sucesión de incrementos temporales $\Delta_1 t$, $\Delta_2 t, \ldots$, tendientes a cero, y cada sucesión de enteros n_j cercanos a $t/\Delta_j t$ para cada j, en el sentido que $n_j \Delta_j t$ tiende a t cuando j tiende a infinito, vale

$$\| C(\Delta_j t)^{n_j} w_o - E(t) w_o \| \to 0$$

para toda w_o en el espacio (de Banach) en el que están definidas las condiciones iniciales.

Estabilidad: La aproximación $C(\Delta t)$ es estable si el conjunto infinito de operadores

$$C(\Delta t)^n \qquad \begin{matrix} 0 < \Delta t < \tau \\ 0 < n\Delta t \leq T \end{matrix}$$

está uniformemente acotado, para algún $\tau > 0$.

TEOREMA: El esquema de Preissmann es consistente para $\mathfrak{B}=L^2(I)$, donde $I = \mathbb{R}$ o $I = [0, 2\Pi]$.

DEMOSTRACION: Como B_o y B_1 se obtienen directamente de la expresión (22), B_1 deberá ser $O(\Delta t^{-1})$ y, por consiguiente, (30) equivale a

$$\frac{\|B_1 w(t+\Delta t) - B_o w(t)\|}{\Delta t} = O(\Delta t^{\rho-1})$$

o sea

$$\|B_1 w(t+\Delta t) - B_o w(t)\| = O(\Delta t^{\rho}) \qquad (31)$$

y es (31) lo que necesitamos probar, para $\rho > 0$.

Sea T el operador de traslación que consiste en reemplazar el valor de una función en un punto x por su valor en $x + \Delta x$. Entonces, usando que A tiene coeficientes constantes, al especializar (31) para la discretización de Preissmann obtenemos (siendo I el operador identidad)

$$B_1 = \frac{(T + I)}{2\Delta t} + \frac{\theta\, A\, (T - I)}{\Delta x}$$

$$B_o = \frac{(T + I)}{2\Delta t} - \frac{(1-\theta)\, A\, (T - I)}{\Delta x}$$

Por otra parte, desarrollando en serie de Taylor hasta la derivada segunda, y usando la identidad $S^{-1}AS = \Lambda$, para S matriz de diagonalización de A y Λ matriz diagonal de sus autovectores, se tiene, si $w = Sv$

$$\|B_1 w(t+\Delta t) - B_o w(t)\| =$$

$$\|(T+I)(w(t+\Delta t)-w(t))/(2\Delta t) + A(\theta(T-I)w(t+\Delta t)+(1-\theta)(T-I)w(t))/\Delta x\|$$

$$=\|(T+I)S((v(t+\Delta t)-v(t))/(2\Delta t) +$$

$$A(\theta(T-I)Sv(t+\Delta t)+(1-\theta)(T-I)Sv(t))/\Delta x\| \qquad (32)$$

Usando ahora que T conmuta con cualquier matriz con coeficientes constantes A, (32) es igual a

$$\|S((T+I)(v(t+\Delta t)-v(t))/(2\Delta t)$$

$$+ AS(\theta(T-I)v(t+\Delta t)+(1-\theta)(T-I)v(t))/\Delta x)\|$$

$$\leq \|S\|(\int \sum_i |(v_{o,i}(x+\Delta x-\lambda_i(t+\Delta t)) + v_{o,i}(x-\lambda_i(t+\Delta t)) -$$

$$v_{o,i}(x+\Delta x-\lambda_i t) - v_{o,i}(x-\lambda_i t))/(2\Delta t) +$$

$$\lambda_i(\theta(v_{o,i}(x+\Delta x-\lambda_i(t+\Delta t))-v_{o,i}(x-\lambda_i(t+\Delta t))) +$$

$$(1-\theta)(v_{o,i}(x+\Delta x-\lambda_i t)-v_{o,i}(x-\lambda_i t)))/\Delta x\ |^2\ dx\ )^{1/2}$$

$$\leq \|S\|(\int \sum_i |-\lambda_i(v'_{o,i}(x+\Delta x-\lambda_i t)+v'_{o,i}(x-\lambda_i t)$$

$$+v''_{o,i}(x+\Delta x-\lambda_i(t+\xi_1\Delta t))\lambda_i\Delta t/2 + v''_{o,i}(x-\lambda_i(t+\xi_2\Delta t))\lambda_i\Delta t/2))/2$$

$$+\lambda_i(\theta(v'_{o,i}(x-\lambda_i(t+\Delta t)) + v''_{o,i}(x+\xi_3\Delta x-\lambda_i(t+\Delta t))\Delta x/2)$$

$$+ (1-\theta)(v'_{o,i}(x-\lambda_i t) +v''_{o,i}(x+\xi_4\Delta x-\lambda_i\Delta t)\Delta x/2))|^2\ dx\ )^{1/2}$$

$$\leq \|S\|((\int \sum_i |-\lambda_i(v'_{o,i}(x+\Delta x-\lambda_i t) + v'_{o,i}(x-\lambda_i t))/2$$

$$+\lambda_i(\Theta\, v'_{o,i}(x-\lambda_i(t+\Delta t) + (1-\Theta)v'_{o,i}(x-\lambda_i t))|^2\, dx\,)^{1/2}$$

$$+ M\ \rho(A)\ (\rho(A)\Delta t + \Delta x)\ n\ N^{1/2}/2\ )$$

$$\leq \|S\|((\int \sum |-\lambda_i(v'_{o,i}(x-\lambda_i t)+v'_{o,i}(x-\lambda_i t)+v''_{o,i}(x+\xi_5\Delta x-\lambda_i t)\Delta x)/2$$

$$+\lambda_i(\Theta(v'_{o,i}(x-\lambda_i t) + v''_{o,i}(x-\lambda_i(t+\xi_6\Delta t)\lambda_i\Delta t)$$

$$+(1-\Theta)v'_{o,i}(x-\lambda_i t))\ |^2\, dx\,)^{1/2} + M\ \rho(A)\ (\rho(A)\Delta t + \Delta x)\ nN^{1/2}/2)$$

$$\leq \|S\|\ M\ \rho(A)\ (\rho(A)\Delta t+\Delta x)\ nN^{1/2}$$

$$= \|S\|\ (O(\Delta x) + O(\Delta t))$$

donde $\rho(A)$ indica el radio espectral de A, M es una cota superior de la norma L^∞ de las derivadas segundas de todas las $v_{o,i}$ (que existe pues las $v_{o,i}$ son funciones C^2 con soporte compacto, si estamos en $L^2(\mathbb{R})$, o con invervalo de definición que nos interesa dado por $[0,2\Pi]$, si estamos en $L^2[0,2\Pi]$), n es el orden de la matriz A, N es la medida de un compacto que contiene a los soportes de todas las $v_{o,i}$ y $0\leq\xi_j=\xi_j\leq 1$ para $1\leq i\leq 6$. Además, las ξ_j dependen de x, de Δt y de Δx.

Suponiendo entonces que la relación $\Delta x=g(\Delta t)$ sea $O(\Delta t)$, queda demostrado el teorema. En particular, se observa que la aproximación es de primer orden. ■

Si probáramos ahora la estabilidad del esquema de Preissmann, podríamos usar el teorema de Lax (Lax y Richtmyer [1956], o Richtmyer y Morton [1967], capítulo 3, sección 5, o Meis y Markowitz [1981], donde hay una demostración muy clara) que dice que dado un problema de valores iniciales bien planteado que satisface la condición de consistencia, estabilidad es condición necesaria y suficiente para convergencia. Y eso es lo que haremos. Restringiéndonos siempre al problema de Cauchy exclusivamente podemos enunciar el siguiente

TEOREMA: Sea A una matriz de orden n cuyos autovalores son reales y sea el sistema diferencial lineal con coeficientes constantes

$$\frac{\partial w}{\partial t} + A\frac{\partial w}{\partial x} = 0 \qquad (33)$$

con condiciones iniciales

$$w(x,t)=0 \qquad (34)$$

siendo $w(x,t)=(w_1(x,t),\ldots w_n(x,t))^t$. Entonces para que se cumpla la condición de von Neumann de estabilidad de soluciones numéricas de ecuaciones diferenciales de evolución para el método de Preissmann deberá ser $\theta\geq 1/2$. Además, en este caso, la condición de von Neumann es también suficiente, es decir, el esquema es estable (para A matriz de autovalores reales) si y sólo si $\theta\geq 1/2$.

DEMOSTRACION: Si $w(x,t) \in C^2[0,2\Pi]$ tal que $w(0,t) = w(2\Pi,t)$, sea

$$w(x,t)= \sum_m \omega(m,t)\, e^{imx}$$

el desarrollo en serie de Fourier de w en t, con $\omega(m,t)$ el m-simo coeficiente de Fourier de $w(x,t)$. Análogamente, si $w(x,t) \in C_0^2(\mathbb{R})$ sea

$$w(x,t)= \int \omega(m,t)e^{imx}\, dm$$

la representación de w por medio de su transformada de Fourier, siendo $\omega(m,t)$ la transformada de Fourier de w en el punto m. Denotemos por ω_m^n a $\omega(m,n\Delta t)$ en ambos casos, y $\mathfrak{F}$ será indistintamente la aplicación de $L^2[0,2\Pi]$ en $\ell^2(\mathbb{Z})$ o de $L^2(\mathbb{R})$ en $L^2(\mathbb{R})$ que transforma una función en su serie de Fourier o en su transformada de Fourier.

Pasando al espacio de las series o transformadas de Fourier, tal como se ve por ejemplo en Richtmyer y Morton [1967], capítulo 4, sección 4, o en Meis y Marcowitz [1981], capítulo 9, bastará estudiar las matrices de amplificación $B(\Delta x,\Delta t,m)$ tales que

$$\omega_m^{n+1}=B(\Delta x,\Delta t,m)\omega_m^n$$

Discretizando según el esquema de Preissmann, y usando que, si T es el operador ya definido tal que $Tf(x)=f(x+\Delta x)$, vale

$$\mathcal{F}T(f)(m) = e^{im\Delta x}\mathcal{F}(f)$$

se tiene

$$\omega_m^{n+1}\left[\frac{e^{im\Delta x}+1}{2\Delta t}\right] - \omega_m^n\left[\frac{e^{im\Delta x}+1}{2\Delta t}\right] +$$

$$A\left[\theta\omega_m^{n+1}\left[\frac{e^{im\Delta x}-1}{\Delta x}\right] +(1-\theta)\,\omega_m^n\left[\frac{e^{im\Delta x}-1}{\Delta x}\right]\right] = 0 \qquad (35)$$

Si llamamos r a $\frac{\Delta t}{\Delta x}$ queda, multiplicando (35) por $2\Delta t$

$$(1+e^{im\Delta x})I\omega_m^{n+1} + 2r\theta(e^{im\Delta x}-1)A\omega_m^{n+1} =$$

$$(1+e^{im\Delta x})I\omega_m^{n} - 2r(1-\theta)(e^{im\Delta x}-1)A\omega_m^{n}$$

Multiplicando por $e^{-im\Delta x/2}$ y agrupando se tiene

$$\left[(e^{im\Delta x/2}+e^{-im\Delta x/2})I + 2r\theta(e^{im\Delta x/2}-e^{-im\Delta x/2})A\right]\omega_m^{n+1} =$$

$$\left[(e^{im\Delta x/2}+e^{-im\Delta x/2})I - 2r(1-\theta)(e^{im\Delta x/2}-e^{-im\Delta x/2})A\right]\omega_m^{n}$$

o sea

$$(\cos(m\Delta x/2)I + 2r\theta\, i\, \mathrm{sen}(m\Delta x/2)A)\omega_m^{n+1} =$$

$$(\cos(m\Delta x/2)I - 2r(1-\theta)\, i\, \mathrm{sen}(m\Delta x/2)A)\omega_m^{n}$$

es decir

$$\omega_m^{n+1} = (\cos(m\Delta x/2)I + 2r\theta\, i\, \mathrm{sen}(m\Delta x/2)A)^{-1}$$
$$(\cos(m\Delta x/2)I - 2r(1-\theta)\, i\, \mathrm{sen}(m\Delta x/2)A)\omega_m^{n}$$

La matriz de amplificación $B(\Delta x,\Delta t,m)$ en el sentido de von Neumann será entonces

$$B(\Delta x,\Delta t,m)=(\cos(m\Delta x/2)I + 2r\theta\, i\, \mathrm{sen}(m\Delta x/2)A)^{-1}.$$
$$(\cos(m\Delta x/2)I - 2r(1-\theta)\, i\, \mathrm{sen}(m\Delta x/2)A)$$

Según el criterio de von Neumann, condición necesaria para que el método converja es que

$$\rho(B(\Delta x(\Delta t),\Delta t,m) \leq 1 + O(\Delta t) \qquad 0<\Delta t<\tau \quad \forall\, m$$

con $\rho(B)$ radio espectral de B. Como B es una función racional $R(A)$ de A, sus autovalores son los obtenidos evaluando R en los autovalores de A. Cada autovalor λ será entonces de la forma

$$(\cos(m\Delta x/2)+2r\theta i\mathrm{sen}(m\Delta x/2)\lambda)^{-1}.(\cos(m\Delta x/2)-2r(1-\theta)i\ \mathrm{sen}(m\Delta x/2)\lambda)$$

En particular, $\rho(B)\leq 1$ cuando para todo autovalor λ de A

$$\frac{\cos^2(m\Delta x/2) + 4r^2(1-\theta)^2\mathrm{sen}^2(m\Delta x/2)\lambda^2}{\cos^2(m\Delta x/2) + 4r^2\theta^2\mathrm{sen}^2(m\Delta x/2)\lambda^2} \leq 1$$

o sea cuando

$$4r^2\theta^2\mathrm{sen}^2(m\Delta x/2)\lambda^2 \geq 4r^2(1-\theta)^2\mathrm{sen}^2(m\Delta x/2)\lambda^2$$

lo cual equivale (simplificando) a $\theta \geq 1/2$.

Si A es normal, la condición de von Neumann es suficiente. No lo es, pero podemos garantizar la suficiencia del criterio de von Neumann si podemos probar, como se demuestra en Richtmyer y Morton [1967], capítulo 4, sección 11, que *si* $B(\Delta x(\Delta t),\Delta t,m)$ *tiene un conjunto de n autovectores linealmente independientes* $v_1,\ldots,v_n$ *y* V *es la matriz cuyas columnas son los autovectores normalizados, entonces existe una constante* $\delta > 0$ *tal que* $\Delta \geq \delta$, *siendo* Δ *el determinante de* V^tV. Y efectivamente podemos probar eso si A es no singular, pues al ser B una función racional de A, los autovectores de B coinciden con los de A, que no dependen ni de Δt ni de m, o sea V y V^{-1} son no singulares y se puede tomar directamente Δ como δ. En nuestro análisis linealizado, los autovectores (constantes) son

$$(H, -\sqrt{gH}) \; / \; \| (H, -\sqrt{gH}) \|$$

$$(H, \sqrt{gH}) \; / \; \| (H, \sqrt{gH}) \|$$

y, para que la matriz **A** sea no singular, basta suponer $H \neq 0$. ■

Este teorema garantiza en algún sentido el uso del esquema de Preissmann en ecuaciones no lineales, por generalización (no fundamentada, naturalmente, sino tan sólo supuesta). Sin embargo, esas generalizaciones deben ser testeadas, aunque sea empíricamente: en la sección 12 se muestra un ejemplo de que se produce inestabilidad con $\theta=2/3$, e incluso con $\theta=0,75$ los resultados no son para nada satisfactorios (las oscilaciones observadas quitan gran margen de precisión a los resultados). Eso con un sistema río-afluente (el esquema en Y). En el apéndice 2 se observa un caso de inestabilidad numérica para $\theta=2/3$ con un tramo simple. No hemos encontrado en la bibliografía citas de inestabilidades para $\theta > 1/2$.

Este autor usó este teorema para aplicar el método de Preissmann a la discretización de modelos unidimensionales con fondo móvil, en los cuales **A** es ahora una matriz de 3x3, con una ecuación adicional representando el balance de masa sólida del fondo móvil, según el modelo planteado por Gradowczyk [1968]. El modelo resultante, que fue aplicado a la modelización de los canales de restitución de dos represas proyectadas sobre el río Limay, puede verse en el informe de EGASAT [1982].

Por último, cabe mencionar que los resultados de esta sección son conocidos, pero no es fácil encontrar en la bibliografía su desarrollo completo y riguroso. Por ese motivo se han expuesto en forma detallada.

7. ALGORITMO DE TRIDIAGONALIZACION PARA TRAMOS SIMPLES

Pasemos ahora a analizar el algoritmo de tridiagonalización de la matriz **A** formada por los pares de ecuaciones lineales (15.i), (16.i), modificando naturalmente el término de la derecha **b** de la ecuación matricial **Ax=b**.

El algoritmo de tridiagonalización de los pares de ecuaciones (15.i), (16.i) es el siguiente:

a) Punto de discretización i genérico:

Se tiene(15.i), (16.i). En un intervalo de discretización (i,i+1), restando (15.i) multiplicado por B_{1i}/A_{1i} de (16.i) se tiene

$$D_{1i}\Delta Z_i+D_{2i}\Delta Q_{i+1}+D_{3i}\Delta Z_{i+1}=D_{4i} \qquad (36.i)$$

con $D_{ki} = B_{k+1,i} - A_{k+1,i}B_{1i}/A_{1i}$ $\qquad k=1,..,4.$

Restando ahora (36.i) multiplicada por A_{4i}/D_{3i} de (15.i) se tiene

$$C_{1i}\Delta Q_i+C_{2i}\Delta Z_i+C_{3i}\Delta Q_{i+1}=C_{4i} \qquad (37.i)$$

con $C_{1i}=A_{1i}$,

$C_{ki}=A_{ki}-D_{k-1,i}A_{4i}/D_{3i}$ $\qquad k=2,3$ $\qquad$ y

$C_{4i}=A_{5i}-D_{4i}A_{4i}/D_{3i}$

A partir de aquí basta analizar los extremos para exhibir la matriz tridiagonal:

b) Extremo aguas arriba

b1) Condición de contorno $Q(x_o,t)$ conocida.

En este caso las dos primeras ecuaciones de la matriz son:

$$C_{20}\Delta Z_0+C_{30}\Delta Q_1 = C_{40}-C_{10}\Delta Q_0 \qquad (37.0)$$

$$D_{10}\Delta Z_0+D_{20}\Delta Q_1+D_{30}\Delta Z_1 = D_{40} \qquad (36.0)$$

donde $\Delta Q_0 = Q_0^{n+1} - Q_0^n$ es conocido.

b2) Condición de contorno $Z(x_0,t)$ conocida:

En este caso las dos primeras ecuaciones resultan ser

$$C_{10}\Delta Q_0+C_{30}\Delta Q_1 = C_{40}-C_{20}\Delta Z_0 \qquad (37'.0)$$

$$D_{20}\Delta Q_1+D_{30}\Delta Z_1 = D_{40}-D_{10}\Delta Z_0 \qquad (36'.0)$$

c) Condición de contorno aguas abajo:

c1) Condición de contorno $Q(x_L,t)$ conocida:

En este caso las dos últimas ecuaciones pasan a ser

$$C_{1,L-1}\Delta Q_{L-1}+C_{2,L-1}\Delta Z_{L-1} = C_{4,L-1}-C_{3,L-1}\Delta Q_L \qquad (37.L-1)$$

$$D_{1,L-1}\Delta Z_{L-1}+D_{3,L-1}\Delta Z_L = D_{4,L-1}-D_{2,L-1}\Delta Q_L \qquad (36.L-1)$$

c2) Condición de contorno $Z(x_L,t)$ conocida.

En este caso las dos últimas ecuaciones del sistema se expresan como

$$C_{1,L-1}\Delta Q_{L-1}+C_{2,L-1}\Delta Z_{L-1}+C_{3,L-1}\Delta Q_L = C_{4,L-1} \qquad (37'.L-1)$$

$$D_{1,L-1}\Delta Z_{L-1}+D_{2,L-1}\Delta Q_L = D_{4,L-1}-D_{3,L-1}\Delta Z_L \qquad (36'.L-1)$$

y se ha obtenido una matriz banda tridiagonal que se puede resolver por el método clásico del "barrido doble" como puede verse en Gelfand y Lokutsievsky [1964], por ejemplo. Si bien no se puede probar en general que se cumplen las condiciones habituales

de dominancia diagonal que aseguran buen condicionamiento numérico para la solución del sistema tridiagonal (36), (37) (ver por ejemplo el capítulo 4, sección 2 de Godunov y Riabenkii [1964]), empíricamente se observa que no hay usualmente problemas numéricos. Más concretamente, no tenemos información sobre dificultades debidas a mal condicionamiento de las matrices en modelizaciones reales. Volveremos sobre este tema del buen condicionamiento del sistema lineal en la sección 8.

La memoria de computadora necesaria para almacenar la matriz **A** y el vector **b** tal que **Ax=b** es la siguiente, si se tienen en total n puntos de discretización:

i) 10 palabras en total para todos los A_{ji} , B_{ji}, j=1,...,5, y C_{ji} , D_{ji} , j=1,...,4, pues el proceso de triangulación avanza de a pares de filas y se puede usar el espacio reservado para las A_{ji} y B_{ji} para almacenar los C_{ji} y D_{ji} (Suponemos, de ahora en adelante, que hay además memoria reservada para almacenar los S_i, B_i, D_i, dD/dZ_i , etc., que determinan A_{ji} y B_{ji} según las ecuaciones (18) y (20), o sea, los A_{ji} y B_{ji} se calculan y usan a medida que se recorren los puntos de discretización en sentido descendente).

ii) 2x2(n-1)=4(n-1) palabras para todos los elementos de la matriz triangular superior, y 2(n-1) palabras para b, o sea 6(n-1) palabras en total.

O sea, aproximadamente 6n palabras.

En cuanto al número de operaciones necesarias para resolver el sistema es:

1 división (pues A_{ii}=-1 siempre, o sea no es necesaria esa división), 7 sumas y 7 multiplicaciones por par de ecuaciones entre puntos de discretización (i,i+1) para la tridiagonalización; si consideramos que una operación equivale a una suma más una multiplicación o una suma más una división, esto significa 8 operaciones por intervalo entre puntos de discretización, es decir, en total 8(n-1) operaciones.

Como para resolver un sistema tridiagonal de orden k se necesitan 3(k-1) multiplicaciones y sumas y 2k-1 divisiones, y nuestra matriz tiene orden k=2(n-1), harán falta 3(2(n-1)-1) multiplicaciones y sumas y 2.2(n-1)-1 divisiones, o sea 10n-14 operaciones.

Por consiguiente, en total habrá 18n - 22 operaciones, o sea aproximadamente 18n operaciones. Si se quiere desglosar en sumas y multiplicaciones por un lado y divisiones por otro (pues éstas suelen demorar más), serán aproximadamente 13n sumas y multiplicaciones y 5n divisiones.

Es interesante comparar la memoria y número de operaciones antes calculados con los necesarios si se considera la matriz A original dada por las ecuaciones (15), (16) como una matriz pentadiagonal (que lo es, con un elemento nulo en cada fila de la banda). La memoria necesaria para almacenar A y b es de 6 palabras por fila, o sea 6x2(n-1)=12(n-1) palabras, es decir, aproximadamente 12n palabras (siempre considerando que n es el número de puntos de discretización, o sea el número de filas es 2(n-1)). En realidad, como en el caso anterior, si triangulamos simultáneamente con el cálculo de los elementos de la banda, para ahorrar espacio, y usamos que cada banda tiene solamente 4 elementos no nulos, esa suma se puede reducir a 4(n-1)+3(n-1)+10 palabras, o sea aproximadamente 7n palabras.

En cuanto al número de operaciones (siempre consideramos pivote diagonal, o sea no hay testeo para buscar el pivote) es directamente, como puede verse en Dahlquist y Björk [1974], capítulo 5, sección 4, de 6x2(n-1)=12(n-1) operaciones para la eliminación y de 5x2(n-1)=10(n-1) operaciones para el barrido descendente, o sea alrededor de 22n operaciones. La comparación favorece ligeramente a nuestro algoritmo, debido a que, como sabemos qué elemento de la banda es nulo (el primero en las filas impares y el último en las filas pares), usamos esa información en el proceso.

Veamos ahora, antes de seguir adelante, un breve ejemplo explicativo. Sea un tramo simple de 4 puntos, x_1, x_2, x_3, y x_4. Supongamos, por ejemplo, que se tienen como condiciones de contorno el caudal dado aguas arriba y el nivel dado aguas abajo. Una vez llevada a cabo la discretización quedará formulado el sistema lineal

$$
\begin{aligned}
A_{21}\Delta Z_1+A_{31}\Delta Q_2+A_{41}\Delta Z_2 &= A_{51}-A_{11}\Delta Q_1\\
B_{21}\Delta Z_1+B_{31}\Delta Q_2+B_{41}\Delta Z_2 &= B_{51}-B_{11}\Delta Q_1\\
A_{12}\Delta Q_2+A_{22}\Delta Z_2+A_{32}\Delta Q_3+A_{42}\Delta Z_3 &= A_{52}\\
B_{12}\Delta Q_2+B_{22}\Delta Z_2+B_{32}\Delta Q_3+B_{42}\Delta Z_3 &= B_{52}\\
A_{13}\Delta Q_3+A_{23}\Delta Z_3+A_{33}\Delta Q_4 &= A_{53}-A_{43}\Delta Z_4\\
B_{13}\Delta Q_3+B_{23}\Delta Z_3+B_{33}\Delta Q_4 &= B_{53}-B_{43}\Delta Z_4
\end{aligned}
$$

donde los A_{ij}, B_{ij}, i=1,..,5, j=1,2,3, tienen los valores antes indicados. Restando la primera fila dividida por A_{11} y multiplicada por B_{11} de la segunda fila, la tercera fila dividida por A_{12} y multiplicada por B_{12} de la cuarta fila y la quinta fila dividida por A_{13} y multiplicada por B_{13} de la sexta fila se tendrá

$$
\begin{aligned}
A_{21}\Delta Z_1+A_{31}\Delta Q_1+A_{41}\Delta Z_2 &= A_{51}-A_{11}\Delta Q_1\\
D_{11}\Delta Z_1+D_{21}\Delta Q_2+D_{31}\Delta Z_2 &= D_{41}\\
A_{12}\Delta Q_2+A_{22}\Delta Z_2+A_{32}\Delta Q_3+A_{42}\Delta Z_3 &= A_{52}\\
D_{12}\Delta Z_2+D_{22}\Delta Q_3+D_{32}\Delta Z_3 &= D_{42}\\
A_{13}\Delta Q_3+A_{23}\Delta Z_3+A_{33}\Delta Q_4 &= A_{53}-A_{43}\Delta Z_4\\
D_{13}\Delta Z_3+D_{23}\Delta Q_4 &= D_{43}-D_{33}\Delta Z_4
\end{aligned}
$$

Ahora se resta la segunda fila dividida por D_{31} y multiplicada por A_{41} de la primera fila, la cuarta fila dividida por D_{32} y multiplicada por A_{42} de la tercera fila y la sexta fila dividida por D_{33} y multiplicada por A_{43} de la quinta fila y queda

$$
\begin{aligned}
C_{21}\Delta Z_1+C_{31}\Delta Q_2 &= C_{41}-C_{11}\Delta Q_1\\
D_{11}\Delta Z_1+D_{21}\Delta Q_2+D_{31}\Delta Z_2 &= D_{41}\\
C_{12}\Delta Q_2+C_{22}\Delta Z_2+C_{32}\Delta Q_3 &= C_{42}\\
D_{12}\Delta Z_2+D_{22}\Delta Q_3+D_{32}\Delta Z_3 &= D_{42}\\
C_{13}\Delta Q_3+C_{23}\Delta Z_3+C_{33}\Delta Q_4 &= C_{43}\\
D_{13}\Delta Z_3+D_{23}\Delta Q_4 &= D_{43}-D_{33}\Delta Z_4
\end{aligned}
$$

y nos queda el sistema $A'x=b'$, con $x=(\Delta Z_1,\Delta Q_2,\Delta Z_2,\Delta Q_3,\Delta Z_3,\Delta Q_4)^t$ y con A' tridiagonal, de modo que se puede resolver por medio del barrido doble.

Resumiendo, la linealización de las ecuaciones de Saint-Venant para un tramo simple discretizado mediante un esquema de diferencias finitas de tipo caja de cuatro puntos, siendo los puntos de discretización designados como $x_o,\ldots x_L$ da origen a un sistema lineal $Ay=b$ que deberá resolverse en cada paso de tiempo, donde:

A es una matriz cuadrada de orden 2L. Las columnas no nulas del primer par de sus filas son las tres primeras. Para las filas 2i-1 y 2i, con $2\leq i<L-1$, los elementos no nulos están ubicados sobre las columnas 2i-2 a 2i+1, es decir, los elementos no nulos, para el par de filas (i-1,i), comienzan a la altura de la tercera columna no nula del par anterior. Para las filas 2L-1 y 2L los elementos no nulos son los correspondientes a las últimas tres columnas. Las filas 2i-1 y 2i son las correspondientes a la discretización en diferencias finitas para un esquema tipo caja de cuatro puntos del intervalo entre los puntos de discretización (i-1,i), y por consiguiente los lados derechos b_{2i-1} y b_{2i} corresponden a los términos independientes de las ecuaciones discretizadas en ese intervalo. Las columnas corresponderán a las incógnitas del siguiente modo:

La primer columna corresponde a la incógnita ΔQ_o (si la condición de contorno es nivel dado) o a la incógnita ΔZ_o (si la condición de contorno es caudal dado). Las columnas 2, 4,.., 2i, ..., 2L-2 corresponden a las incógnitas ΔQ_1, $\Delta Q_2,\ldots,\Delta Q_i,\ldots,\Delta Q_{L-1}$. Las columnas 3, 5,.., 2i+1,..2L-1 corresponden a las incógnitas

ΔZ_1, ΔZ_2,..,ΔZ_i,..ΔZ_{L-1}. La columna 2L corresponde a la incógnita ΔQ_L (si la condición de contorno aguas abajo es el nivel) o ΔZ_L (si la condición de contorno dada aguas abajo es el caudal).

Obsérvese que al efectuar una linealización completa, incluso de los términos no homogéneos, en los incrementos ΔZ y ΔQ, y a linealizar despreciando los términos de orden superior en el desarrollo de Taylor, el método de Preissmann evita la formación de un sistema no lineal.

8. ANALISIS HEURISTICO DE LA ESTABILIDAD NUMERICA DE LA MATRIZ A

Ya hemos mencionado que no se conocen problemas de inestabilidad numérica debida a mal condicionamiento de la matriz **A** en aplicaciones reales de modelos hidrodinámicos unidimensionales que usan el esquema de Preissmann. Dado que en última instancia las ecuaciones que se discretizan son no lineales, tropezamos con problemas del mismo tipo que para el análisis de la estabilidad numérica del esquema como tal. Procederemos de la misma manera para tratar de evaluar el buen condicionamiento de la matriz A: supondremos en primer lugar un cauce fluvial de sección transversal simple y uniforme (canal prismático) y una situación de caudal y alturas desde el lecho del cauce constantes, para ver qué pasa cuando se tridiagonaliza y luego se triangulan las ecuaciones (15.i) y (16.i), y poder extrapolar conclusiones a casos más generales. Examinemos entonces los coeficientes (18) y (20).

Supondremos entonces Δt=1 día, intervalo temporal usual en modelización de períodos prolongados sin crecidas bruscas, o sea Δt=86400 segundos. Es razonable pensar que el intervalo Δx no es mayor que 10 km (éste es usualmente un valor muy exagerado, salvo en ríos muy grandes; con valores menores las cotas numéricas que se indican más abajo son aún menores), y suponemos un ancho de 100 m (cauce mediano). Tomamos Θ=0,8. Entonces, por (18.2) será $A_{2i} = A_{4i} = R < 10$. (Incluso es posible, con Δx más pequeño, que $R < 1$, lo cual mejora el condicionamiento de la matriz A en nuestro ejemplo simplificado, como veremos).

Si consideramos una velocidad media de 1 m/s, totalmente razonable para regímenes subcríticos, y tomando un caudal de 1000 m^3/s (siempre para cauce mediano), se tendrá una superficie mojada S de 1000 m^2. Además, como puede verse por ejemplo en Chow [1959], Q^2/D^2 tiende a la pendiente de superficie del agua para regímenes estacionarios, y este es un número pequeño (digamos menor que 0,0001 para un río mediano de llanura). Eso implica que D será siempre un número grande; por consiguiente, podemos decir respecto de los tres términos de B_{1i} lo siguiente:

$\Delta x/(2g\Delta tS) \sim 10000/(20 \times 86400 \times 1000) << 1$

$\theta Q/(gS^2) \sim 1000/(10 \times 1000000) << 1$

$\theta\Delta x|Q|/D^2 \sim 10000 \times 0,0001 / 1000 < 0,001$

Despreciaremos por lo tanto, en primera aproximación, B_{1i} , y análogamente B_{3i}. Con el mismo razonamiento, las estimaciones de B_{2i} y B_{4i} nos indican que, salvo el término θ (o $-\theta$), los demás términos pueden despreciarse en primera aproximación, pues

$\Delta xQB/(2g\Delta tS^2) \sim 10000 \times 1000 \times 100 / (20 \times 86400 \times 1000 \times 1000)$
$\sim 0,0005$

$\theta Q^2B/(gS^3) \sim 1000000 \times 100 / (10 \times 1000000000) \sim 0,01$

y además $D \sim 100000$. Como $\partial D/\partial Z$ no varía mucho (digamos un orden de magnitud menos que D) se tiene

$\theta Q|Q|\Delta x.\partial D/\partial Z/D^3 \sim 0,0001 \times 10000 / 10 \sim 0,1$

que tampoco tomaremos en cuenta en este análisis heurístico simplificado.

Tomemos ahora otro caso de dimensiones un poco distintas, y calculemos exactamente los coeficientes: sean por ejemplo dos puntos de discretización k, k+1, tales que sus respectivas secciones transversales son rectangulares, de ancho en ambos casos 250 m. Supondremos un caudal en ambos puntos de 1500 m^3/s, un coeficiente de conducción que varía linealmente desde 0 a nivel del fondo hasta 44100 m^3/s a 10 m sobre el nivel del fondo, θ=0,8, Δx=1000 m, Δt=1 día (o sea 86400 s), y una altura sobre el nivel del fondo en ambos puntos de 10 m en el instante conocido. Estos también son datos perfectamente representativos de un río de mediano caudal. Se puede observar que

$A_{1,i}$=-1	$A_{2,i}$= 1,808449	$A_{3,i}$=1	$A_{4,i}$=1,808449
$B_{1,i}$=0,000597691	$B_{2,i}$=-0,889654	$B_{3,i}$=0,000636834	$B_{4,i}$=0,704475

Es decir, $|B_{1,i}|<<1$ y $|B_{3,i}|<<1$. Analizaremos entonces la matriz cuyos pares de filas dados por (15.i) y (16.i) sean, despreciando $B_{1,i}$ y $B_{3,i}$

$$\begin{matrix} -1 & R & 1 & R \\ 0 & -\epsilon & 0 & \epsilon \end{matrix}$$

respectivamente. Veamos cuán bien condicionada (o no) es la matriz con estas filas. La matriz general será (suponiendo condiciones de contorno Q aguas arriba y Z aguas abajo) del tipo

$$\begin{bmatrix} R & 1 & R & & & & & & & \\ -\epsilon & 0 & \epsilon & & & & & & & \\ & -1 & R & 1 & R & & & & & \\ & 0 & -\epsilon & 0 & \epsilon & & & & & \\ & & & -1 & R & 1 & R & & & \\ & & & 0 & -\epsilon & 0 & \epsilon & & & \\ & & & & & \cdot & & & & \\ & & & & & & & \cdot & & \\ & & & & & & & -1 & R & 1 & R \\ & & & & & & & 0 & -\epsilon & 0 & \epsilon \end{bmatrix} \qquad (38)$$

Se observa que al tridiagonalizar (38) según el procedimiento de la sección 7 la matriz se transforma en

$$B= \begin{bmatrix} 2R & 1 & & & & & & & & \\ -\epsilon & 0 & \epsilon & & & & & & & \\ & -1 & 2R & 1 & & & & & & \\ & & -\epsilon & 0 & \epsilon & & & & & \\ & & & -1 & 2R & 1 & & & & \\ & & & & -\epsilon & 0 & \epsilon & & & \\ & & & & & \cdot & & & & \\ & & & & & & \cdot & & & \\ & & & & & & -1 & 2R & 1 & \\ & & & & & & & -\epsilon & 0 & \epsilon \\ & & & & & & & & -1 & 2R & 1 \\ & & & & & & & & & -\epsilon & 0 \end{bmatrix} \qquad (39)$$

Esta matriz no es diagonal dominante. Volviendo al ejemplo concreto de antes, la tridiagonalización produce los siguientes resultados:

$C_{1i}=-1$ $\quad C_{2i}=4,0859948$ $\quad C_{3i}=0,99683563$

$D_{1i}=-0,88857311$ $\quad D_{2i}=0,00123453$ $\quad D_{3i}=0,70555589$

Claramente no es diagonal dominante.

Para esta triangulación, por la forma de las matrices, las filas impares no se modifican, y a las filas pares 2i se les suman las filas impares siguientes, 2i+1, multiplicadas por -R/ϑ. Dado que usualmente 0,1 < R/ϑ <10, y dado que el error no se propaga de fila en fila, porque las operaciones con filas se realizan de a pares *aislados* de filas, despreciaremos el error cometido al pasar de la matriz **A** a la matriz **B** dada por (39), o sea nos dedicaremos a estudiar el buen condicionamiento de **B** como equivalente al de **A**.

Si aplicamos a esta matriz tridiagonal **B** una eliminación gaussiana, obtendremos que **B=LU**, con **L** "bidiagonal inferior", o sea triangular inferior con una sola diagonal menor debajo de la diagonal principal no nula, y con **U** "bidiagonal superior", o sea triangular superior con una sola diagonal menor encima de la diagonal principal no nula, siendo

$$L = \begin{bmatrix} 1 & & & & & & & & \\ -\vartheta/(2R) & 1 & & & & & & & \\ & -2R/\vartheta & 1 & & & & & & \\ & & -\vartheta/(4R) & 1 & & & & & \\ & & & -4R/\vartheta & 1 & & & & \\ & & & & & \ddots & & & \\ & & & & & -2(L-1)R/\vartheta & 1 & & \\ & & & & & & -\vartheta/(2LR) & 1 \end{bmatrix} \qquad (40)$$

$$U = \begin{bmatrix} 2R & 1 & & & & & & & \\ & \vartheta/(2R) & \vartheta & & & & & & \\ & & 4R & 1 & & & & & \\ & & & \vartheta/(4R) & \vartheta & & & & \\ & & & & 6R & 1 & & & \\ & & & & & & \ddots & & \\ & & & & & & 2LR & 1 & \\ & & & & & & & \vartheta/(2LR) & \end{bmatrix} \qquad (40')$$

$(x_0,\ldots,x_L)$ es el conjunto ordenado de puntos de discretización, o sea hay L intervalos de cálculo y 2L filas en la matriz. Los elementos no nulos de L fuera de la diagonal son los multiplicadores

$$m_{2i,2i-1} = -\vartheta/(2iR)$$
$$m_{2i+1,2i} = -2iR/\vartheta$$

y los elementos de la diagonal de **U** son

$$u_{2i-1,2i-1} = 2iR$$
$$u_{2i,2i} = \Theta/(2iR)$$

Los elementos $u_{i,i+1}$ de **U** coinciden con los de **B**. Veremos cómo se propaga el error de redondeo en la triangulación de B, es decir, en el proceso de obtención de la factorización **B=LU**.

TEOREMA: Sean $\bar{m}_{i,j}$ y $\bar{u}_{i,j}$ los valores aproximados calculados por computadora de $m_{i,j}$ y u_{ij}. Sea u la unidad de redondeo o de corte de la computadora, o sea $u=2^{-(t+1)}$ o $u=2^{-t}$, respectivamente, donde t es el número de dígitos significativos de la mantisa en la representación normalizada de los números reales en punto flotante por computadora. Entonces, para la factorización **B=LU** de (39) dada por (40) se tiene, si se desprecian los términos de orden $O(u^2)$

$$\bar{m}_{i,i-1} = m_{i,i-1}\,(1 + (3i-2)\varepsilon_{i-1,i})$$

$$\bar{u}_{i,i} = u_{ii}\,(1 + 3i\varepsilon_{i,i})$$

donde $|\varepsilon_{i,j}| \leq u \qquad i \geq 2.$

DEMOSTRACION: La demostración es por inducción en i. Para i=2 se tiene

$$\bar{m}_{2,1} = -\Theta/(2R).(1 + \varepsilon) \text{ para } |\varepsilon| \leq u \text{ o sea } \varepsilon_{2,1}=\varepsilon/4$$

$\bar{u}_{2,2} = \Theta/(2R).(1 + \varepsilon)$ y valen las consideraciones anteriores. (Se considera que la multiplicación por 1 es exacta, lo mismo que la suma de cero).

Veamos ahora qué pasa para i > 2. Sea i impar. Entonces se tiene

$$\bar{m}_{i,i-1} = -1/\bar{u}_{i-1,i-1}(1 + \varepsilon) \quad \text{con } |\varepsilon| \leq u$$

$$=-1/(u_{i-1,i-1}.(1 + 3(i-1)\varepsilon_{i-1,i-1})).(1+\varepsilon) \quad \text{con } |\varepsilon_{i-1,i-1}| \leq u$$

$$= m_{i,i-1}.(1 + \varepsilon).(1 - 3(i-1)\varepsilon_{i-1,i-1} + O(u^2))$$

$$= m_{i,i-1}.(1 + \varepsilon - 3(i-1)\varepsilon_{i-1,i-1}) + O(u^2)$$

$$= m_{i,i-1}.(1 + (3i-2)\varepsilon_{i,i-1}) = -(i-1)R/\Theta.(1 + (3i-2)\varepsilon_{i,i-1})$$

Para la última igualdad se despreció $O(u^2)$ y se tomó

$\varepsilon_{i,i-1} = (\varepsilon - 3(i-1)\varepsilon_{i-1,i-1})/(3i-2)$ y se tiene

$|\varepsilon_{i,i-1}| \leq (u + 3(i-1)u)/(3i-2) = (3i-2)u/(3i-2) = u$

Por su parte, para $u_{i,i}$ se tiene

$u_{i,i} = (2R - m_{i,i-1}.u_{i-1,i}.(1+\varepsilon_1)).(1+\varepsilon_2)$

$= (2R + (i-1)R/\Theta.(1+(3i-2)\varepsilon_{i,i-1}).\Theta.(1+\varepsilon_1)).(1+\varepsilon_2)$

$= (2R + (i-1)R.(1+(3i-1)\varepsilon+O(u^2)).(1+\varepsilon_2)$

con $\varepsilon=((3i-2)\varepsilon_{i,i-1}+\varepsilon_1)/(3i-1)$, $|\varepsilon| \leq u$

$= ((i+1).R.(1 + (3i-1)\varepsilon').(1+\varepsilon_2)$

despreciando $O(u^2)$ y tomando $\varepsilon'=(i-1)/(i+1)\varepsilon$

$= (i+1)R.(1+ 3i\varepsilon_{i,i})$ tomando $\varepsilon_{i,i}=((3i-1)\varepsilon'+\varepsilon_2)/(3i)$ y despreciando de nuevo $O(u^2)$.

Sea ahora i par. Entonces se tiene

$m_{i,i-1} = -\Theta/u_{i-1,i-1}.(1+\varepsilon)$

$= -\Theta/(u_{i-1,i-1}.(1+3(i-1)\varepsilon_{i-1,i-1})).(1+\varepsilon)$

$= m_{i,i-1}.(1-3(i-1)\varepsilon_{i-1,i-1}+O(u^2)).(1+\varepsilon)$

$= -\Theta/(iR).(1+(3i-2)\varepsilon_{i,i-1})$

tomando $\varepsilon_{i,i-1} = (-3(i-1)\varepsilon_{i-1,i-1} + \varepsilon)/(3i-2)$ y despreciando $O(u^2)$

En cuanto a $u_{i,i}$, se tiene

$u_{i,i} = \Theta/(iR).(1+3i\varepsilon_{i,i})$ tomando $\varepsilon_{i,i}= (3i-2)\varepsilon_{i,i-1}/(3i)$

con lo que queda demostrado el teorema. ■

Es decir: B se puede triangular por el método de eliminación de Gauss sin que se produzca en ningún momento división por 0, pues los pivotes exactos no se anulan nunca y los aproximados son muy poco perturbados: la perturbación numérica máxima de los multiplicadores es del orden de 1 + 3Lu, y 3Lu <<1, dado que el orden de la matriz 2L (normalmente menor que 200) es mucho menor que t.

Con el mismo razonamiento, usamos el teorema (ver por ejemplo Dahlquist y Björk [1974], sección 5.5.4) que dice que las matrices $\mathcal{L}$ y $\mathcal{U}$ computadas cumplen que

$$\mathcal{L}\mathcal{U} = B + E$$

es decir, son la solución exacta de un producto de matrices que difiere de B en una matriz de perturbación $E=(e_{i,j})$, donde

$$|e_{i,j}| \leq 3\ (2L-1)\ u\ \max_k |b_{i,j}^{(k)}|$$

donde $b_{i,j}^{(k)}$ indica el elemento (i,j) de la matriz obtenida siguiendo el procedimiento de eliminación de Gauss hasta la k-sima fila. Pero en nuestro caso, como vimos en el reciente teorema.

$$|b_{i,i}^{(k)}| = |u_{i,i}| \leq 2LR.(1+6Lu)$$

y además

$$|b_{i,i+1}^{(k)}| = |b_{i,i+1}| \leq 1$$

o sea

$$\max_k\ |\ b_{i,j}^{(k)}\ | \leq 2LR.(1+6Lu)$$

Por otra parte, en el lenguaje TURBOPASCAL de la computadora Toshiba 3100 bajo sistema operativo MS/DOS en que se prepararon los programas y se realizaron los experimentos numéricos de este trabajo, los números reales se representan con una mantisa de 40 dígitos binarios, o sea t=40, equivalente a aproximadamente 12 dígitos decimales, o sea (la representación en computadora es por corte)

$$|e_{i,j}| \leq 3.200.10^{-12}.200.10\ (1 + 600.10^{-12})$$

$$\leq 12.\ 10^{2}\ .\ 10^{-12}\ .\ 10^{2}\ .\ 10\ .\ (1 + 10^{-9})$$

$$\sim 12.\ 10^{-7}$$

o sea la perturbación (en este caso muy simplificado, claro está), es muy pequeña.

Claramente, éste es un análisis heurístico, pues hemos considerado un caso particular. Sin embargo, el mismo representa adecuadamente los casos que se presentan en la práctica. De todos modos, no hemos proseguido la investigación pues para ello debe tomarse en cuenta el número de condición de **B** (o de **A**), cond(**B**) o cond(**A**), y eso implica estimarlo, siguiendo la línea, por ejemplo, de Golub y Van Loan [1983], capítulo 4. El estudio sistemático de este problema está fuera de los objetivos de esta tesis.

SEGUNDA PARTE

9. ESTRUCTURAS FLUVIALES ARBORESCENTES

En una red fluvial se tienen afluentes que desembocan en otros cursos de agua más importantes, efluentes por donde se deriva parte del caudal, etc. Como ya indicamos en la sección 3, además de las ecuaciones de Saint-Venant (1), de las condiciones iniciales (2) y de las condiciones de contorno (10) en extremos de la red fluvial - que ahora podrán ser más de uno aguas arriba y más de uno aguas abajo - será necesario imponer las condiciones de compatibilidad (12) de Stoker en las confluencias de la red. Pero la solución de las ecuaciones de Saint-Venant deberá hallarse también numéricamente, por supuesto; y si usamos el método de Preissmann, tendremos que resolver en cada intervalo de tiempo un sistema lineal **Ax=b**. Interesa ver si la matriz **A** en cada intervalo de tiempo será también fácilmente tridiagonalizable, o si su estructura tendrá otras características convenientes. Convendrá por lo tanto definir con precisión los conceptos que se manejan, aunque resulten intuitivos.

Definición: Una *discretización de una red fluvial* es una elección de un número finito de secciones transversales en los cauces de la red fluvial, tales que cada vez que se produce una confluencia (dos tramos que afluyen a un tercero o un tramo que se bifurca), a dicha confluencia están asociados tres puntos de discretización, uno correspondiente a cada curso de agua que afluye o efluye, tal como se indica en la figura 3.

Nos interesarán en esta parte del trabajo las cuencas fluviales arborescentes, o sea las redes fluviales formadas por ríos, afluentes, afluentes de afluentes, etc. En este contexto las confluencias serán siempre afluencias. En lo que sigue nos conviene usar algunos conceptos de teoría de grafos, para lo cual usaremos terminología (traducida) de Christofides [1975] y Even [1979]. Indicamos a continuación las definiciones que necesitamos:

Un *grafo dirigido* está formado por un conjunto V de *vértices* $v_1, v_2, \ldots$, y un conjunto E de *arcos* e_i que son pares ordenados (v_i, v_j) , es decir, $E \subset V^2$. Si (v_i , v_j) es un arco, v_i es el vértice de *origen* y v_j es el vértice de *destino* del arco; *grado de*

entrada $gr_E(v)$ es el conjunto de los arcos de los cuales v es vértice de destino, *grado de salida* $gr_S(v)$ es el conjunto de los arcos de los cuales v es vértice de origen, y *grado* $gr(v) = gr_E(v) + gr_S(v)$ es la suma del grado de entrada más el grado de salida. Un *camino dirigido* $e_1, e_2, \ldots, e_k$ es una sucesión de arcos tal que el vértice de destino de e_{i-1} es el vértice de origen de e_i. El vértice de origen de e_1 es el vértice de origen del camino; el vértice de destino de e_k es el vértice de destino del camino. Un *circuito* es un camino dirigido tal que su vértice de origen coincide con su vértice de destino. Un grafo dirigido G es *débilmente conexo* si para cada par de vértices v_i , v_j hay un camino no dirigido entre ellos, es decir, un conjunto ordenado de vértices v_i , v_{i+1} , v_{i+2} , ... , v_{j-1} , v_j tales que para cada par consecutivo (v_l, v_{l+1}), el arco (v_l, v_{l+1}) pertenece a E o el arco (v_{l+1}, v_l) pertenece a E. En adelante hablaremos de camino a secas, entendiéndose que es dirigido. Un grafo dirigido tiene una *raíz* r si $r \in V$ y todo vértice es alcanzable desde r, es decir, para todo vértice v existe un camino cuyo origen es r y su destino es v. Un *àrbol dirigido* o *arborescencia* es un grafo tal que tiene una (única) raíz r con $gr_E(r)=0$ y para todo otro vértice v, $gr_E(v)=1$. Se prueba en Even [1979], capítulo 2, sección 4, que la definición de arborescencia es equivalente a decir que G tiene una raíz desde la cual hay un único camino a todo otro vértice; y también es equivalente a decir que G tiene una raíz y si se quita un arco (pero ningún vértice) al grafo la raíz deja de serlo. Si G es una arborescencia y se tiene un vértice v tal que $gr_S(v) = 0$, se dice que v es una *hoja* del árbol dirigido.

Ahora estamos en condiciones de definir los conceptos relacionados con las discretizaciones que efectuaremos.

Definición: Una *discretización arborescente de una red fluvial* es un grafo dirigido finito G(V,E) tal que

a) G(V,E) es un árbol orientado (o arborescencia).

b) Los vértices $v \in V$ son exactamente los puntos de discretización de la red fluvial.

c) Si gr(v), $gr_E(v)$ y $gr_S(v)$ indican, respectivamente, el grado, el grado de entrada y el grado de salida del vértice v, vale:

c1) $gr(v) \leq 3$.

c2) Si $gr_E(v) = 0$ entonces $gr_S(v) = 1$ (o sea, de la raíz de G sale solamente un arco).

d) Si $gr(v_k) = 3$, y si v_i y v_j son los dos vértices a los cuales llegan los arcos (v_k , v_i) y (v_k , v_j) de los cuales v_k es el vértice de origen, entonces vale que los puntos de discretización forman una confluencia $\{v_i, v_j, v_k\}$ como la afluencia que se indica en la figura 3, en que los cursos de agua que terminan en v_i y v_j afluyen al que comienza en v_k . Recíprocamente, dada una confluencia sobre la cual hay tres puntos de discretización v_i , v_j y v_k , donde v_k es extremo aguas arriba de un curso formado por la afluencia de dos cursos que terminan en v_i y v_j . vale que $gr(v_k)=3$ y que (v_k,v_i) y (v_k,v_j) son arcos del grafo, o sea $(v_k,v_i) \in E$ y $(v_k, v_j) \in E$.

e) Dados dos arcos consecutivos (v_i , v_j) y (v_j , v_k) tales que $gr_E(v_j) = gr_S(v_j) = 1$, los puntos de discretización correspondientes están sobre un cauce de red fluvial y, yendo desde aguas abajo hacia aguas arriba, primero está v_i , luego v_j y finalmente v_k , y no hay ningún punto de discretización entre ellos.

Es decir, se da al grafo la dirección "desde aguas abajo hacia aguas arriba", y las confluencias son afluencias: indican siempre una unión de dos afluentes: en las redes arborescentes no hay efluentes.

Si v es la raíz del grafo (o sea $gr_E(v) = 0$) diremos que v es el (único) *extremo abierto aguas abajo* de la red, o la *desembocadura* de la red; si $gr_S(v) = 0$ (o sea v es una hoja del grafo) diremos que v es *extremo abierto aguas arriba* de la red; si $gr(v) = 3$, y $(v,w) \in E$, $(v,z) \in E$, reiteremos que la terna $\{w,z,v\}$ forma un *nodo de afluencia*. Escribiremos siempre como último elemento de la terna al vértice v cuyo grado de salida sea 2. En este caso diremos que w y z son extremos aguas arriba de sendos tramos que *afluyen al* (o *son afluentes del*) tramo del cual v es extremo aguas arriba.

La dirección del grafo será, como ya dijimos, desde aguas abajo hacia aguas arriba; en la tercera parte de este trabajo usaremos, para redes fluviales deltaicas, grafos con la dirección opuesta, es decir, desde aguas arriba hacia aguas abajo, que parece más "natural". El motivo de usar esta dirección para las redes arborescentes se debe a que es la dirección de los árboles dirigidos, en los que la raíz (o sea, la desembocadura) tiene grado de entrada cero, y por lo tanto cambiar la dirección complicaría innecesariamente nuestro análisis.

Los pares de ecuaciones obtenidos según el método de Preissmann ligan los vértices unidos por un arco, con excepción de vértices tales que el vértice de comienzo del grafo tenga grado de salida 2. Analizaremos en este trabajo situaciones en que se dará una condición de contorno sobre cada uno de los extremos de la red fluvial, y las condiciones de compatibilidad (12) sobre los nodos de afluencia. Estas situaciones corresponden a régimen subcrítico en toda la red; comentaremos algo más al respecto en la sección 11.

Un *tramo simple discretizado* de la red fluvial es un conjunto ordenado $\{v_1, v_2, \ldots, v_n\}$ de vértices (es decir, de puntos de discretización de la red) tales que (v_1,v_2), $(v_2,v_3),\ldots,(v_{n-1},v_n)$ forman un camino del grafo G y además se cumple:

a) $gr_S(v_1)=gr_S(v_2)=\ldots=gr_S(v_{n-1})=1$.

b) $gr_S(v_n)=2$ o $gr_S(v_n)=0$.

c) Si v_1 no es la raíz del grafo, entonces si $w \in V$ es tal que $(w,v) \in E$, vale que $gr_S(w) = 2$.

Es decir, un tramo simple discretizado es el conjunto de puntos de discretización que forman un camino entre (e incluyéndolos) dos extremos abiertos (este caso sólo se da si no hay confluencias), dos puntos de discretización pertenecientes a nodos de afluencia, o un punto de discretización perteneciente a un nodo de afluencia y otro extremo abierto de la red. En el interior de un tramo simple discretizado no puede haber puntos que pertenezcan a nodos de afluencia. Si v es el origen del tramo simple discretizado, v será el *extremo aguas abajo* del tramo, y si es el destino, será el *extremo aguas arriba* del tramo.

El tramo de río discretizado con esos puntos de discretización será un *tramo simple*; es decir, un tramo simple de una red fluvial es cada uno de los tramos de la red cuyos extremos son o extremos de la red o confluencias, de modo que en su interior no hay ninguna confluencia.

El concepto de tramo simple será usado no solamente en redes arborescentes sino también en la tercera parte de este trabajo, cuando estudiemos las redes deltaicas. Las diferencias de ambas estructuras se pueden ver gráficamente en las figuras 6 y 7. En la figura 6, (red arborescente) los extremos de la red son los puntos 1, 2 y 4 (extremos aguas arriba), y 6 (extremo aguas abajo), los puntos en que se producen confluencias son 3 y 5, y los tramos simples son (1,3), (2,3), (3,5), (4,5) y (5,6). En la figura 7 (red deltaica), los extremos de la red son los puntos 1, 2, 3 (extremos aguas arriba), y 11 y 12 (extremos aguas abajo), los puntos en que se producen confluencias son los puntos 4 a 10, y los tramos simples son (1,8), (2,4), (3,7), (4,5), (4,6), (5,7), (5,6), (6,8), (7,9), (8,9), (9,10), (10,11), (10,12). Por simplicidad, hablaremos de tramo simple tanto para referirnos a tramos simples como a tramos simples discretizados.

En cada tramo simple se asignan números reales x_i a los vértices v_i de modo que para el tramo simple $\{v_1, v_2, \ldots, v_n\}$ se tiene $x_1<x_2<\ldots<x_n$ o $x_1>x_2>..>x_n$. Esos números reales representan la distancia de cada nodo a un punto de referencia sobre el eje longitudinal del tramo (la *progresiva*, según la terminología de ingeniería hidráulica), y es indistinto si las coordenadas aumentan o disminuyen en la dirección del grafo, pues lo que importa es la distancia $|x_i-x_{i+1}|$ entre dos puntos de discretización consecutivos. Las coordenadas de los puntos de discretización de cada tramo simple son independientes de las de cualquier otro tramo simple. En lo sucesivo indicaremos un punto de discretización también por su coordenada x_i si no se crean confusiones (pues dos puntos de discretización sobre distintos tramos simples pueden tener la misma coordenada).

Como ya mencionamos, las condiciones de contorno del sistema hiperbólico (1), (2), con condiciones de compatibilidad (12) deberán darse sobre extremos abiertos. Y si el régimen del curso fluvial es subcrítico, se dará exactamente una condición de contorno sobre cada extremo abierto, o sea

$$F_v(Q(x_v,t),Z(x_v,t))=f_v(t) \qquad (10.3)$$

para cada x_v extremo abierto de la discretización arborescente.

10. ALGORITMO DE TRIANGULACION PARA REDES FLUVIALES ARBORESCENTES

El algoritmo a usarse para la triangulación de la matriz correspondiente a una red fluvial arborescente depende de la numeración de los puntos de discretización. Se usarán las condiciones de compatibilidad de Stoker, requeridas en cada nodo de confluencia, y cuya discretización es muy simple:

$$\Delta Q_i + \Delta Q_j = \Delta Q_k \qquad (41.1)$$

$$\Delta Z_i = \Delta Z_j = \Delta Z_k \qquad (41.2)$$

cuando i,j y k son los subíndices correspondientes a los respectivos puntos del nodo de confluencia (v_i,v_j,v_k).

La numeración de los puntos de discretización es la siguiente:

Algoritmo de numeración de puntos de discretización para redes fluviales arborescentes:

Sea n el número total de puntos de discretización.

a) Asignemos el número n a la raíz (extremo abierto aguas abajo de la red).

b) Se numeran los puntos de discretización del tramo simple cuyo extremo aguas abajo es la raíz del siguiente modo: si el tramo simple tiene m puntos de discretización consecutivos desde la raíz (es, decir, la desembocadura) v_n hasta su extremo aguas arriba, su numeración será v_n, v_{n-1},..., v_{n-m+2}, v_{n-m+1}.

c) Al término de esto, pueden pasar dos cosas:

1) Se llega a un punto de discretización cuyo grado de salida es 2. En ese caso se pasa a d).

2) Se llega a un punto de discretización cuyo grado de salida es 0 (un extremo abierto aguas arriba). En este caso, si la numeración de dicho punto de discretización es mayor que 1, se pasa a e). Y si es igual a 1 (lo cual debe pasar en algún momento, pues el árbol es finito), se termina el algoritmo.

d) Se elige uno de los dos tramos simples afluentes del tramo recién numerado, y se lo recorre hacia aguas arriba, es decir, se sigue el camino desde el extremo aguas abajo hasta el extremo aguas arriba, numerando los puntos de discretización consecutivamente en orden decreciente, comenzando en k-1 si k era el número secuencial asignado al extremo aguas arriba del tramo recién numerado. Se vuelve a c).

e) Se toma v tal que

1) v ya ha sido numerado.

2) $gr_s(v)=2$ (o sea v es extremo aguas arriba de un tramo ya numerado).

3) Si w y z son tales que $(v,w) \in E$, $(v,z) \in E$, entonces w fue numerado y z no, o w no fue numerado y z sí (una de las dos alternativas debe cumplirse, por c) y d)).

4) v tiene el número secuencial más alto entre los puntos de discretización que cumplen 1), 2) y 3).

5) Sea w ya numerado y z no. Entonces se recorre el tramo simple cuyo extremo aguas abajo es z desde aguas abajo hacia aguas arriba, numerando los puntos de discretización consecutivamente en orden decreciente, comenzando por k si k-1 era el número secuencial del último punto de discretización numerado. Se pasa a c). Análogamente si el ya numerado es z, se recorre el tramo simple cuyo extremo aguas abajo es w hacia aguas arriba de la misma manera.

De este modo, se tiene la seguridad de que la numeración de un tramo afluente es siempre anterior a la del tramo al cual afluye, y de que dado un nodo de confluencia (v_i,v_j,v_k), será i=k-1 o j=k-1. Si i=k-1, será j<i, y en caso contrario será i<j. Intuitivamente, el tramo unión del tramo simple con extremo aguas abajo v_{k-1} con el tramo simple con extremo aguas arriba v_k será el "tramo principal" respecto del tramo con extremo aguas abajo v_j (si i=k-1) o v_i (si j=k-1), que será su afluente.

La figura 8 ejemplifica una numeración de los puntos de una red fluvial arborescente según este procedimiento.

Si en lugar de árbol dirigido G pensamos en la noción de árbol binario, en estructuras de información, tal como lo define, por ejemplo, Knuth [1975], capítulo 2, sección 3, la formalización de nuestra estructura fluvial es tal vez más trabajosa, pero el algoritmo de numeración de la red es inmediato: basta atravesar el árbol binario en postorden. En efecto, la definición de árbol binario es la siguiente: *un àrbol binario es un conjunto finito de puntos que es vacío, o consiste de una raíz y dos àrboles binarios disjuntos, llamados subàrboles izquierdo y derecho de la raíz.*

Con esta definición (que es recursiva), definimos una discretización de una red fluvial como un árbol binario cuyos puntos son los puntos de discretización de la red y tal que:

i) La raíz del árbol es el extremo abierto aguas abajo de la red.

ii) Las hojas (o sea los puntos del árbol que son raíces de dos subárboles vacíos) son los extremos aguas arriba de la red.

iii) Los nodos de confluencia $\{v_1 , v_2 , v_3\}$, donde v_3 es el extremo aguas arriba, son exactamente aquéllos en los que v_3 es raíz de dos subárboles no vacíos, las raíces de los cuales son v_1 y v_2 , respectivamente.

iv) Los tramos simples son los conjuntos $\{v_1, v_2,\ldots,v_n\}$ tales que cada $i < n$ es raíz de un solo subárbol no vacío; para cada i los puntos $\{v_{i+1} , v_{i+2} ,\ldots,v_n\}$ pertenecen a dicho subárbol cuya raíz es v_{i+1}; v_n es raíz de dos subárboles no vacíos (pertenece como extremo aguas arriba a un nodo de confluencia) o de ninguno (es extremo abierto).

v) Si v es la raíz del árbol total (o sea es el extremo abierto aguas abajo) tiene un solo subárbol no vacío.

Se puede ver fácilmente que los extremos, nodos de afluencia y tramos simples así definidos coinciden con los definidos anteriormente. Y entonces el proceso de numeración que queremos es el de recorrido en postorden, o sea dado el número de rotulación que inicialmente es cero el procedimiento "rotular" consiste en:

1) Rotular subárbol a izquierda.

2) Rotular el subárbol a derecha.

3) Incrementar en uno el número de rotulación y rotular la raíz.

Con este algoritmo recursivo se ve la potencia de nuestra definición de árbol binario, aunque sea menos intuitivo asociar este concepto a nuestra red fluvial arborescente.

Ahora estamos en condiciones de triangular la matriz del esquema de cuatro puntos. Veremos que es posible una triangulación en la cual sobre cada fila hay a lo sumo dos elementos no nulos.

En primer lugar, recordemos el proceso de triangulación de una matriz tridiagonal. Si la matriz de orden n tiene forma

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i \qquad i=1,\ldots n$$

con $a_1=0$, $c_n=0$, la triangulación consiste en pasar el sistema de ecuaciones a la forma

$$\alpha_i x_i + \beta_i x_{i+1} = \gamma_i$$

con $\beta_n=0$ y con α_i , β_i y γ_i dadas por

$$\alpha_1=b_1 \qquad \beta_1=c_1 \qquad \gamma_1=d_1$$

$$\alpha_i=b_i - (a_i/\alpha_{i-1})\beta_{i-1} \qquad \beta_i=c_i \qquad \gamma_i=d_i-(a_i/\alpha_{i-1})\gamma_{i-1} \qquad i>1$$

para poder después resolver el sistema ("barrido descendente") en la forma

$$x_n=\gamma_n/\alpha_n$$

$$x_i=(\gamma_i-\beta_i x_{i+1})/\alpha_i \qquad i<n$$

Esto es, por supuesto, la factorización LU con pivote diagonal aplicada a matrices tridiagonales.

En nuestro esquema de cuatro puntos para tramos simples hay dos tipos distintos de incógnitas (ΔQ_i y ΔZ_i) que conviene distinguir, y por eso usaremos una notación diferenciada, quedando las ecuaciones (con identificación obvia)

$$A_{1,1,i}\Delta Z_i + A_{1,2,i}\Delta Q_{i+1} = A_{1,3,i} \qquad (42.i)$$

$$A_{2,1,i}\Delta Q_{i+1} + A_{2,2,i}\Delta Z_{i+1} = A_{2,3,i} \qquad (43.i)$$

El armado de la matriz triangular correspondiente al barrido descendente es entonces el siguiente, recorriendo la numeración de la red en sentido ascendente:

i) Punto extremo v_i aguas arriba:

i.i) Condición de contorno Q conocida.

Será entonces

$$\begin{aligned} A_{1,1,i} &= C_{2,i} \qquad A_{1,2,i} = C_{3,i} \qquad A_{1,3,i} = C_{4,i} - C_{1,i}\Delta Q_i \\ A_{2,1,i} &= D_{2,i} - A_{1,2,i}(D_{1,i}/A_{1,1,i}) \\ A_{2,2,i} &= D_{3,i} \\ A_{2,3,i} &= D_{4,i} - A_{1,3,i}(D_{1,i}/A_{1,1,i}) \end{aligned} \qquad (44.i)$$

i.ii) Condición de contorno Z conocida:

En este caso queda

$$\begin{aligned} A_{1,1,i} &= C_{1,i} \qquad A_{1,2,i} = C_{3,i} \qquad A_{1,3,i} = C_{4,i} - C_{2,i}\Delta Z_i \\ A_{2,1,i} &= D_{2,i} \qquad A_{2,2,i} = D_{3,i} \qquad A_{2,3,i} = D_{4,i} - D_{1,i}\Delta Z_i \end{aligned} \qquad (45.i)$$

ii) Punto v_i intermedio (no es inicial ni final de tramo simple)

Será

$$A_{1,1,i}=C_{2,i}- (C_{1,i}/A_{2,1,i-1})A_{2,2,i-1}$$

$$A_{1,2,i}=C_{3,i}$$

$$A_{1,3,i}=C_{4,i}- (C_{1,i}/A_{2,1,i-1})A_{2,3,i-1}$$

$$A_{2,1,i}=D_{2,i} - (D_{1,i}/A_{1,1,i})A_{1,2,i} \qquad (46.i)$$

$$A_{2,2,i}=D_{3,i}$$

$$A_{2,3,i}=D_{4,i}- (D_{1,i}/A_{1,1,i})A_{1,3,i}$$

iii) Punto extremo v_i aguas abajo. En este caso, por la forma de barrer los puntos, hay tres posibilidades:

a) El extremo v_i forma parte de un nodo de afluencia $\{v_i,v_j,v_k\}$ con $k > i+1$.

b) El extremo v_i forma parte de un nodo de afluencia $\{v_i,v_j,v_k\}$ con $k = i+1$.

c) El extremo v_i es el punto extremo abierto aguas abajo. O sea $i=n$.

TEOREMA: Estas son las únicas posibilidades de un extremo aguas abajo.

DEMOSTRACION: La única alternativa adicional es que v_i pertenezca a un nodo de afluencia $\{v_i,v_j,v_k\}$, con $k<i$. Pero la recorrida del árbol desde el primer nodo es simplemente una recorrida en sentido inverso al elegido para la numeración original, y $k<i$ indicaría que un tramo afluente fuera recorrido antes que un tramo la cual afluye, lo cual contradice el método usado. ■

Analicemos los casos a) y b) anteriores conjuntamente con el caso d) siguiente:

d) Punto extremo aguas arriba v_i que forma parte del nodo de afluencia $\{v_{i-1},v_j,v_i\}$, con $j<i$. (éste es el único caso que faltaba analizar).

Se tiene

$$A_{2,1,i-2}\Delta Q_{i-1}+A_{2,2,i-2}\Delta Z_{i-1}=A_{2,3,i-2} \qquad (43.i-2)$$

$$A_{2,1,j-1}\Delta Q_{j}\ +A_{2,2,j-1}\Delta Z_{j}\ =A_{2,3,j-1} \qquad (43.j-1)$$

Como por las condiciones de compatibilidad de Stoker se tiene $\Delta Q_{i-1}+\Delta Q_{j}=\Delta Q_{i}$, $\Delta Z_{i-1}=\Delta Z_{j}=\Delta Z_{i}$ valdrá

$$A_{2,1,i-2}\Delta Q_{i-1}+A_{2,2,i-2}\Delta Z_{i}=A_{2,3,i-2}$$

$$A_{2,1,j-1}\Delta Q_{j}\ +A_{2,2,j-1}\Delta Z_{i}=A_{2,3,j-1}$$

$$A_{2,1,i-1}\Delta Q_{i}\ +A_{2,2,i-1}\Delta Z_{i}=A_{2,3,i-1} \qquad (43.i-1)$$

con

$$A_{2,1,i-1}=1$$

$$A_{2,2,i-1}=A_{2,2,i-2}/A_{2,1,i-2}+A_{2,2,j-1}/A_{2,1,j-1}$$

$$A_{2,3,i-1}=A_{2,3,i-2}/A_{2,1,i-2}+A_{2,3,j-1}/A_{2,1,j-1}$$

De la ecuación (42.i-1) se infiere cómo continuar hacia aguas abajo recorriendo el tramo que comienza en i: se realizan las operaciones indicadas en (46) para obtener las ecuaciones (42.i) y (43.i), y se sigue.

Queda por analizar el caso c) para el barrido descendente. En este caso se tiene

c.1) Extremo abierto v_n aguas abajo con caudal Q conocido.

Será

$$\begin{aligned} A_{1,1,n-1}\Delta Z_{n-1} &= A_{1,3,n-1}-A_{1,2,n-1}\Delta Q_{n} \\ A_{2,2,n-1}\Delta Z_{n} &= A_{2,3,n-1}-A_{2,1,n-1}\Delta Q_{n} \end{aligned} \qquad (47.n-1)$$

de donde se obtienen ΔZ_{n-1} y ΔZ_{n}.

c.2) Extremo abierto v_n aguas abajo con nivel Z conocido.

Será

$$A_{1,1,n-1}\Delta Z_{n-1}+A_{1,2,n-1}\Delta Q_n = A_{1,3,n-1} \qquad (48.n-1)$$

$$A_{2,1,n-1}\Delta Q_n = A_{2,3,n-1}-A_{2,2,n-1}\Delta Z_n$$

de donde se calcula ΔQ_n y luego ΔZ_{n-1}.

Barrido ascendente: de c1 o c2 se obtiene ΔZ_{n-1}. A partir de allí se sigue iterativamente

$$\Delta Q_i=(A_{2,3,i-1}-A_{2,2,i-1}\Delta Z_i)/A_{2,1,i-1}$$

$$\Delta Z_{i-1}=(A_{1,3,i-1}-A_{1,2,i-1}\Delta Q_i)/A_{1,1,i-1}$$

hasta llegar al extremo v_i aguas arriba del tramo simple en cuestión. En ese caso hay dos posibilidades: que se haya llegado a un extremo abierto o a un extremo cerrado que pertenece a un nodo de afluencia $\{v_{i-1},v_j,v_i\}$. En el primer caso falta completar con la ecuación

$$x=(A_{1,3,i}-A_{1,2,i}\Delta Q_{i+1})/A_{1,1,i}$$

donde $x=\Delta Z_i$ o $x=\Delta Q_i$, según la condición de contorno aguas arriba sea Q o Z conocido, respectivamente.

En el segundo caso, la ecuación (43.i-1) permite calcular ΔQ_i. Como $\Delta Z_i=\Delta Z_{i-1}=\Delta Z_j$, usando (43.i-2) se puede ascender por ese tramo también. Una vez recorrido "aguas arriba" (o sea, en el sentido del árbol), todos los tramos que sean necesarios, se llegará a v_j, y se repite el procedimiento. Es decir, dada una red fluvial arborescente, discretizada en la forma indicada, con n puntos de discretización, si se excluye la posibilidad de división por cero, se puede resolver el sistema linealizado de las ecuaciones de Saint-Venant con un método implícito de "caja de cuatro puntos" usando esencialmente la misma memoria de computadora y número de operaciones que para un tramo simple que conste de n puntos.

Exhibiremos ahora un ejemplo sencillo para aclarar el procedimiento utilizado. Sea un tramo (x_1,x_2,x_3) que desemboca en un cauce discretizado como $(x_4,x_5,x_6,x_7,x_8,x_9)$ entre los puntos x_6 y x_7, o sea los tres tramos son: (x_1,x_2,x_3), (x_4,x_5,x_6) y (x_7,x_8,x_9), siendo (x_1,x_2,x_3) el tramo afluente, (x_4,x_5,x_6) el tramo simple del cauce principal aguas arriba de la desembocadura del afluente y (x_7,x_8,x_9) el tramo del cauce principal aguas abajo de la desembocadura del afluente (ver figura 9).

Entonces el sistema lineal en cada paso de tiempo quedará (suponiendo las siguientes condiciones de contorno: caudal conocido en los dos extremos aguas arriba - x_1 y x_4 - y nivel conocido en el extremo aguas abajo x_9):

$$\begin{aligned}
A_{21}\Delta Z_1+A_{31}\Delta Q_2+A_{41}\Delta Z_2 &= A_{51}-A_{11}\Delta Q_1\\
B_{21}\Delta Z_1+B_{31}\Delta Q_2+B_{41}\Delta Z_2 &= B_{51}-B_{11}\Delta Q_1\\
A_{12}\Delta Q_2+A_{22}\Delta Z_2+A_{32}\Delta Q_3+A_{42}\Delta Z_3 &= A_{52}\\
B_{12}\Delta Q_2+B_{22}\Delta Z_2+B_{32}\Delta Q_3+B_{42}\Delta Z_3 &= B_{52}
\end{aligned}$$

que se tridiagonaliza, como ya hemos visto, como

$$\begin{aligned}
C_{21}\Delta Z_1+C_{31}\Delta Q_2 &= C_{41}-C_{11}\Delta Q_1\\
D_{11}\Delta Z_1+D_{21}\Delta Q_2+D_{31}\Delta Z_2 &= D_{41}\\
C_{12}\Delta Q_2+C_{22}\Delta Z_2+D_{32}\Delta Q_3 &= C_{42}\\
D_{12}\Delta Z_2+D_{22}\Delta Q_3+D_{32}\Delta Z_3 &= D_{42}
\end{aligned}$$

y análogamente, el bloque compuesto por el tramo (x_4,x_5,x_6) se tridiagonaliza

$$\begin{aligned}
C_{24}\Delta Z_4+C_{34}\Delta Q_5 &= C_{44}-C_{14}\Delta Q_4\\
D_{14}\Delta Z_4+D_{24}\Delta Q_5+D_{34}\Delta Z_5 &= D_{44}\\
C_{15}\Delta Q_5+C_{25}\Delta Z_5+C_{35}\Delta Q_6 &= C_{45}\\
D_{15}\Delta Z_5+D_{25}\Delta Q_6+D_{35}\Delta Z_6 &= D_{45}
\end{aligned}$$

y el bloque compuesto por el tramo (x_7,x_8,x_9) se tridiagonaliza

$$\begin{aligned}
C_{17}\Delta Q_7+C_{27}\Delta Z_7+C_{37}\Delta Q_8 &= C_{47}\\
D_{17}\Delta Z_7+D_{27}\Delta Q_8+D_{37}\Delta Z_8 &= D_{47}\\
C_{18}\Delta Q_8+C_{28}\Delta Z_8+C_{38}\Delta Q_9 &= C_{48}\\
D_{18}\Delta Z_8+D_{28}\Delta Q_9 &= D_{48}-D_{38}\Delta Z_8
\end{aligned}$$

Tomemos el primer bloque. Restando recurrentemente cada fila, multiplicada convenientemente, de la siguiente, para que se anule el único término de la derecha de la diagonal principal, se obtiene, con la notación de (42) y (43)

$$\begin{aligned}
A_{111}\Delta Z_1+A_{121}\Delta Q_2 &= A_{131}\\
A_{211}\Delta Q_2+A_{221}\Delta Z_2 &= A_{231}\\
A_{112}\Delta Z_2+A_{122}\Delta Q_3 &= A_{132}\\
A_{212}\Delta Q_3+A_{222}\Delta Z_3 &= A_{232}
\end{aligned}$$

y análogamente para el bloque de (x_4,x_5,x_6) se obtiene

$$\begin{aligned}
A_{114}\Delta Z_4+A_{124}\Delta Q_5 &= A_{134}\\
A_{214}\Delta Q_5+A_{224}\Delta Z_5 &= A_{234}\\
A_{115}\Delta Z_5+A_{125}\Delta Q_6 &= A_{135}\\
A_{215}\Delta Q_6+A_{225}\Delta Z_6 &= A_{235}
\end{aligned}$$

Pero dado que $\Delta Q_3+\Delta Q_6=\Delta Q_7$ y $\Delta Z_3=\Delta Z_6=\Delta Z_7$, por las condiciones de compatibilidad de Stoker, se tendrá

$$(\Delta Q_3+\Delta Q_6)+(A_{222}\Delta Z_3/A_{212} + A_{225}\Delta Z_3/A_{215})=(A_{232}/A_{212}+A_{235}/A_{215})$$

es decir

$$\Delta Q_7+A_{226}\Delta Z_7=A_{236} \qquad \text{con}$$

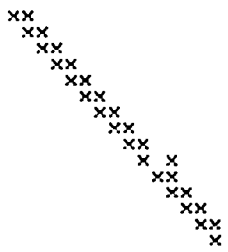
$$\begin{aligned}
A_{226}&=A_{222}/A_{212}+A_{225}/A_{215}\\
A_{236}&=A_{232}/A_{212}+A_{235}/A_{215}
\end{aligned}$$

y el tercer bloque se escribe

$$\begin{aligned}
\Delta Q_7+A_{226}\Delta Z_7 &= A_{236}\\
C_{17}\Delta Q_7+C_{27}\Delta Z_7+C_{37}\Delta Q_8 &= C_{47}\\
D_{17}\Delta Z_7+D_{27}\Delta Q_8+D_{37}\Delta Z_8 &= D_{47}\\
C_{18}\Delta Q_8+C_{28}\Delta Z_8+C_{38}\Delta Q_9 &= C_{48}\\
D_{18}\Delta Z_8+D_{28}\Delta Q_9 &= D_{48}-D_{38}\Delta Z_9
\end{aligned}$$

de donde se puede seguir triangulando a partir de ΔQ_7, que está sobre la diagonal principal, lo que permite ir anulando el término a la derecha de la diagonal principal que hay en cada una de las filas subsiguientes.

La matriz final obtenida no es exactamente bidiagonal (en el sentido de tener solo valores sobre la diagonal principal y la diagonal inmediata superior) pero su estructura es muy simple:



y no tiene más de dos elementos no nulos en cada fila.

Para n puntos de discretización, con m tramos simples, t nodos de afluencia y s extremos aguas arriba (siempre hay un solo extremo aguas abajo, la raíz del árbol) el número total de incógnitas es 2n-(s+1) que es, por supuesto, el orden de la matriz. Sin embargo, desde el punto de vista de la programación del modelo, es más sencillo (y no influye sustancialmente en la memoria) tomar como incógnitas directamente 2n. Con el mismo criterio, la memoria necesaria para resolver **Ax=b** en cada paso de tiempo es la correspondiente a los $A_{s,l,i}$, s=1,2; l=1,...3; i todos los puntos de discretización Además se necesitan los diez coeficientes A_l, B_l l=1,...,5 *sin distinción de punto de discretización* pues se usan para calcular los $C_{l,i}$ y $D_{l,i}$ nada más, y esos $C_{l,i}$, $D_{l,i}$ se pueden almacenar en los propios A_l, B_l, que no se vuelven a usar en su forma original. Se requieren entonces 6n+10 palabras de computadora, o sea aproximadamente 6n.

En cuanto al número de operaciones, será a lo sumo (sin contar las necesarias para calcular los coeficientes de cada fila de la matriz):

1) 1 división y 7 sumas y multiplicaciones para cada punto de discretización v_i para tridiagonalizar la matriz si v_i no es extremo aguas abajo de ningún tramo simple, o sea 8(n-m), donde m es el número de tramos simples.

2) A lo sumo 4 sumas y multiplicaciones y 2 divisiones para cada uno de esos puntos para triangular la matriz, o sea $6(n-m) = 6n-6m$.

3) 4 divisiones más dos sumas para cada nodo de afluencia para completar la triangularización, o sea 4t (despreciamos las sumas). Si consideramos que la división demanda mucho más tiempo que la multiplicación, podemos ,decir en lugar de la cifra anterior, dos divisiones, más cuatro multiplicaciones, más dos sumas.

4) 1 división y una suma y multiplicación por variable en el barrido ascendente, o sea a lo sumo $2n-s-1$ sumas y multiplicaciones y $2n-s-1$ divisiones, es decir, $4n-2s-2$ operaciones.

Esto hace un total del orden de $18n + 4t$ operaciones, o, separando divisiones, $5n + 2t$ divisiones y $13n + 4t$ sumas y divisiones.

Tanto las estimaciones para la memoria como las estimaciones para las operaciones son similares a las obtenidas en la sección 7 para tramos simples de la misma cantidad de puntos de discretización.

11. COMPARACION DE RESULTADOS ANALITICOS CON NUMERICOS

Interesa ensayar el esquema de Preissmann para una cuenca fluvial con estructura arborescente lo suficientemente simple como para tener para ella soluciones teóricas explícitas, y poder entonces comparar los resultados teóricos con los resultados obtenidos mediante el método numérico. Demostraremos entonces en primer lugar un teorema que asegura que, en régimen subcrítico, el sistema de red fluvial arborescente con coeficientes constantes tiene solución dando una condición de contorno para cada extremo abierto. Este resultado nos garantiza en cierto sentido el uso de las condiciones de contorno para redes fluviales con ecuaciones casilineales, por generalización.

TEOREMA: Sea una red fluvial arborescente. Entonces el problema

$$\frac{\partial w}{\partial t} + A\frac{\partial w}{\partial x} = 0 \qquad (49)$$

siendo $w = (u,h)^t$,

$$A = \begin{bmatrix} U & g \\ H & U \end{bmatrix}$$

matriz de 2 x 2 con coeficientes constantes y con autovalores reales $\lambda_1 = U-\sqrt{gH} < 0 < \lambda_2 = U+\sqrt{gH}$ generada "congelando" los coeficientes de la matriz de las ecuaciones (3) de aguas poco profundas; con condiciones iniciales

$$w_o(x) = w(x,0) \qquad (50)$$

con condiciones de contorno

$$b_L . w = b_{1,L}\, u(x_L,t) + b_{2,L}\, h(x_L,t) = f_L(t) \qquad (51)$$

en cada uno de los extremos abiertos x_L de la red; y con condiciones de compatibilidad

$$\begin{aligned} u(x_i,t) + u(x_j,t) &= u(x_k,t) \\ h(x_i,t) = h(x_j,t) &= h(x_k,t) \end{aligned} \qquad (52)$$

en cada uno de los nodos de confluencia (x_i, x_j, x_k) tiene solución única dos veces derivable en ambas variables en cada tramo simple que constituye la red fluvial, siempre que w_o y f_L (para todo L) sean dos veces derivables, y $b_{1,L}$ y $b_{2,L}$ sean variables binarias 0/1 tales que su suma de 1. La solución existe para todo $t > 0$.

DEMOSTRACION:

Sea $A=S^{-1}\Lambda S$, con Λ matriz diagonal de los autovalores y $S=(s_{i,j})$ matriz de transformación de A en Λ. Sea $y=(v,z)$ tal que $w=Sy$. El problema (49), (50), (51), (52) equivale entonces al problema de encontrar la solución de

$$\begin{aligned} \frac{\partial v}{\partial t} + \lambda_1 \frac{\partial v}{\partial x} &= 0 \\ \frac{\partial z}{\partial t} + \lambda_2 \frac{\partial z}{\partial x} &= 0 \end{aligned} \qquad (53)$$

con condiciones iniciales

$$y_o(x) = y(x,0) = S^{-1} w_o(x) \quad (54)$$

condiciones de contorno

$$b_L . Sy(x_L,t) = c_L . y(x_L,t) = c_{1,L} v(x_L,t) + c_{2,L} z(x_L,t) = f_L(t) \quad (55)$$

y condiciones de compatibilidad

$$\begin{aligned} &s_{11}v(x_i,t)+s_{12}z(x_i,t)+s_{11}v(x_j,t)+s_{12}z(x_j,t)-s_{11}v(x_k,t)-s_{12}z(x_k,t) \\ &\qquad\qquad =0 \\ &s_{21}v(x_i t)+s_{22}z(x_i,t)-s_{21}v(x_j,t)-s_{22}z(x_j,t) \qquad =0 \\ &s_{21}v(x_i,t)+s_{22}z(x_i,t) \qquad -s_{21}v(x_k,t)-s_{22}z(x_k,t) \\ &\qquad\qquad =0 \quad (56) \end{aligned}$$

Para cada tramo simple m de extremos (x_{m_1}, x_{m_2}) con $x_{m_1} < x_{m_2}$ sea $T_m > 0$ tal que $x_{m_1} - \lambda_1 T_m < x_{m_2}$ y tal que $x_{m_1} < x_{m_2} - \lambda_2 T_m$, es decir, las características que llegan a los extremos se originan en el mismo tramo en el instante inicial. Sea $0 < T = \min_m T_m$, y sea $0 < t < T$.

Tomemos un punto extremo aguas arriba x_L. Se tiene

$$c_{1,L}v(x_L,t) + c_{2,L}z(x_L,t) = f_L(t)$$

Supongamos que $c_{2,L} \neq 0$. Definimos $v(x_L,t) = v_o(x_L-\lambda_1 t)$, que está en el mismo tramo simple pues $t < T$, y por lo tanto $z(x_L,t)=(f_L(t)-c_{1,L}v(x_L,t))/c_{2,L}$. Sea ahora el extremo abierto aguas abajo x_B. Se tiene

$$c_{1,B}v(x_B,t) + c_{2,B}z(x_B,t) = f_B(t)$$

y definiendo

$$z(x_B,t) = z_o(x_B - \lambda_2 t)$$

obtenemos también el valor de $v(x_B,t)$, siempre que $c_{1,B} \neq 0$. Dejaremos para el final la demostración de que $c_{2,L} \neq 0$ si x_L es un extremo abierto aguas arriba y $c_{1,B} \neq 0$ si x_B es el extremo abierto aguas abajo.

Sea ahora un nodo de afluencia $\{x_i,x_j,x_k\}$, donde x_i y x_j son extremos aguas abajo de tramos simples m_i, m_j, afluyentes al tramo m_k cuyo extremo aguas arriba es x_k. Definimos $z(x_i,t) = z_o(x_i-\lambda_2 t)$, $z(x_j,t) = z_o(x_j-\lambda_2 t)$, $v(x_k,t) = v_o(x_k-\lambda_1 t)$, y, como $t < T$, $x_i-\lambda_2 t$, $x_j-\lambda_2 t$, y $x_k-\lambda_1 t$ pertenecen, respectivamente, a los tramos simples m_i, m_j, y m_k. Pasando los términos que contienen como factores a $z(x_i,t)$, $z(x_j,t)$ y $v(x_k,t)$ al lado derecho de las igualdades (56), se tienen definidas $v(x_i,t)$, $v(x_j,t)$ y $z(x_k,t)$ siempre que la matriz

$$D = \begin{bmatrix} s_{11} & s_{11} & -s_{12} \\ s_{21} & -s_{21} & 0 \\ s_{21} & 0 & -s_{22} \end{bmatrix}$$

no sea singular. Supondremos también por ahora que D no es singular.

Sea ahora x interior a un tramo simple. Entonces puede pasar lo siguiente:

a1) $x-\lambda_1 t$ pertenece al tramo simple m. En ese caso, se define $v(x,t)=v_o(x-\lambda_1 t)$.

a2) Existe t_1 tal que $0 < t_1 < t$ y tal que $x_B-x = \lambda_1(t_1-t)$, siendo x_B el punto extremo aguas abajo del tramo simple m. En ese caso $t_1=t_1(x,t) = (x_B-x)/\lambda_1 + t$, y se define $v(x,t) = v(x_B,t_1)$, o sea

$$v(x,t) = [\ f_B(t_1(x,t)) - c_{2B}z_o(x_B-\lambda_2 t_1(x,t))\ ] / c_{1B}$$

si el extremo es el extremo abierto aguas abajo y

$$\begin{aligned} v(x,t) = {} & r_{11}(-s_{12}z(x_B-\lambda_2 t_1(x,t))-s_{12}z(x_j-\lambda_2 t_1(x,t))+ \\ & s_{11}v(x_k-\lambda_1 t_1(x,t))) + \\ & r_{12}(-s_{22}z(x_B-\lambda_2 t_1(x,t))+s_{22}z(x_j-\lambda_2 t_1(x,t))) \\ & +r_{13}(-s_{22}z(x_B-\lambda_2 t_1(x,t))+s_{21}v(x_k-\lambda_1 t_1(x,t))) \end{aligned}$$

donde x_B pertenece, como extremo aguas abajo, al nodo de afluencia $\{x_B,x_j,x_k\}$ y los r_{1l} son los elementos de la primera fila de la matriz inversa de D. (Si el nodo de confluencia es $\{x_i,x_B,x_{i+1}\}$ la demostración es la misma pero con los elementos r_{21} , r_{22} y r_{23} de la fila 2 de D^{-1} y el correspondiente término de la derecha de la igualdad).

b1) $x-\lambda_2 t$ pertenece al tramo simple m. En ese caso, se define $z(x,t)=z_o(x-\lambda_2 t)$.

b2) Existe t_2 tal que $0 < t_2 < t$ y tal que $x_L-x = \lambda_2(t_2-t)$, siendo x_L el punto extremo aguas arriba del tramo simple m. En ese caso también $t_2=t_2(x,t)=(x_L-x)/\lambda_2+t$ y se define $z(x,t)$ en forma análoga a la usada en a).

Es inmediato que v y z así definidas cumplen las ecuaciones diferenciales (53) -pues t_1 y t_2 son funciones C^∞ de x y de t en un entorno conveniente-, las condiciones iniciales (54), las condiciones de contorno (55) y las condiciones de compatibilidad

(56), o sea el problema está resuelto en el intervalo $0 \le t \le T$. Pero tomando ahora como condiciones iniciales $y_o(x)=y(x,T)$, se puede extender la solución al intervalo temporal $T \le t \le 2T$, y así sucesivamente.

Sólo nos queda por demostrar que efectivamente $c_{2L} \neq 0$ si x_L es extremo abierto aguas arriba, $c_{1,B} \neq 0$ si x_B es el extremo abierto aguas abajo, y la matriz D tiene inversa.

La matriz S es

$$S = \begin{bmatrix} H & H \\ -\sqrt{gH} & \sqrt{gH} \end{bmatrix}$$

$b_L . S_L = (b_{1L}H - b_{2L}\sqrt{gH} \ , b_{1L}H + b_{2L}\sqrt{gH})$ y ninguna de sus dos componentes es 0, pues las b_{iL} son variables binarias una de las cuales vale 1 y la otra 0.

Finalmente, la matriz D será

$$D = \begin{bmatrix} H & H & -H \\ -\sqrt{gH} & \sqrt{gH} & 0 \\ -\sqrt{gH} & 0 & -\sqrt{gH} \end{bmatrix}$$

cuyo determinante es $-3gH^2 \neq 0$. Con esto queda demostrado el teorema. ■

Obsérvese que la demostración es en realidad más general: basta que A, S y b sean tales que:

i) A tenga autovalores reales $\lambda_1 < 0 < \lambda_2$;

ii) $b_{1B}s_{11} + b_{2B}s_{21} \neq 0$ si x_B es extremo abierto aguas abajo;

iii) $b_{1L}s_{12} + b_{2L}s_{22} \neq 0$ si x_L es extremo abierto aguas arriba;

iv)

$$0 \neq \det \begin{bmatrix} s_{11} & s_{11} & -s_{12} \\ s_{21} & -s_{21} & 0 \\ s_{21} & 0 & -s_{22} \end{bmatrix}$$

Esto nos servirá para el ejemplo que se dará a continuación, en el que, por razones de simplicidad, usaremos A dada por

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad (57)$$

S estará dada entonces por

$$S = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

y se comprueba inmediatamente que se cumplen las condiciones i) a iv) arriba enunciadas.

Se utiliza entonces, para una cuenca fluvial, el sistema simplificado cuya matriz A es la dada por (57). El motivo de usar este sistema sencillo es que para él, cuando $\theta=1/2$ y $\Delta t=\Delta x$, *la solución analítica coincide con la solución numérica dada por el esquema de Preissmann*, lo cual permite validar los resultados. Esto es inmediato si se observa que cuando el dominio de dependencia de un punto (x,t) está contenido en el tramo simple al cual pertenece x vale, para $(h,u)^t = Sw = S(w_1,w_2)^t$

$$h(x,t)= w_1(x,t) + w_2(x,t)$$

$$u(x,t)= -w_1(x,t) + w_2(x,t)$$

con $w_1(x,t)=w_1(x+t,0)$ y $w_2(x,t)=w_2(x-t,0)$. De aquí se deduce que que la solución analítica coincide con la numérica para w_1, w_2 (siempre que $\Theta=1/2$ y $\Delta x=\Delta t$), y por consiguiente para h y u.

La red a usar será la indicada en la figura 10, siendo los valores x_i los indicados en el cuadro 1, con las siguientes condiciones iniciales y de contorno:

a) Condiciones iniciales:

Tramos 1 a 4 y 13 a 19:

$u(x,0)=10 + \cos x$
$h(x,0)= 6 + \text{sen } x$

Tramos 5 a 8 y 9 a 12:

$u(x,0)=5 + 0,5 \cos x$
$h(x,0)=6 + 0,5 \text{ sen } x$

Tramo 20 a 35:

$u(x,0)= 20 + 2 \cos x$
$h(x,0)= 6 + 2 \text{ sen } x$

b) Condiciones de contorno:

$u(x_1,t)=10$
$u(x_5,t)= 5$
$u(x_9,t)= 5$
$h(x_{35},t)=6+2(\cos t+\text{sen } t)$

Cuadro 1

Coordenadas espaciales de los puntos de discretización

Nodo	Coordenada espacial
1	3Π/2
2	5Π/3
3	11Π/6
4	2Π
5	Π/2
6	2Π/3
7	5Π/6
8	Π
9	Π/2
10	2Π/3
11	5Π/6
12	Π
13	Π
14	7Π/6
15	4Π/3
16	3Π/2
17	5Π/3
18	11Π/6
19	2Π
20	2Π
21	13Π/6
22	7Π/3
23	5Π/2
24	8Π/3
25	17Π/6
26	3Π
27	19Π/6
28	10Π/3
29	7Π/2
30	11Π/3
31	23Π/6
32	4Π
33	25Π/6
34	13Π/3
35	9Π/2

La solución exacta del sistema diferencial (57) con estas condiciones iniciales y de contorno es:

En los tramos 1 a 4 y 13 a 19:

$$u(x,t)=10 + (\cos t - \operatorname{sen} t) \cos x$$
$$h(x,t)= 6 + (\cos t + \operatorname{sen} t) \operatorname{sen} x$$

En los tramos 5 a 8 y 9 a 12:

$$u(x,t)= 5 + 0,5 (\cos t - \operatorname{sen} t) \cos x$$
$$h(x,t)= 6 + 0,5 (\cos t + \operatorname{sen} t) \operatorname{sen} x$$

En el tramo 20 a 35:

$$u(x,t)=20 + 2 (\cos t - \operatorname{sen} t) \cos x$$
$$h(x,t)= 6 + 2 (\cos t + \operatorname{sen} t) \operatorname{sen} x$$

La solución numérica según el esquema de Preissmann para $\theta=1/2$ y $\Delta t=\Delta x$ debería coincidir con la solución analítica, como ya comentamos,y esto es exactamente lo que sucede. En el cuadro 2 se muestra la solución según el modelo arborescente, que coincide con la analítica, como se comprueba inmediatamente:

Cuadro 2

Altura (h) y velocidad (v) en distintos puntos de discretización para corridas del modelo lineal simplificado con $\theta=1/2$ y $\Delta t=\Delta x=\Pi/6$

	Nodo 6		Nodo 15		Nodo 27		Nodo 34	
t	h	v	h	v	h	v	h	v
0	6,433	4,750	5,134	9,500	5,000	18,268	7,732	21,000
$\Pi/2$	6,433	5,250	5,134	10,500	5,000	21,732	7,732	19,000
Π	5,567	5,250	6,866	10,500	7,000	21,732	4,268	19,000
$3\Pi/2$	5,567	4,750	6,866	9,500	7,000	18,268	4,268	21,000
2Π	6,433	4,750	5,134	9,500	5,000	18,268	7,732	21,000

Nota: Se indican las soluciones cada 3 intervalos de tiempo.

Los coeficientes $A_{1i}, A_{2i}, A_{3i}, A_{4i}, A_{5i}, B_{1i}, B_{2i}, B_{3i}, B_{4i}, B_{5i}$ serán, para el esquema simplificado

$A_{1i}=1$ $\quad A_{2i}=-2r\theta$ $\quad A_{3i}=1$ $\quad A_{4i}=2r\theta$ $\quad A_{5i}= 2r(h_i-h_{i+1})$

$B_{1i}=-2r\theta$ $\quad B_{2i}=1$ $\quad B_{3i}=2r\theta$ $\quad B_{4i}=1$ $\quad B_{5i}= 2r(u_i-u_{i-1})$

como se ve rápidamente.

La verificación de que el esquema es estable solamente si $\theta \geq 1/2$ es inmediata: corriendo el mismo modelo con los mismos intervalos espacial y temporal, pero con $\theta=1/4$, el mismo se inestabiliza rápidamente: al intentar pasar del tiempo Π al tiempo $\Pi+\Pi/6$ ya se obtiene una altura negativa y se interrumpe el proceso. En el cuadro 3 se observan los resultados obtenidos para los tiempos $\Pi/2$ y Π en los mismos puntos que en el cuadro 2:

Cuadro 3

Verificación de la inestabilidad del sistema para $\theta<1/2$

	Nodo 6		Nodo 15		Nodo 27		Nodo 34	
t	h	v	h	v	h	v	h	v
$\Pi/2$	6,541	5,297	4,919	10,594	4,732	22,078	7,726	19.248
Π	5,389	5,389	7,217	10,784	12,927	17,180	3,636	18,363

En el paso de tiempo siguiente se hace negativa la altura en el punto 23 y el programa se interrumpe.

Interesa, para analizar la atenuación y dispersión de la onda sinusoidal generada, observar el comportamiento de la simulación con distintos valores de $\theta \geq 1/2$ para $\Delta t \neq \Delta x$. Se analizarán ahora resultados para distintos valores de θ y de $r=\Delta t/\Delta x$. Dado que el problema que analizamos es el de una red arborescente, para mayor claridad en los gráficos observaremos la onda teórica y la onda numérica en el instante $t=2\Pi$, en que se cumple un período de cálculo, para distintos casos, pero en el "tramo principal" del modelo, o sea, en este caso, en el tramo entre los puntos de

discretización 20 y 35. No se efectuará ningún análisis teórico de los problemas de atenuación y dispersión, pues han sido estudiados, en particular por Cunge; interesa solamente constatar que esos fenómenos se repiten en el caso de una red arborescente.

a) *Distintos valores de r para θ=1/2.*

El cuadro 4 compara alturas teóricas (o, lo que es lo mismo, con r=1) en el instante t=2Π en el tramo simple entre los puntos 20 y 35 con alturas calculadas con r=0,5 y 1,5 (o sea, respectivamente, con Δt=Π/12 y Π/4). La figura 11 grafica esos resultados.

Cuadro 4

Comparación de alturas teóricos del modelo simplificado con alturas con r=1/2 y 3/2 para θ=1/2 en el instante t=2Π para los puntos de discretización 20 a 35

Punto	20	21	22	23	24	25	26	27	28
r=1/2	5,998	7,129	7,876	8,181	7,829	7,102	6,020	5,017	4,216
Teórica	6,000	7,000	7,732	8,000	7,732	7,000	6,000	5,000	4,268
r=3/2	5,999	6,815	7,409	7,618	7,379	6,747	5,887	5,037	4,438

Punto	29	30	31	32	33	34	35
r=1/2	3,950	4,156	4,890	5,833	6,887	7,596	8,000
Teórica	4,000	4,268	5,000	6,000	7,000	7,732	8,000
r=3/2	4,239	4,484	5,297	6,280	7,149	7,848	8,000

Se observa que la onda se atenúa algo para r=3/2 y se empunta para r=1/2.

b) *Distintos valores de θ con r=1.*

El cuadro 5 compara alturas teóricas (θ=1/2) con alturas calculadas para θ=2/3 y 1 en los mismos puntos de discretización que antes, con r=1. La figura 12 grafica esos resultados.

Cuadro 5

Comparación de alturas teóricas del modelo simplificado con alturas con $\theta=2/3$ y 1 con $r=1$ en el instante $t=2\Pi$ para los puntos de discretización 20 a 35

Punto	20	21	22	23	24	25	26	27	28
Teóricas	6,000	7,000	7,732	8,000	7,732	7,000	6,000	5,000	4,268
$\theta=2/3$	6,005	6,577	7,007	7,186	7,068	6,670	6,070	5,396	4,800
$\theta=1$	6,040	6,182	6,285	6,314	6,247	6,076	5,814	5,492	5,163

Punto	29	30	31	32	33	34	35
Teóricas	4,000	4,268	5,000	6,000	7,000	7,732	8,000
$\theta=2/3$	4,434	4,414	4,783	5,494	6,409	7,322	8,000
$\theta=1$	4,900	4,788	4,915	5,338	6,061	7,007	8,000

Se puede observar en la figura 12 que se produce el mismo fenómeno que para tramos simples: la onda se atenúa, y la atenuación es mayor con θ mayor.

Análogamente, comparando la curva teórica con la calculada para $r=1$ y $\theta=2/3$ y 1, se observa similar atenuación, como puede verse en el cuadro 6 y en la figura 13:

Cuadro 6

Comparación entre alturas teóricas y calculadas para $\theta=2/3$ y $\theta=1$ ($r=1$) para tiempos entre 0 y 2Π en puntos de discretización 20 y 27

	Punto 20			Punto 27		
Tiempo	*Teórica*	$\theta=2/3$	$\theta=1$	*Teórica*	$\theta=2/3$	$\theta=1$
0	6	6	6	5,000	5,000	5,000
$\Pi/6$	6	6	6	4,634	4,695	4,807
$\Pi/3$	6	6	6	4,634	4,748	4,924
$\Pi/2$	6	6	6	5,000	5,115	5,257
$2\Pi/3$	6	6	6	5,634	5,677	5,691
$5\Pi/6$	6	6	6	6,366	6,274	6,116
Π	6	6	6	7,000	6,756	6,450
$7\Pi/6$	6	6	6,002	7,366	7,019	6,641
$4\Pi/3$	6	6	6,004	7,366	7,027	6,666
$3\Pi/2$	6	6	6,009	7,000	6,802	6,524
$5\Pi/3$	6	6	6,017	6,366	6,396	6,243
$11\Pi/6$	6	6,002	6,028	5,634	5,893	5,874
2Π	6	6,005	6,040	5,000	5,396	5,492

Se incluyó en el cuadro el punto 20, que pertenece a un nodo de afluencia, que tiene un ajuste excelente (dado que es un punto de valor constante, la atenuación allí colabora en mejorar el ajuste). La atenuación de la onda crece con θ, como en el caso unidimensional estudiado por Cunge [1966], y como ya vimos.

No se insistirá más sobre el método de Preissmann aplicado a redes arborescentes con el esquema lineal simplificado, dado que sólo interesaba comprobar en algunos casos que se mantienen las características que Cunge analizó para tramos simples. Analizaremos ahora experimentos numéricos con el esquema propuesto y con el sistema casilineal no homogéneo (1), (2), (10) y (12).

12. EXPERIMENTOS NUMERICOS CON REDES FLUVIALES ARBORESCENTES

Se describirán a continuación una serie de experimentos numéricos desarrollados en una red fluvial compuesta por un río y un afluente, de características que se indican más abajo. La red fluvial se muestra en la figura 14, y sus características se detallan en el cuadro 7: las secciones transversales discretizadas son trapeciales, y se dan el ancho de fondo, en metros, y el ancho a otro nivel (10 metros por encima del nivel del fondo, en este caso). Si bien en este caso parece más intuitivo dar el "talud medio" (promedio de taludes a izquierda y derecha), el programa computacional está preparado para interpolar linealmente anchos, para secciones transversales irregulares, y por consiguiente se dan pares de puntos de tablas nivel/ancho. Para esos mismos niveles se dan los valores correspondientes de coeficientes de conducción, con lo cual se interpolan dichos coeficientes[3].

Cuadro 7.

Red fluvial arborescente de prueba del modelo hidrodinámico unidimensional casilineal

Punto de discr.	Coordenada espacial x (en Km)	Tabla de niveles (en m)	Anchos corresp. (en m)	Areas corresp. (en m^2)	Coef. de cond. corresp. (en m^3/s)
1	0	99	20	0	0
		109	50	350	20000
2	10	97	20	0	0
		107	50	350	20000
3	20	95	20	0	0
		105	50	350	20000
4	30	93	20	0	0
		103	50	350	20000
5	0	98	100	0	0
		108	120	1100	40000
6	10	97	100	0	0
		107	120	1100	40000
7	20	96	100	0	0
		106	120	1100	40000
8	30	95	100	0	0
		105	120	1100	40000
9	40	94	100	0	0
		104	120	1100	40000
10	50	93	100	0	0
		103	120	1100	40000
11	50	93	120	0	0
		103	140	1300	60000
12	60	92	120	0	0
		102	140	1300	60000
13	70	91	120	0	0
		101	140	1300	60000
14	80	90	120	0	0
		100	140	1300	60000

Nota: Las tablas de áreas se indican para clarificar las dimensiones del modelo; naturalmente son calculadas por el programa computacional.

El modelo, en resumen, consta de un afluente pequeño pero con el doble de pendiente de fondo que el cauce principal (2/10000) y un tramo principal de pendiente 1/10000 que aumenta su sección transversal después de la desembocadura del afluente (entre los puntos 10 y 11).

Uno de los problemas técnicos más difíciles en modelos fluviales reales es encontrar condiciones iniciales razonables. Lo ideal es comenzar con condiciones iniciales de fluido estacionario, en cuyo caso las ecuaciones de Saint-Venant (1) quedan (igualando a 0 las derivadas $\frac{\partial Z}{\partial t}$ y $\frac{\partial Q}{\partial t}$)

Q=constante

$$(gS-Q^2B/S^2)\frac{\partial Z}{\partial x} - Q^2S_x/S^2 + gQ|Q|S^2/D^2 = 0 \qquad (58)$$

ecuación diferencial ordinaria llamada ecuación de la curva de remanso. En lugar de implementar un módulo computacional con una solución numérica de dicha ecuación (que para el modelo arborescente debe resolverse teniendo en cuenta dicha arborescencia, o sea por ejemplo a partir de la condición inicial conocida $Z(x_L)$ retrocediendo hacia aguas arriba) se puede usar el programa general forzando una estabilización manteniendo constantes las condiciones de contorno durante un tiempo suficientemente prolongado. Si bien evitar el uso de un procedimiento para resolver numéricamente una ecuación diferencial ordinaria no es una ventaja importante, sí lo es el hecho de que de este modo se llega a un estado estacionario "lo más parecido posible" al de las condiciones iniciales originales. Esto es exactamente lo que se hizo en esta oportunidad, con las siguientes condiciones iniciales que *no* representan una situación estacionaria:

Cuadro 8

Condiciones iniciales para llegar a una situación estacionaria

Punto	Q (m^3/s)	Z (m)	V (m)
1	310	112,50	0,57
2	310	110,00	0,60
3	310	107,50	0,64
4	310	105,00	0,68
5	540	110,90	0,37
6	540	109,90	0,37
7	540	108,70	0,38
8	540	107,50	0,38
9	540	106,20	0,39
10	540	105,00	0,40
11	850	105,00	0,54
12	850	103,50	0,56
13	850	101,90	0,60
14	850	100,00	0,65

(V indica la velocidad).

Tomando como condición de contorno caudal en los dos extremos aguas arriba igual a 310 y 540 m^3/s, respectivamente, y nivel en el extremo aguas abajo igual a 100 m, constantes durante varios días, con intervalos temporales $\Delta t=1$ día, se llega a las siguientes condiciones estacionarias (el número de días necesario depende del θ usado: a mayor θ, más rápida la estabilización; por consiguiente, para este tipo de experimento conviene tomar $\theta=1$, que produce la máxima atenuación, lo que favorece la estabilización):

Cuadro 9

Condiciones iniciales estacionarias

Punto	Q (m^3/s)	Z (m)	V (m)
1	310	112,53	0,57
2	310	110,55	0,57
3	310	108,58	0,57
4	310	106,61	0,56
5	540	113,19	0,31
6	540	111,98	0,31
7	540	110,73	0,32
8	540	109,43	0,33
9	540	108,06	0,34
10	540	106,61	0,35
11	850	106,61	0,47
12	850	104,84	0,50
13	850	102,76	0,55
14	850	100,00	0,65

Es interesante observar la curva de "velocidad de convergencia" al régimen estacionario (es decir, número de días necesarios para llegar, en el modelo, a valores estacionarios de las variables) en función del valor del "parámetro de implicitud" θ usado. Se indican en el cuadro 10 y en la figura 15. Δt es 1 día.

Cuadro 10

Convergencia a régimen estacionario en función de θ

θ	Tiempo (en días) en el que se produce estabilización
0,5	inestable
0,667	inestable
0,75	oscila
0,80	36
0,85	19
0,90	12
0,95	10
1	10

Puede aquí observarse que para $\theta=2/3$ pueden producirse inestabilidades, pese a que la teoría para sistemas lineales indica que el esquema de Preissmann es estable para $\theta \geq 1/2$.

Con estas nuevas condiciones iniciales obtenidas se realizaron diferentes pruebas. En primer lugar, se realizó un experimento de consistencia, en el cual se repitió una corrida de computadora cambiando las condiciones de contorno, es decir, en la segunda corrida se reemplazaron las condiciones de contorno de la primera (caudal aguas arriba en los dos puntos extremos y nivel aguas abajo) por los valores calculados de la otra variable (nivel aguas arriba en los dos puntos extremos y caudal aguas abajo). Los resultados debían ser iguales, y lo fueron, como se ve en el cuadro 11. Las condiciones iniciales fueron las obtenidas en la corrida de estabilización, y las condiciones de contorno consistieron en disminuir linealmente a lo largo del tiempo los caudales de ingreso al afluente y aumentarlos en el cauce principal, y elevar el nivel aguas abajo. Las correspondientes condiciones de contorno se indican en el cuadro 12. Se usó $\theta=0,85$.

Cuadro 11

Estado del modelo en el día 18 para corridas con condiciones de contorno intercambiadas. $\Delta t=1$ día. $\theta=0.85$

Punto	Corrida 1		Corrida 2	
	Q (m^3/s)	Z (m)	Q (m^3/s)	Z (m)
1	250	114,72	250	114,72
2	248	113,84	248	113,84
3	246	113,10	246	113,10
4	243	112,46	243	112,46
5	1200	124,60	1199	124,60
6	1190	122,62	1189	122,62
7	1180	120,50	1180	120,50
8	1172	118,18	1171	118,18
9	1164	115,57	1164	115,57
10	1158	112,46	1158	112,46
11	1401	112,46	1401	112,46
12	1395	110,08	1395	110,08
13	1390	107,20	1391	107,20
14	1387	103,00	1387	103,00

Cuadro 12

Condiciones de contorno para las corridas 1 y 2 del Cuadro 11

Tiempo	Corrida 1			Corrida 2		
(días)	Q_1 (m^3/s)	Q_5 (m^3/s)	Z_{14} (m)	Z_1 (m)	Z_5 (m)	Q_{14} (m^3/s)
0	310	540	100	112,53	113,19	850
1				112,53	113,57	853
2				112,52	114,16	874
3				112,56	114,82	896
4				112,62	115,50	926
5				112,70	116,19	956
6				112,80	116,88	988
7				112,91	117,56	1020
8				113,03	118,24	1054
9				113,16	118,91	1087
10				113,30	119,57	1120
11				113,45	120,22	1153
12				113,61	120,87	1187
13				113,78	121,51	1220
14				113,96	122,14	1253
15				114,14	122,77	1286
16				114,33	123,38	1320
17				114,52	124,00	1353
18	250	1200	103	114,72	124,60	1387

Nota: Los valores correspondientes a la corrida 1 entre los instantes 1 y 18 se interpolaron linealmente respecto de los valores extremos.

Se puede observar que la coincidencia es excelente.

Por último, se llevaron a cabo varias corridas de computadora que responden a ciertas situaciones reales para las cuales el modelo es particularmente útil, que pasamos a enumerar:

3) El extremo abierto aguas arriba del afluente se "cierra", esto es, deja de aportar caudal, y el afluente recibe o entrega agua según aumente o disminuya la entrada de caudal al extremo aguas arriba del cauce principal. Las condiciones de contorno se indican en el cuadro 13. Se usó θ=0,75 y Δt=6 horas. (Se partió de las condiciones iniciales anteriores). Aguas abajo se considera nivel fijo debido, por ejemplo, a que existe un embalse regulador.

Cuadro 13

Condiciones de contorno para corrida 3 con afluente cerrado

Tiempo (horas)	Punto 1 Q (m^3/s)	Punto 5 Q (m^3/s)	Punto 14 Z (m)
0	310	540	100
24	0	850	100
48	0	1500	100
72	0	540	100
96	0	1800	100
120	0	540	100

Como se puede observar en el cuadro 14 y en la figura 16, se produce un movimiento oscilatorio del agua, que cambia el sentido del escurrimiento en el afluente convertido en un "desvío muerto".

Cuadro 14

Resultados en puntos 2, 4 y 14 para corrida 3

Tiempo	Punto 2		Punto 4		Punto 14	
(horas)	Q (m^3/s)	Z (m)	Q (m^3/s)	Z (m)	Q (m^3/s)	Z (m)
0	310	110,55	310	106,61	850	100
6	260	110,00	282	106,57	846	100
12	194	108,91	231	106,46	837	100
18	117	107,58	159	106,29	822	100
24	34	106,27	72	106,05	800	100
30	1	105,84	8	105,88	784	100
36	-5	106,00	-16	105,99	790	100
42	-9	106,38	-31	106,40	822	100
48	-14	106,98	-48	107,02	876	100
54	-15	107,65	-50	107,71	938	100
60	-12	108,20	-38	108,23	991	100
66	-4	108,45	-13	108,45	1019	100
72	5	108,33	17	108,33	1016	100
78	7	108,06	23	108,05	998	100
84	3	107,88	12	107,87	980	100
90	-4	107,98	-15	107,99	987	100
96	-13	108,45	-44	108,47	1022	100
102	-16	109,09	-55	109,15	1081	100
108	-14	109,68	-45	109,71	1133	100
114	-5	109,95	-15	109,95	1165	100
120	7	109,78	25	109,77	1156	100

4) Operación de un embalse más crecida extraordinaria: en este caso se supone una operación normal diaria de embalse en el punto de discretización 5, seguido de una crecida brusca que obliga a derivar caudal por el vertedero ubicado en el punto 1. Aguas abajo se considera una marea sinusoidal. Se usó θ=0,85 y Δt=3 horas. Las condiciones iniciales se dan en el cuadro 15, y fueron obtenidas estabilizando el régimen del río con su afluente en una corrida previa. Obsérvese el "efecto embalse" en el afluente: el nivel es

el mismo en los cuatro puntos de discretización, o sea no hay pendiente de niveles (de "pelo de agua", en la terminología de los ingenieros hidráulicos).

Cuadro 15

Condiciones iniciales de corrida 4 con simulación de embalse y vertedero

Punto	Caudal (m^3/s)	Nivel (m)
1	0	106,61
2	0	106,61
3	0	106,61
4	0	106,61
5	850	118,26
6	850	116,52
7	850	114,62
8	850	112,48
9	850	109,96
10	850	106,61
11	850	106,61
12	850	104,84
13	850	102,76
14	850	100,00

En el cuadro 16 se dan las condiciones de contorno de la corrida 4.

Cuadro 16

Condiciones de contorno para corrida 4 con simulación de embalse y vertedero

Tiempo (horas)	Punto 1 Q (m^3/s)	Punto 5 Q (m^3/s)	Punto 14 Z (m)
0	0	850	100,00
3	0	850	100,71
6	0	1500	101,00
9	0	1500	100,71
12	0	1500	100,00
15	0	1500	99,29
18	0	1500	99,00
21	0	850	99,29
24	0	850	100,00
27	0	850	100,71
30	0	1500	101,00
33	0	1500	100,71
36	0	1500	100,00
39	0	1500	99,29
42	0	1500	99,00
45	0	850	99,29
48	0	850	100,00
51	500	850	100,71
54	500	1500	101,00
57	500	1500	100,71
60	500	1500	100,00
63	500	1500	99,29
66	500	1500	99,00
69	500	850	99,29
72	500	850	100,00
75	0	850	100,71
78	0	1500	101,00
81	0	1500	100,71
84	0	1500	100,00
87	0	1500	99,29
90	0	1500	99,00
93	0	850	99,29
96	0	850	100,00

En el cuadro 17 se muestran los resultados de la corrida 4 en los puntos de discretización 4 y 12 (sobre la desembocadura del afluente y un poco más adelante en el cauce principal).

Cuadro 17

Resultados de la corrida 4 en los puntos 4 y 12

Tiempo	Punto 4		Punto 12	
(horas)	Z (m)	Q (m^3/s)	Z (m)	Q (m^3/s)
0	106,61	0	104,84	850
3	106,64	-5	104,91	833
6	106,72	-13	104,99	852
9	106,91	-27	105,12	865
12	107,14	-34	105,23	920
15	107,46	-47	105,43	945
18	107,79	-50	105,65	1001
21	108,10	-48	105,91	1017
24	108,32	-35	106,13	1043
27	108,47	-23	106,33	1032
30	108,60	-21	106,49	1048
33	108,75	-25	106,63	1059
36	108,94	-31	106,76	1092
39	109,17	-37	106,92	1113
42	109,41	-41	107,11	1146
45	109,61	-33	107,29	1159
48	109,72	-18	107,41	1168
51	110,46	310	107,90	1269
54	110,77	184	108,24	1278
57	111,27	359	108,65	1348
60	111,76	312	109,06	1389
63	112,27	392	109,52	1444
66	112,77	371	110,00	1485
69	113,16	419	110,40	1521
72	113,41	424	110,67	1538
75	113,31	358	110,69	1510
78	113,00	229	110,52	1461
81	112,65	143	110,25	1419
84	112,38	77	110,02	1390
87	112,20	40	109,86	1369
90	112,11	16	109,76	1363
93	112,01	19	109,66	1358
96	111,86	29	109,52	1347

Por último se realizaron pruebas para determinar el tiempo de cálculo, minimizando el acceso a archivos periféricos y de impresión, en la computadora personal TOSHIBA 3100 en la que se implementaron los modelos. Se obtuvo, para la red de 14 puntos antes descripta, un intervalo de aproximadamente 0,37 segundos por intervalo Δt de discretización.

Con esto queda concluido el análisis de las estructuras fluviales arborescentes. En la tercera parte de este trabajo analizaremos las estructuras fluviales deltaicas.

TERCERA PARTE

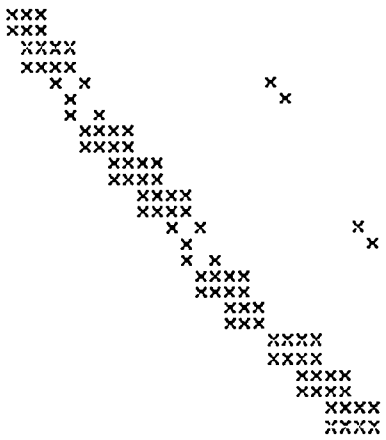
13. REDES FLUVIALES DELTAICAS

Una red fluvial compleja con estructura deltaica es una red en la cual además de afluentes puede haber efluentes, bifurcaciones que se cierran (por ejemplo, un tramo de río que se abre en dos por tener una isla en el medio, o directamente un delta, como el del río Paraná, si se lo quiere modelizar unidimensionalmente), etc. Por ejemplo, la red indicada en la figura 7. Los conceptos de tramos simples, tramos simples discretizados, etc., tienen una generalización obvia a estas redes, e introducimos en forma natural la definición de *nodo de efluencia* para una terna de puntos de discretización $\{v_i, v_j, v_k\}$ tal que ahora puede indicar que el tramo simple cuyo extremo aguas abajo es v_i se bifurca en dos tramos simples cuyos extremos aguas arriba son v_j y v_k . *Nodo de confluencia* indicará un nodo de afluencia o un nodo de efluencia.

En este caso también valen, por supuesto, las ecuaciones de compatibilidad de Stoker (12) en los nodos, y habrá eventualmente un número mayor que uno de extremos abiertos aguas arriba con condición de contorno dada, y también un número eventualmente mayor que uno de extremos abiertos aguas abajo.

La esencia del método descripto para redes arborescentes es que, gracias a la estructura arborescente de la red fluvial, se puede triangular completamente la matriz del sistema. Tal posibilidad no existe cuando se modelizan redes fluviales con estructura deltaica, es decir, donde hay bifurcaciones que se cierran (por ejemplo, un tramo de río que se abre en dos por tener una isla en el medio, o directamente un delta, como el del río Paraná, que se quiere modelizar unidimensionalmente). En este último caso, desde el punto de vista de teoría de grafos lo que se tiene es un grafo dirigido débilmente conexo sin circuitos, que se precisará en la sección 14. Las ecuaciones de compatibilidad entre los tramos son las ya enunciadas.

Consideremos como ejemplo la red dada por la figura 17. La matriz correspondiente será, sin simplificaciones,



y es obvio que la estructura de la matriz depende de la topología de la red. Se puede compactar ligeramente la matriz mediante manipulaciones algebraicas generales, por ejemplo reemplazando ΔQ_k por $\Delta Q_i+\Delta Q_j$ o por $\Delta Q_i-\Delta Q_j$ (según se trate de un nodo $\{v_i, v_j, v_k\}$ de afluencia o de efluencia) en todos los pares de ecuaciones que corresponda, con lo cual uno se ahorra una variable y una ecuación por nodo de confluencia, y análogamente reemplazando ΔZ_i y ΔZ_j por ΔZ_k, con lo cual se ahorran dos ecuaciones y dos incógnitas por nodo de confluencia. Lo que queda es una matriz de orden menor que 2n, si n es el número total de puntos de discretización; más concretamente, si se aplican los reemplazos mencionados, y la red tiene s extremos (aguas arriba o aguas abajo), y t nodos de confluencia, el orden de la matriz A será 2n-s-3t. La matriz A será rala: habrá filas con tres, cuatro y cinco elementos no nulos (las de tres elementos no nulos corresponden a pares de puntos de discretización en que un punto es un extremo aguas arriba o aguas abajo y el otro es un punto simple; las de cuatro elementos corresponden a pares de puntos correspondientes a puntos simples -o, eventualmente, un punto extremo y el otro perteneciente a un nodo de confluencia-, y la de cinco ecuaciones a un par de puntos con un punto simple y el otro perteneciente a un nodo de confluencia. En todos los casos todos los elementos de la fila, salvo a lo sumo uno, son contiguos).

Se presentan entonces dos caminos para tratar de resolver el sistema lineal de manera eficiente: por un lado, las matrices ralas tienen una característica que sugiere el uso de métodos

iterativos para la resolución de los correspondientes sistemas lineales: con métodos iterativos *se aprovecha la ralidad de la matriz* (dado que no se modifica) con lo cual se puede optimizar el uso de memoria, usando el mínimo de memoria compatible con un sistema de punteros de armado (y llamado) no demasiado complicado. Y también es obvio que los métodos iterativos, para estas matrices que no son simétricas, ni diagonalmente dominantes, son muy lentos. En particular, para los métodos de tipo Gauss-Seidel y Jacobi, nada asegura ni siquiera la convergencia, dado que, por ejemplo, no se puede asegurar la dominancia diagonal.

Por otro lado, se puede tratar de encontrar una estructura de la matriz A que permita una resolución por algún método directo en forma eficiente, es decir, con economía de memoria de computadora y de tiempo de cálculo. Este fue el punto de vista adoptado, teniendo en cuenta que la estructura de la red fluvial determina en principio la estructura de la matriz. Las próximas secciones que componen esta tercera parte del trabajo se dedicarán a analizar la estructura de la matriz A, la implementación de los correspondientes programas computacionales y las consecuentes experimentaciones numéricas.

14. DISCRETIZACION DE REDES FLUVIALES DELTAICAS

Tal como se hizo para redes fluviales arborescentes, se formalizarán los conceptos a usar en redes fluviales deltaicas complejas, a fin de emplear terminología precisa. Usaremos siempre la terminología de Christofides [1975] o Even [1979].

Definición: Una *discretización deltaica de una red fluvial* es un grafo dirigido finito G(V,E) tal que

a) G(V,E) es un grafo débilmente conexo sin circuitos.

b) Los vértices $v \in V$ son exactamente los puntos de discretización de la red fluvial.

c) Si $gr(v)$, $gr_E(v)$ y $gr_S(v)$ indican como antes, respectivamente, el grado, el grado de entrada y el grado de salida del vértice v, vale:

c1) $gr(v) \leq 3$.
c2) Si $gr_E(v) = 0$ entonces $gr_S(v) = 1$.
c3) Si $gr_S(v) = 0$ entonces $gr_E(v) = 1$.

d) Si $gr(v_k) = 3$ y $gr_E(v_k)=2$, entonces si v_i y v_j son vértices de origen de los arcos (v_i,v_k) y (v_j,v_k), vale que los puntos de discretización v_i, v_j y v_k forman un nodo de afluencia $\{v_i,v_j,v_k\}$ en que los tramos simples cuyos extremos aguas abajo son v_i y v_j afluyen al tramo simple cuyo extremo aguas arriba es v_k ; si por el contrario es $gr_S(v_k)=2$, entonces si v_i y v_j son vértices de destino de los arcos (v_k,v_i), (v_k,v_j) vale que los puntos de discretización v_k , v_i y v_j forman un nodo de efluencia $\{v_k,v_i,v_j\}$ en que el tramo simple cuyo extremo aguas abajo es v_k se bifurca en dos tramos simples cuyos extremos aguas arriba son v_i y v_j .

Recíprocamente, dada una confluencia sobre la cual hay tres puntos de discretización v_i , v_j y v_k , si v_k es extremo aguas arriba de un tramo formado por la afluencia de dos tramos cuyos extremos aguas abajo son v_i y v_j , valdrá que $gr_E(v_k)=2$ y (v_i,v_k) y (v_j,v_k) son arcos de G; y si v_k es extremo aguas abajo de un tramo que se bifurca en dos tramos cuyos extremos aguas arriba son v_i y v_j valdrá que $gr_S(v_k)=2$ y (v_k,v_i) y (v_k,v_j) son arcos de G.

e) Dados dos arcos consecutivos (v_i,v_j) y (v_j,v_k) tales que $gr_E(v_j) = gr_S(v_j) = 1$, los puntos de discretización correspondientes están sobre un cauce de red fluvial y, yendo desde aguas arriba hacia aguas abajo, primero está v_i , luego v_j y finalmente v_k , y no hay ningún punto de discretización entre ellos.

Es decir, se da al grafo la dirección "desde aguas arriba hacia aguas abajo". Los vértices con $gr_E(v)=0$ serán extremos abiertos aguas arriba de la red; los vértices con $gr_S(v)=0$ serán extremos abiertos aguas abajo de la red. El grafo G´ de dirección invertida es "desde aguas abajo hacia aguas arriba". Un camino del grafo G es un camino *descendente*; un camino del grafo G´ será un camino *ascendente*.

Como se quiere modelizar un sistema tipo delta, se supone que la dirección de los arcos de G es la del flujo "normal" (recordemos que en el caso de redes arborescentes habíamos elegido la dirección *opuesta* a la del flujo normal).

Por convención, tanto para nodos de afluencia como para nodos de efluencia $\{v_i,v_j,v_k\}$, v_i indicará un extremo aguas abajo de tramo; si el nodo es de efluencia indicará *el* extremo aguas abajo de tramo; v_k indicará un extremo aguas arriba de tramo: si el nodo es de afluencia, indicará *el* extremo aguas arriba; v_j indicará un extremo aguas abajo (si el nodo es de afluencia) o un extremo aguas arriba (si el nodo es de efluencia). Llamaremos extremo cerrado de tramo a un extremo de tramo que pertenece a un nodo de confluencia, es decir, que no es extremo abierto de la red.

TEOREMA: Sea una discretización deltaica de una red fluvial. Se puede entonces elegir una numeración de los puntos de discretización en la cual los nodos de afluencia y de efluencia son de la forma $\{v_i,v_j,v_{i+1}\}$ para enteros i, j, donde v_i es extremo aguas abajo de un tramo simple y v_j es extremo aguas arriba de un tramo simple.

N.B. : La demostración que sigue es algorítmica, y por consiguiente se puede programar la asignación de la numeración, cuya utilidad se verá al estructurar la matriz A en la forma deseada gracias a ella.

DEMOSTRACION:

Numeraremos los puntos de discretización del siguiente modo:

1) Se toma un extremo abierto aguas arriba . Este existe, pues a partir de un vértice cualquiera se puede seguir un camino ascendente y en algún punto se debe llegar a un extremo abierto, dado que el grafo es finito y sin circuitos y el camino ascendente no puede terminar en un nodo de confluencia: a éste pertenece al menos un extremo aguas abajo de tramo simple y entonces se podría seguir ascendiendo por el camino. El extremo abierto encontrado será el punto de discretización 1.

2) Se numeran los vértices consecutivamente siguiendo un camino descendente, a partir del primero, hasta que el camino termina (porque el grafo es finito) en un extremo abierto. No puede terminar en un vértice perteneciente a un nodo de confluencia, pues para cada nodo de confluencia hay uno (o dos) de los puntos de discretización que lo componen que es extremo aguas arriba de otro tramo simple, o sea el camino podría continuar. Y no puede terminar en un vertice perteneciente a un nodo de confluencia ya recorrido pues el grafo no tiene circuitos.

3) Si hay otro extremo abierto aguas arriba, se sigue numerando entonces el camino a partir de este extremo abierto. Ese camino termina en un extremo abierto, como antes, o en un vértice perteneciente a un nodo de confluencia del cual los otros dos tramos ya fueron recorridos. En este último caso, los otros dos vértices del nodo de confluencia estarán numerados como v_i y v_{i+1}, para algún i.

4) En cualquiera de ambos casos, se recomienza con un nuevo extremo abierto aguas arriba, si hay, y se procede de la misma manera. Cuando ya se recorrieron todos los caminos que comienzan en extremos abiertos aguas arriba, se busca el primer nodo de

efluencia del cual se ha recorrido el tramo que llega al mismo (o sea, del cual uno de los puntos de discretización es extremo aguas abajo) y uno de los tramos que salen del mismo, y del cual el otro tramo efluente es el que numeramos ahora. Se sigue con este camino hasta que termina en un extremo abierto o en un punto de discretización perteneciente a un nodo de confluencia cuyos otros dos puntos de discretización ya han sido recorridos.

5) Cuando se han recorrido todos los caminos que comienzan así (que puede no ser ninguno) se ha terminado la numeración de los vértices. En efecto, si hubiera un tramo que no hubiera sido recorrido, no podría éste comenzar en un extremo abierto (pues éstos han sido numerados ya todos) ni en un nodo de confluencia (pues ya han sido recorrido todos). Y no puede haber un tramo no relacionado con estos pues el grafo es débilmente conexo. ■

Esta será la discretización que usaremos de ahora en adelante. Observemos que por el algoritmo de la numeración, todo nodo de confluencia es de la forma $\{v_i, v_j, v_{i+1}\}$.

Cabe señalar que no hemos requerido para la demostración que el grafo sea planar, pese a que las redes fluviales se representan por grafos planares. O sea el teorema de numeración vale también para grafos no planares. Esto no es una abstracción teórica: si en lugar de ríos tenemos cañerías (que forman grafos claramente no planares), se aplica la misma teoría. Y las cañerías se pueden modelizar con las ecuaciones de Saint-Venant usando el artificio ideado por Cunge y Wegner [1964]: si se tiene una cañería a presión se la considera con superficie libre, tal como se indica en la figura 18, suponiendo una estrecha ranura ficticia extendida a lo largo de toda la longitud de la cañería, y de altura infinita a partir del extremo superior de aquélla.

Las redes fluviales que estudiaremos serán todas, al igual que para el caso arborescente, de régimen subcrítico, o sea será necesario dar una condición de contorno en cada extremo abierto de la red.

15. UN METODO DIRECTO PARA REDES FLUVIALES DELTAICAS

La matriz A estará constituida del siguiente modo:

a) Se numeran los vértices en la forma indicada en el teorema anterior, de tal modo que siempre que se produzca una confluencia, la numeración de los vértices del tramo aguas abajo de los dos afluentes sea consecutiva a la de los vértices de uno de los dos afluentes (el "tramo principal"). Análogamente, cuando se produzca una bifurcación, la numeración de una de los vértices de uno de los dos tramos resultantes debe ser consecutiva de la de los vértices del tramo aguas arriba de ambas.

b) A cada tramo simple corresponderá un bloque diagonal de la matriz A. Si el tramo es el k-simo tramo simple y es el correspondientea a los puntos de discretización j_1 a j_2, el bloque será una submatriz cuadrada de orden $2(j_2-j_1)$, y le corresponderán las $2(j_2-j_1)$ filas y columnas siguientes a las de los k-1 tramos simples anteriores, en la forma que se detallará más abajo.

c) La matriz A se completa con t columnas a la derecha de los bloques y t filas debajo de los bloques, cada una de las filas y las columnas de longitud igual a la suma de los órdenes de las matrices-bloques, en la forma que se detallará, y una submatriz cuadrada de txt llena de ceros en el extremo inferior derecho (completando las filas y las columnas recién mencionadas). t es el número de nodos de confluencia.

d) Todos los elementos no descriptos antes son nulos.

Veamos ahora el proceso constructivo preciso de las partes de la matriz A.

Usaremos el siguiente teorema:

TEOREMA: Sea n el número total de puntos de discretización de la red fluvial considerada, m el número total de tramos simples de la red, s el número total de extremos abiertos, aguas arriba o aguas abajo, y t el número total de nodos de confluencia. Entonces, si se identifican como una única incógnita los tres niveles (iguales, por las ecuaciones de compatibilidad de Stoker) correspondientes a

cada punto de confluencia, vale que el orden N de la matriz A (o sea, el número de incógnitas resultantes a partir de esas identificaciones), será:

$$N= 2(n-t)-s$$

Además, se puede estructurar la matriz A, el vector de incógnitas x, y el vector de datos b de la ecuación matricial Ax=b de modo que

$$A = \begin{vmatrix} A_1 & A_2 \\ A_3 & A_4 \end{vmatrix}$$

donde:

a) A_1 es una matriz cuadrada de orden N-t=2n-3t-s, con m bloques diagonales $A_{11},\ldots,A_{1m}$, y el resto de sus elementos nulos:

$$A_1 = \begin{vmatrix} A_{11} & & & & \\ & A_{12} & & & \\ & & A_{13} & & \\ & & & & \\ & & & & A_{1m} \end{vmatrix}$$

donde, a su vez, A_{1k} tendrá la forma

$$A_{1k}= \begin{bmatrix} \times\times\times & & & & \\ \times\times\times & & & & \\ & \times\times\times\times & & & \\ & \times\times\times\times & & & \\ & & \times\times\times\times & & \\ & & \times\times\times\times & & \\ & & & \times\times\times\times & \\ & & & \times\times\times\times & \\ & & & & \times\times\times \\ & & & & \times\times\times \end{bmatrix}$$

es decir, cada bloque A_{1k} tiene la misma estructura de la de la matriz que se origina en la discretización del tramo k como si fuera un único tramo unidimensional: dos primeras filas con los elementos correspondientes a las tres primeras columnas no nulos, seguidos por un par de filas con cuatro elementos no nulos correspondientes a las columnas 2 a 5, seguidos por pares de filas de cuatro elementos contiguos no nulos, que comienzan en la columna correspondiente al tercer elemento no nulo del par de filas anterior, hasta llegar al último par de filas, que tiene sus últimos tres elementos no nulos exclusivamente. El orden del bloque A_{1k} es $2(j_2-j_1)$, si el tramo simple k contiene los puntos de discretización $j_1, j_1+1, \ldots, j_2$.

b) A_2 es una matriz rectangular de N-t filas por t columnas, cuya estructura analizaremos más adelante.

c) A_3 es una matriz rectangular de t filas por N-t columnas, cuyos elementos no nulos valen 1 y -1, y cuya estructura general definiremos más adelante.

d) A_4 es una matriz nula de t filas por t columnas.

e) Las t incógnitas finales entre las que componen el vector x son las variaciones de nivel ΔZ_j de los puntos de confluencia.

f) Los t elementos inferiores del vector b son nulos.

N.B. : Observemos que de esta manera se garantiza que el orden y toda la estructura de la matriz A quedan determinados una vez que se numeraron los puntos de la red fluvial. Cualquiera sea la numeración de la red - siempre que se respete el algoritmo de la sección 14 - la estructura de la matriz es lá misma.

DEMOSTRACION:

La demostración de la primera parte del teorema es trivial: Para cada punto de discretización i hay dos valores, ΔZ_i y ΔQ_i. De esos las condiciones de contorno, que serán s, no son variables a calcular en cada intervalo de tiempo, y, por cada nodo de confluencia, dos de los tres niveles (pues son iguales al que sí se toma como variable).

Antes de pasar a la demostración de la segunda parte del teorema, observemos que la matriz **A** servirá para resolver un sistema de ecuaciones lineales **Ax=b**, donde las incógnitas $x=(x_1,\ldots,x_N)$ serán las siguientes:

Para el k-simo tramo simple, que comprende los puntos de discretización j_1 a j_2, x_{l+1}, donde l es el conjunto de variables ya asignadas a los tramos simples 1,...,k-1, será:

ΔQ_{j_1} si el punto de discretización j_1 es un extremo abierto aguas arriba con condición de contorno Z_{j_1} dada, o si el punto de discretización x_{j_1} pertenece a un nodo de confluencia (recibe dos afluentes o es el extremo aguas arriba de una de los dos tramos en que se bifurca un tramo).

ΔZ_{j_1} si el punto de discretización j_1 es un extremo abierto aguas arriba con condición de contorno Q_{j_1} dada.

$x_{l+2}, x_{l+4}, \ldots, x_{l+2(j_2-j_1)-2}$ serán, respectivamente, $\Delta Q_{j_1+1}, \Delta Q_{j_1+2}, \ldots, \Delta Q_{j_2-1}$.

$x_{l+3}, x_{l+5}, \ldots, x_{l+2(j_2-j_1)-1}$ serán, respectivamente,

$\Delta Z_{j_1+1}, \Delta Z_{j_1+2}, \ldots, \Delta Z_{j_2-1}$.

$x_{l+2(j_2-j_1)}$ será:

ΔZ_{j_2} si el punto extremo aguas abajo del k-simo tramo simple es un extremo abierto con condición de contorno dada Q_{j_2}.

ΔQ_{j_2} si el punto extremo aguas abajo es un extremo abierto con condición de contorno dada Z_{j_2}, o si el punto de discretización j_2 corresponde a un nodo de confluencia (es extremo aguas abajo de un tramo que afluye a otro o de un tramo que se bifurca).

Ahora bien, si hay s extremos abiertos, t confluencias y m tramos simples, cada uno con n_j puntos de discretización, de modo que

$$n= \sum_{j=1}^{m} n_j$$

el orden de cada submatriz A_{1j} es $2n_j-2$, pues en cada extremo se da una sola incógnita, o porque se trata de un extremo abierto o porque el nivel de un extremo perteneciente a un nodo de confluencia no corresponde al bloque. Por consiguiente, el orden de la submatriz A_1 es 2n-2m. El orden de A será, por otra parte, N-t, pues sólo los niveles correspondientes a los nodos de confluencia no son incógnitas en A_1. Entonces N-t=2n-2m, o sea N=2n-2m+t. Por otra parte N=2n-s-2t; en particular, s+2t=2m-t, es decir, s+3t=2m.

Las N-t primeras filas de A serán entonces las correspondientes a los pares de ecuaciones consecutivas entre los puntos (i,i+1), tales que (i,i+1) pertenecen al mismo tramo simple.

La submatriz A_2 de N-t filas por t columnas tendrá por columnas las correspondientes a los ΔZ_j consecutivamente, para Z_j el nivel común a los tres puntos de discretización de un nodo de confluencia $\{v_i, v_j, v_{i+1}\}$, según el orden dado por la numeración general de la discretización. Habrá elementos no nulos en las siguientes filas, para cada columna:

a) El par de filas consecutivas correspondiente a un intervalo (i+1,i+2) con i+1 punto extremo cerrado de la red (no abierto) aguas arriba, o sea perteneciente a un nodo de confluencia. Su coeficiente será A_{2i+1} o B_{2i+1}, según sea la primera o segunda ecuación del par.

b) El par de filas correspondiente a un intervalo (i-1,i) con i punto extremo aguas abajo. Su coeficiente será A_{4i-1} o B_{4i-1}, según sea la primera o segunda ecuación del par.

c) El par de filas consecutivas correspondientes a un intervalo (j,j+1) o (j-1,j), según sea v_j extremo aguas arriba o extremo aguas abajo, respectivamente, de un tramo simple, o sea según sea el nodo $\{v_i, v_j, v_{i+1}\}$ nodo de efluencia o de afluencia, respectivamente. Su coeficiente será A_{2j} (primera ecuación del par) y B_{2j} (segunda ecuación del par), si v_j es extremo aguas arriba, y A_{4j-1} y B_{4j-1} si v_j es extremo aguas abajo.

De estos 6 valores no nulos, 4 están en filas consecutivas (A_{4i-1}, B_{4i-1}, A_{2i}, B_{2i}).

Analicemos ahora la submatriz A_3 de t filas por N-t columnas. A ella corresponderán las ecuaciones de conservación de la masa líquida, o sea ecuaciones del tipo

$$\Delta Q_i + \Delta Q_j - \Delta Q_{i+1} = 0 \qquad \text{o} \qquad \Delta Q_i - \Delta Q_j - \Delta Q_{i+1} = 0$$

o sea los coeficientes distintos de 0 son 1 o -1. Además, esos coeficientes no nulos corresponden a columnas "extremas" de las submatrices $A_{1,k}$, es decir, a primeras o últimas columnas de $A_{1,k}$. Si el coeficiente es 1, será última columna (punto extremo aguas abajo perteneciente a nodo de confluencia), y si es -1 será primera columna (punto extremo aguas arriba perteneciente a nodo de confluencia). Por abuso de lenguaje, estamos denominando "columna de $A_{1,j}$ " a una columna de A algunos de cuyos elementos (los N-t superiores) pertenecen a $A_{1,j}$.

Finalmente, la cuarta submatriz A_4 estará compuesta exclusivamente de ceros, pues las filas de las ecuaciones de continuidad de los puntos de confluencia no tienen términos en los que figuren niveles. Y análogamente, los correspondientes términos independientes son nulos. ■

Se da ahora un ejemplo, correspondiente al esquema de la figura 17. Suponiendo condiciones de contorno en el extremo aguas arriba 1 caudal dado, y en el extremo aguas abajo 10 nivel dado, tendremos m=4, s=2, t=2, y Ax=b, con

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \qquad A_1 = \begin{bmatrix} A_{11} & & & \\ & A_{12} & & \\ & & A_{13} & \\ & & & A_{14} \end{bmatrix}$$

$$A_{11} = \begin{bmatrix} A_{21} & A_{31} & A_{41} & \\ B_{21} & B_{31} & B_{41} & \\ & A_{12} & A_{22} & A_{32} \\ & B_{12} & B_{22} & B_{32} \end{bmatrix}$$

$$A_{12} = \begin{bmatrix} A_{14} & A_{34} & A_{44} & & \\ B_{14} & B_{34} & B_{44} & & \\ & A_{15} & A_{25} & A_{35} & A_{45} \\ & B_{15} & B_{25} & B_{35} & B_{45} \\ & & & A_{16} & A_{26} & A_{36} \\ & & & B_{16} & B_{26} & B_{36} \end{bmatrix}$$

$$A_{13} = \begin{bmatrix} A_{18} & A_{38} & A_{48} & \\ B_{18} & B_{38} & B_{48} & \\ & A_{19} & A_{29} & A_{39} \\ & B_{19} & B_{29} & B_{39} \end{bmatrix}$$

$$A_{14} = \begin{bmatrix} A_{1,11} & A_{3,11} & A_{4,11} & & & \\ B_{1,11} & B_{3,11} & B_{4,11} & & & \\ & A_{1,12} & A_{2,12} & A_{3,12} & A_{4,12} & \\ & B_{1,12} & B_{2,12} & B_{3,12} & B_{4,12} & \\ & & & A_{1,13} & A_{2,13} & A_{3,13} \\ & & & B_{1,13} & B_{2,13} & B_{3,13} \end{bmatrix}$$

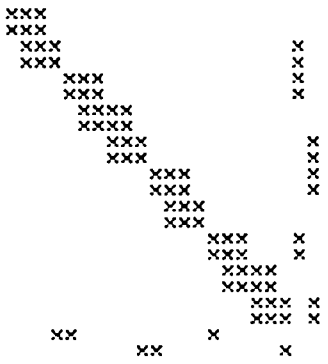
$$A_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ A_{42} & 0 \\ B_{42} & 0 \\ A_{24} & 0 \\ B_{24} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & A_{46} \\ 0 & B_{46} \\ 0 & A_{28} \\ 0 & B_{28} \\ 0 & 0 \\ 0 & 0 \\ A_{2,11} & 0 \\ B_{2,11} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & A_{4,13} \\ 0 & B_{4,13} \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4 = 0$$

$$b = \begin{bmatrix} A_{51}-A_{11}\Delta Q_1 \\ B_{51}-B_{11}\Delta Q_1 \\ A_{52} \\ B_{52} \\ A_{54} \\ B_{54} \\ A_{55} \\ B_{55} \\ A_{56} \\ B_{56} \\ A_{58} \\ B_{58} \\ A_{59}-A_{49}\Delta Z_{10} \\ B_{59}-B_{49}\Delta Z_{10} \\ A_{5,11} \\ B_{5,11} \\ A_{5,12} \\ B_{5,12} \\ A_{5,13} \\ B_{5,13} \\ 0 \\ 0 \end{bmatrix} \qquad x = \begin{bmatrix} \Delta Z_1 \\ \Delta Q_2 \\ \Delta Z_2 \\ \Delta Q_3 \\ \Delta Q_4 \\ \Delta Q_5 \\ \Delta Z_5 \\ \Delta Q_6 \\ \Delta Z_6 \\ \Delta Q_7 \\ \Delta Q_8 \\ \Delta Q_9 \\ \Delta Z_9 \\ \Delta Q_{10} \\ \Delta Q_{11} \\ \Delta Q_{12} \\ \Delta Z_{12} \\ \Delta Q_{13} \\ \Delta Z_{13} \\ \Delta Q_{14} \\ \Delta Z_8 \\ \Delta Z_7 \end{bmatrix}$$

La matriz A tendrá entonces la estructura siguiente de elementos no nulos:



16. RESOLUCION DEL SISTEMA DELTAICO

Sea entonces la ecuación

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} x = b \qquad (59)$$

donde A y b tienen la estructura indicada en la sección anterior, siendo N el orden de la matriz, n el número total de puntos de discretización, m el de tramos simples de la red, s el de extremos abiertos, aguas arriba o aguas abajo, y t el de nodos de confluencia. Demostraremos que - si ningún pivote se anula - se puede resolver el sistema $Ax = b$ en un número de alrededor de $28n+t^3/3$ de pasos almacenando la matriz en aproximadamente $6n+t^2$ posiciones de memoria.

TEOREMA: Mediante operaciones gaussianas de eliminación el sistema lineal (59) se puede transformar en un sistema

$$A'x = b' \qquad (60)$$

donde la matriz A' es de la forma

$$A' = \begin{bmatrix} A'_1 & A'_2 \\ A'_3 & A'_4 \end{bmatrix}$$

y las submatrices A'_1, A'_2, A'_3 y A'_4 tienen la siguiente forma:

1) A'_1 es triangular superior, compuesta por submatrices bidiagonales $A'_{1,j}$

$$A'_1 = \begin{bmatrix} A'_{11} & & & \\ & A'_{12} & & \\ & & \ddots & \\ & & & A'_{1m} \end{bmatrix}$$

tales que

$$A'_{1j} = \begin{bmatrix} \times\times & & & \\ \times\times & & & \\ \times\times & & & \\ & \ddots & & \\ & & \times\times & \\ & & \times\times & \\ & & \times & \end{bmatrix}$$

2) Ninguna columna de A_2' tiene más de 4.máx $(n_j-1) + 1$, con $1 \leq j \leq m$, elementos no nulos, y forman hasta dos sucesiones consecutivas (con lo cual basta indicar el comienzo y el tamaño de cada sucesión para tener definida A_2'). n_j es el número de puntos de discretización del j-simo tramo simple.

3) $A_3' = 0$

4) A_4' es triangular superior.

Pero entonces A' es triangular superior, y el sistema $A'x = b'$ se resuelve realizando un barrido ascendente.

DEMOSTRACION: Supongamos que hemos ya triangulado el sistema lineal hasta el último elemento de la diagonal de la submatriz $A_{1,j-1}$ (la primera vez, no hemos triangulado nada). Sea entonces $A_{1,j}$ una submatriz. Supongamos que el extremo aguas arriba del tramo j sea abierto. Entonces, dicho extremo no pertenece a ningún nodo de confluencia, y ninguna fila de la matriz A_3 tiene elementos en columnas que corten $A_{1,j}$, excepto tal vez la última, puesto que sólo la primera y última columnas de $A_{1,j}$ pueden tener elementos no nulos en sus últimos t elementos. Tampoco hay ninguna columna de A_2 que tenga elementos en filas que corten $A_{1,j}$, por el mismo motivo, salvo tal vez las dos últimas.

Por consiguiente, $A_{1,j}$ se triangula aisladamente en la forma indicada en la sección 7, hasta llegar al último par de ecuaciones, digamos entre los puntos de discretización k-1 y k; es decir, k es el último punto de discretización del tramo. Si el extremo aguas abajo no es un nodo de confluencia, la modelización resultó ser la de un tramo simple unidimensional aislado, y no la de una red deltaica. De lo contrario, se tiene un nodo de confluencia.

Entonces se tendrá, usando la notación de la sección 7, si se trianguló hasta las filas que relacionan las incógnitas ΔQ_{k-1}, ΔZ_{k-1}, ΔQ_k, ΔZ_k

$$
\begin{array}{cccc|c|c}
\cdot & & & & \cdot & \\
x & & & & 0 & \\
x & x & & & 0 & \\
 & x & & & 0 & \\
 & A_{1,k-1} & A_{2,k-1} & A_{3,k-1} & A_{4,k-1} & A_{5,k-1} \\
 & B_{1,k-1} & B_{2,k-1} & B_{3,k-1} & B_{4,k-1} & = B_{5,k-1} \\
\hline
 & & & 1 & c_{uu} & b_{N-t+u}
\end{array}
$$

Aquí el nodo de confluencia en cuestión es el u-simo, y los cuatro cuadrantes indican, empezando en el superior izquierdo y siguiendo en el sentido de las agujas del reloj, elementos de los bloques A'_1, A'_2, A'_4, y A'_3 que están sobre las mismas filas y/o columnas. El elemento c_{uu} de A'_4 está ubicado sobre la fila N-t+u y la columna N-t+u, y puede *no* ser cero, pues puede haber sido afectado anteriormente, al igual que b_{N-t+u}.

Realizando las operaciones de tridiagonalización (36) y (37) resulta entonces

$$
\begin{array}{cccc|c|c}
 & & & & & \\
 & x & x & & 0 & \\
 & & x & x & 0 & C_{4,k-1} \\
 & & C_{1,k-1} & C_{2,k-1} \; C_{3,k-1} & D_{3,k-1} & = D_{4,k-1} \\
 & & & D_{1,k-1} \; D_{2,k-1} & & \\
\hline
 & & & 1 & c_{uu} & b_{N-t+u}
\end{array}
$$

Usando ahora (46.k-1) se obtiene

$$\begin{array}{cc|c} x \quad x & & \\ x \quad x & & \\ A_{1,1,k-1} & A_{1,2,k-1} & 0 \\ & A_{2,1,k-1} & A_{2,2,k-1} \\ \hline & & \\ & 1 & c_{uu} \end{array} = \begin{array}{c} \\ \\ A_{1,3,k-1} \\ A_{2,3,k-1} \\ \\ b_{N-t+u} \end{array} \qquad (61)$$

de donde se obtiene

$$\begin{array}{cc|c} A_{1,1,k-1} & A_{1,2,k-1} & 0 \\ & A_{2,1,k-1} & A_{2,2,k-1} \\ \hline & 0 & c_{uu}-A_{2,2,k-1}/A_{2,1,k-1} \end{array} = \begin{array}{c} A_{1,3,k-1} \\ A_{2,3,k-1} \\ \\ b' \end{array}$$

con $b'=b_{N-t+u}-A_{2,3,k-1}/A_{2,1,k-1}$.

Así se triangula la matriz **A** hasta el elemento diagonal correspondiente al punto de discretización k.

Supongamos ahora que el extremo aguas arriba i del tramo j pertenezca al q-simo nodo de confluencia.

Entonces las filas de **A** correspondientes a las filas de la submatriz $A_{1,j}$ y las de A_2 y A_3 relacionadas con el nodo serán

$$
\begin{array}{llllll|c|c}
A_{1i} & A_{3i} & A_{4i} & & & & A_{2i} & A_{5i} \\
B_{1i} & B_{3i} & B_{4i} & & & & B_{2i} & B_{5i} \\
 & A_{1i+1} & A_{2i+1} & A_{3i+1} & A_{4i+1} & & 0 & A_{5i+1} \\
 & B_{1i+1} & B_{2i+1} & B_{3i+1} & B_{4i+1} & & 0 \quad = & B_{5i+1} \\
 & & & \cdot & & & \cdot & \cdot \\
 & & & \cdot & & & \cdot & \cdot \\
\hline
 & \cdot & & \cdot & & & \cdot & \\
-1 & 0 & 0 & & & & c_{qq} & b_{N-t+q}
\end{array}
$$

es decir, en la submatriz A_2 habrá dos elementos no nulos sobre su columna q (que es la columna N-t+q de la matriz A´), en los lugares correspondientes a las filas mencionadas. Sobre esa columna, no habrá más elementos no nulos correspondientes a filas de la submatriz $A_{1,j}$. Además, en la fila q-sima de la submatriz A_3 (la fila N-t+q de la matriz A´), habrá un -1 en la columna correspondiente a la primera columna de $A_{1,j}$, y todos los otros elementos sobre columnas de $A_{1,j}$ serán nulos. Se triangula entonces $A_{1,j}$ haciendo intervenir la q-sima fila de A_3 y la q-sima columna de A_2. El procedimiento es el siguiente:

Primero se tridiagonaliza cada par de ecuaciones mediante el uso de las fórmulas (36) y (37), lo que convierte a los elementos A_{2i} y B_{2i}de la columna q de A_2' en C_{2i} y D_{1i}

$$
\begin{array}{llll|c|c}
C_{1i} & C_{3i} & & & C_{2i} & C_{4i} \\
 & D_{2i} & D_{3i} & & D_{1i} & D_{4i} \\
 & C_{1i+1} & C_{2i+1} & C_{3i+1} & 0 & C_{4i+1} \\
 & & & \cdot & \cdot & \cdot \\
\hline
1 & & & & c_{qq} & b_{N-t+q}
\end{array}
$$

Para completar la triangulación de A´ hasta la segunda fila de A_{1j}' es necesario anular el 1 sobre la fila q-sima de A_3' , o sea la fila N-t+q de A´. Si definimos

$$a = -C_{3i}/C_{1i}$$

$$c_{qq} := c_{qq} - C_{2i}/C_{1i}$$

$$b'_{N-t+q} = b_{N-t+q} - C_{4i}/C_{1i}$$

se tendrá

$$\begin{array}{ccccccc|c|c}
C_{1i} & C_{3i} & & & & & & C_{2i} & C_{4i} \\
 & D_{2i} & D_{3i} & & & & & D_{1i} & D_{4i} \\
 & C_{1i+1} & C_{2i+1} & C_{3i+1} & & & & 0 & C_{4i+1} \\
 & & D_{1i+1} & D_{2i+1} & D_{3i+1} & & & 0 & D_{4i+1} \\
 & & & & & \ddots & & \vdots & \vdots \\
\hline
0 & a & 0 & & & & & c_{qq} & b'_{N-t+q}
\end{array}$$

Aquí usamos el símbolo := de asignación para indicar la variable obtenida con el mismo nombre que una variable ya usada, que no volveremos a emplear. Así aligeramos la notación. Repetiremos este procedimiento más abajo.

Si ahora definimos

$$A_{1,1,i}=C_{1i} \qquad A_{1,2,i}=C_{3i} \qquad A_{1,3,i}=C_{4i} \qquad A_{1,4,i}=C_{2i}$$

$$A_{2,1,i}=D_{2i} \qquad A_{2,2,i}=D_{3i} \qquad A_{2,3,i}=D_{4i} \qquad A_{2,4,i}=D_{1i}$$

es decir, llamamos $A_{1,4,i}$ y $A_{2,4,i}$ a los elementos de la columna N-t+q con los que trabajamos, podemos proseguir la triangulación usando las ecuaciones (46) recurrentemente, y, además, las submatrices A'_2, A'_3, A'_4 y el vector b se modifican del siguiente modo al llegar al punto p (y *en ese orden*, debido al uso del signo de asignación :=):

$$A_{1,4,p} = - A_{2,4,p-1}.C_{1,p}/A_{2,1,p-1}$$

$$c_{qq} := c_{qq} - A_{2,4,p-1}.a /A_{2,1,p-1}$$

$$b_{N-t+q} := b_{N-t+q} - A_{2,3,p-1}.a /A_{2,1,p-1}$$

$$a := - A_{2,2,p-1}.a /A_{2,1,p-1}$$

$$A_{2,4,p} = -A_{1,4,p}.D_{1,p}/A_{1,1,p}$$

$$c_{qq} := c_{qq} - A_{1,4,p}.a\ /A_{1,1,p}$$

$$b_{N-t+q} := b_{N-t+q} - A_{1,3,p}.a\ /A_{1,1,p}$$

$$a := -A_{1,2,p}.a\ /A_{1,1,p}$$

Obsérvese que sobre la fila N-t+q hay, en las columnas que cortan a $A'_{1,j}$, un solo elemento distinto de cero, que llamamos siempre a y que se va corriendo a la derecha de a una columna por vez.

Al llegar a las dos últimas filas de $A_{1,j}$ se tendrá, si el extremo inferior k del tramo j es abierto:

1) Condición de contorno en el extremo abierto aguas abajo nivel conocido Z_k: en ese caso

$$\begin{array}{ccc|cc}
A_{2,1,k-2} & A_{2,2,k-2} & & A_{2,4,k-2} & A_{2,3,k-2} \\
C_{1,k-1} & C_{2,k-1} & C_{3,k-1} & 0 & C_{4,k-1} \\
 & D_{1,k-1} & D_{2,k-1} & 0 & =D_{4,k-1}-D_{3,k-1}\Delta Z_k \\
\hline
 & . & . & . & . \\
a & 0 & 0 & c_{qq} & b_{N-t+q}
\end{array}$$

y la triangulación finaliza con

$$A_{1,1,k-1}=C_{2,k-1}-A_{2,2,k-2}.C_{1,k-1}/A_{2,1,k-2}$$

$$A_{1,2,k-1}=C_{3,k-1}$$

$$A_{1,4,k-1}=-A_{2,4,k-2}.C_{1,k-1}/A_{2,1,k-2}$$

$$A_{1,3,k-1}=C_{4,k-1}-A_{2,3,k-2}.C_{1,k-1}/A_{2,1,k-2}$$

$$c_{qq}:=c_{qq} - A_{2,4,k-2}.\ a/A_{2,1,k-2}$$

$$b_{N-t+q} := b_{N-t+q} - A_{2,3,k-2}.\ a/A_{2,1,k-2}$$

$$a:=-A_{2,2,k-2}.\ a/A_{2,1,k-2}$$

$$A_{2,1,k-1} = D_{2,k-1} - A_{1,2,k-1} \cdot D_{1,k-1} / A_{1,1,k-1}$$

$$A_{2,4,k-1} = -A_{1,4,k-1} \cdot D_{1,k-1} / A_{1,1,k-1}$$

$$A_{2,3,k-1} = D_{4,k-1} - D_{3,k-1} \Delta Z_k - A_{1,3,k-1} \cdot D_{1,k-1} / A_{1,1,k-1}$$

$$c_{qq} := c_{qq} - A_{1,4,k-1} \cdot a / A_{1,1,k-1}$$

$$b_{N-t+q} := b_{N-t+q} - A_{1,3,k-1} \cdot a / A_{1,1,k-1}$$

$$a := -A_{1,2,k-1} \cdot a / A_{1,1,k-1}$$

$$c_{qq} := c_{qq} - A_{2,4,k-1} \cdot a / A_{2,1,k-1} \qquad (62)$$

$$b_{N-t+q} := b_{N-t+q} - A_{2,3,k-1} \cdot a / A_{2,1,k-1}$$

2) Condición de contorno aguas abajo conocida caudal Q_k.

En ese caso

$$\begin{array}{ccc|c|c} A_{2,1,k-2} & A_{2,2,k-2} & & A_{2,4,k-2} & A_{2,3,k-2} \\ C_{1,k-1} & C_{2,k-1} & & 0 & C_{4,k-1} - C_{3,k-1} \Delta Q_k \\ & D_{1,k-1} & D_{3,k-1} & 0 & = D_{4,k-1} - D_{2,k-1} \Delta Q_k \\ & & . & . & . \\ \hline . & . & & . & . \\ a & 0 & 0 & c_{qq} & b_{N-t+q} \end{array}$$

y la triangulación finaliza con

$$A_{1,1,k-1} = C_{2,k-1} - A_{2,2,k-2} \cdot C_{1,k-1} / A_{2,1,k-2}$$

$$A_{1,2,k-1} = 0$$

$$A_{1,4,k-1} = -A_{2,4,k-2} \cdot C_{1,k-1} / A_{2,1,k-2}$$

$$A_{1,3,k-1} = C_{4,k-1} - C_{3,k-1} \Delta Q_k - A_{2,3,k-2} \cdot C_{1,k-1} / A_{2,1,k-2}$$

$$c_{qq} := c_{qq} - A_{2,4,k-2} \cdot a / A_{2,1,k-2}$$

$$b_{N-t+q} := b_{N-t+q} - A_{2,3,k-2} \cdot a/A_{2,1,k-2}$$

$$a := -A_{2,2,k-2} \cdot a/A_{2,1,k-2}$$

$$A_{2,1,k-1} = D_{3,k-1}$$

$$A_{2,4,k-1} = -A_{1,4,k-1} D_{1,k-1}/A_{1,1,k-1}$$

$$A_{2,3,k-1} = D_{4,k-1} - D_{2,k-1}\Delta Q_k - A_{1,3,k-1} \cdot D_{1,k-1}/A_{1,1,k-1} \qquad (63)$$

$$c_{qq} := c_{qq} - A_{1,4,k-1} \cdot a/A_{1,1,k-1}$$

$$b_{N-t+q} := b_{N-t+q} - A_{1,3,k-1} \cdot a/A_{1,1,k-1}$$

En ambos casos, queda triangulado el bloque $A'_{1,j}$, y anulados los elementos de la fila N-t+q que están sobre columnas que intersecan a $A'_{1,j}$.

Supongamos ahora que el extremo aguas abajo k corresponde al nodo de confluencia r. Entonces interviene en la eliminación otra fila de A'_3 , la r-sima, y otra columna de A'_2 , la r-sima, y se tiene

$$\begin{array}{ccc|cc|c}
 & & & & 0 & \\
A_{2,1,k-2} & A_{2,2,k-2} & & A_{2,4,k-2} & 0 & A_{2,3,k-2} \\
C_{1,k-1} & C_{2,k-1} & C_{3,k-1} & 0 & 0 & = C_{4,k-1} \\
 & D_{1,k-1} & D_{2,k-1} & 0 & D_{3,k-1} & D_{4,k-1} \\
\hline
a & 0 & 0 & c_{qq} & c_{qr} & b_{N-t+q} \\
 & & 1 & c_{rq} & c_{rr} & b_{N-t+r}
\end{array}$$

La triangulación hasta la siguiente fila requiere las siguientes operaciones:

$$A_{1,1,k-1} = C_{2,k-1} - A_{2,2,k-2} \cdot C_{1,k-1}/A_{2,1,k-2}$$

$$A_{1,2,k-1} = C_{3,k-1}$$

$$A_{1,4,k-1} = - A_{2,4,k-2} \cdot C_{1,k-1}/A_{2,1,k-2}$$

$$A_{1,3,k-1} = C_{4,k-1} - A_{2,3,k-2} \cdot C_{1,k-1}/A_{2,1,k-2}$$

$$c_{qq} := c_{qq} - A_{2,4,k-2} \cdot a/A_{2,1,k-2}$$

$$b_{N-t+q} := b_{N-t+q} - A_{2,3,k-2} \cdot a/A_{2,1,k-2}$$

$$a := -A_{2,2,k-2} \cdot a/A_{2,1,k-2}$$

con lo cual se obtiene

$$\begin{array}{cc|cc} A_{1,1,k-1} & A_{1,2,k-1} & A_{1,4,k-1} & 0 \\ D_{1,k-1} & D_{2,k-1} & 0 & D_{3,k-1} \\ \hline a & 0 & c_{qq} & c_{qr} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & c_{rq} & c_{rr} \end{array} = \begin{array}{c} A_{1,3,k-1} \\ D_{4,k-1} \\ \\ b_{N-t+q} \\ \vdots \\ b_{N-t+r} \end{array}$$

o sea, fórmulas muy similares a las primeras de (63). Finalmente, para completar la triangulación, será necesario usar las igualdades o asignaciones

$$A_{2,1,k-1} = D_{2,k-1} - A_{1,2,k-1} \cdot D_{1,k-1}/A_{1,1,k-1}$$

$$A_{2,4,k-1} = -A_{1,4,k-1} \cdot D_{1,k-1}/A_{1,1,k-1}$$

$$A_{2,3,k-1} = D_{4,k-1} - A_{1,3,k-1} \cdot D_{1,k-1}/A_{1,1,k-1}$$

$$c_{qq} := c_{qq} - A_{1,4,k-1} \cdot a/A_{1,1,k-1}$$

$$b_{N-t+q} := b_{N-t+q} - A_{1,3,k-1} \cdot a/A_{1,1,k-1}$$

$$a := -A_{1,2,k-1} \cdot a/A_{1,1,k-1}$$

con lo cual se tiene

$A_{2,1,k-1}$	$A_{2,4,k-1}$	$D_{3,k-1}$	=	$A_{2,3,k-1}$
a	c_{qq}	c_{qr}		b_{N-t+q}
1	c_{rq}	c_{rr}		b_{N-t+r}

y por último

$$c_{qq} := c_{qq} - A_{2,4,k-1} \cdot a/A_{2,1,k-1}$$

$$c_{qr} := c_{qr} - D_{3,k-1} \cdot a/A_{2,1,k-1}$$

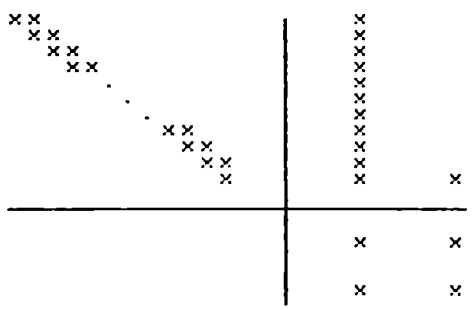
$$b_{N-t+q} := b_{N-t+q} - A_{2,3,k-1} \cdot a/A_{2,1,k-1}$$

$$c_{rq} := c_{rq} - A_{2,4,k-1}/A_{2,1,k-1}$$

$$c_{rr} := c_{rr} - D_{3,k-1}/A_{2,1,k-1}$$

$$b_{N-t+r} := b_{N-t+r} - A_{2,3,k-1}/A_{2,1,k-1}$$

De este modo se triangula toda la submatriz A_1, y se va anulando la submatriz A_3. La estructura que queda se puede representar como



Por consiguiente, al terminar este proceso hemos llegado a un sistema lineal

$$\begin{bmatrix} A_1' & A_2' \\ 0 & A_4' \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

donde A_1' es triangular superior y está formada por bloques bidiagonales. Entonces, para obtener la solución, basta resolver por el método de Gauss el sistema de t x t $A_4'x_2 = b_2'$ y luego aplicar el barrido descendente a $A_1'x_1 = b_1' - A_2'x_2$, pues x_2 ya se conoce. ■

Con esto queda (¡por fin!) demostrado el teorema. Pero no es necesario almacenar A_2 (y A_2') en (N-t) x t lugares. Veremos que son necesarios solamente (2g+1) x t, donde g es el orden de la mayor de las submatrices $A_{i,j}$, o sea $g = \max_{1 \le j \le m} 2(n_j - 1)$, con n_j= número de puntos de j-simo tramo simple. En efecto, se presentan dos posibilidades: que el nodo al cual corresponde la q-sima columna de la submatriz A_2' sea de afluencia o de efluencia.

Si el nodo al cual corresponde la q-sima columna de la submatriz A_2 (o sea la N-t+q-sima columna de A) es un nodo de afluencia, sean A_{j_1}, A_{j_2} y A_{j_3} las submatrices de A_1 correspondientes a los tramos de los cuales los puntos de discretización del nodo son extremos aguas arriba o aguas abajo. Supongamos que los puntos de discretización del nodo sean extremos aguas arriba del tramo j_1 y aguas abajo de los tramos j_2 y j_3. Entonces la columna q de A_2 tendrá ceros en todos sus elementos excepto algunos de las filas que cortan a A_{j_1}, A_{j_2} y A_{j_3}, que analizaremos. Constatemos en primer lugar que , por la estructura de bloque diagonales de las A_{j_k} en A, las operaciones gaussianas con filas en una submatriz no afectan las demás. Por consiguiente, al terminar el proceso antes descripto y llegar a las submatrices A_1', A_2', A_3' y A_4', las filas de la columna q correspondientes a A_{j_2} y A_{j_3} tendrán solamente el *último* elemento no nulo, como se ve en (61) y las filas correspondientes a A_{j_1} tendrán los dos primeros elementos no nulos.

Al realizar las operaciones con las filas antes indicadas, se llenarán (presumiblemente) de elementos no nulos los restantes elementos de este conjunto de filas sobre la columna q, como se ve en (62) y (63), pues se reemplazarán los ceros por los elementos de la fila de arriba divididos por el pivote y cambiados de signo. Esto da un total de 2+g elementos no nulos por nodo de confluencia, a lo sumo.

Si lo que se tiene es un nodo de efluencia, habrá dos puntos del nodo correspondientes a extremos aguas arriba (y por consiguiente, al realizar operaciones gaussianas, reemplazarán los ceros por elementos no nulos en el resto de las filas correspondientes a su tramo) y un punto del nodo correspondiente a un extremo aguas abajo, que tiene solamente el último elemento no nulo, que no se alteran con la modificación del sistema. O sea que finalmente habrá a lo sumo 2g+1 elementos no nulos de esa columna.

En suma, como 2g+1 > g+2, habrá a lo sumo (2g+1) x t elementos no nulos en la matriz transformada A_2'.

El número total máximo de palabras necesarias para almacenar la matriz A y el vector de la derecha b será entonces:

10 para almacenar las ecuaciones de Saint-Venant discretizadas (no 10 por intervalo, pues al llegar a cada intervalo se tridiagonaliza, y basta usar las mismas 10 variables).

3(N-t) correspondientes a las submatrices trianguladas (y los correspondientes términos de la derecha). Como

$$N=2(n-t)-s$$

ese valor será 3(2(n-t)-s-t)=6n-9t-3s.

(2g+1)t correspondientes a A_2.

1 elemento correspondiente a A_3. Es la variable a usada en (61), (62) y (63).

t^2 +t elementos correspondientes a la submatriz cuadrada A_4 y los términos de la derecha de sus filas (no se ha hecho ningún esfuerzo por reducir esta submatriz, que será bastante rala, pues

no se considera que el costo de programación lo justifica, habiéndose ya reducido el almacenamiento necesario de $N^2 + N$ a menos de $3N + t^2 + t$, con t usualmente mucho menor que N).

Esto hace un total (*sin incluir*, vale la pena mencionar, el sistema de punteros necesario para definir los valores no nulos de la matriz) de

$$10 + 6n - 9t - 3s + 2g + 1 + 1 + t^2 + t =$$

$$12 + 6n + t^2 - 8t - 3s + 2g$$

que es un número del orden de $6n+t^2$.

En esencia, si suponemos una "densidad de confluencias" no mayor de 1/10, lo cual es bastante razonable, usaremos una cantidad de memoria del orden de $6n + n^2/100$. Si se tienen 100 puntos de discretización, eso significa 600 + 100 =700 posiciones de memoria, en lugar de n^2+n= 10100, lo cual constituye un ahorro realmente significativo.

En cuanto al número máximo de operaciones necesarias (siempre considerando pivote diagonal en todas las operaciones) es (incluimos como siempre las operaciones sobre b):

1) 1 división y 7 sumas y multiplicaciones para cada punto de discretización para tridiagonalizar cada matriz A_{ij}, o sea 8(n-m) operaciones en total.

2) Suponiendo, para simplificar, que todos los tramos tienen nodo de confluencia aguas arriba y abajo, lo cual es evidentemente absurdo, pero es la situación que más operaciones requiere, a lo sumo $2(2n_j-2)$ divisiones y $6(2n_j-2)$ sumas y multiplicaciones por submatriz A_{ij} para triangularla,o sea a lo sumo 4n-4m divisiones y 12n-12m sumas y multiplicaciones, es decir del orden de 16n operaciones en total como máximo.

3) Las operaciones necesarias para resolver el sistema $A'_4 x_2 = b_2$. Aquí no se ha intentado optimizar el uso de memoria ni la cantidad de operaciones. Se resuelve dicho sistema por el método de Gauss con pivoteo parcial por columnas, lo cual hace un total de $t^3/3$ operaciones (despreciando los términos de orden t^2 y t, entre ellos las divisiones).

4) Las operaciones necesarias para resolver $A'_1 x_1 = b_1 - A'_2 x_2$, que son a lo sumo n divisiones y 3n sumas y multiplicaciones.

Esto hace un total del orden de 6n divisiones y $22n + t^3/3$ sumas y multiplicaciones, o sea un total del orden de $28n + t^3/3$. Si suponemos como antes 100 puntos de discretización y 10 % de nodos de confluencia, se tendrán aproximadamente 2800 + 333 operaciones, cifra sensiblemente inferior a los 333000 necesarios usando un método de Gauss global.

17. ANALISIS DE CONSISTENCIA DEL MODELO DELTAICO

En primer lugar se efectuaron varias corridas de consistencia del modelo deltaico. Estas son:

1) Corrida de un tramo simple. Se corrió el modelo con un tramo simple indicado en el Apéndice 2, y se compararon los resultados con los obtenidos con el modelo arborescente para ese tramo. Los resultados, totalmente satisfactorios (es decir, coincidentes con los obtenidos con el modelo de red arborescente), se indican en el mencionado apéndice.

2) Corrida del tramo con afluente (modelo arborescente indicado en el cuadro 7). Se corrió con las condiciones de contorno indicadas en el cuadro 11 (corrida 1) y luego con las condiciones de contorno intercambiadas (mismo cuadro, corrida 2). La coincidencia fue total (para la precisión usada) en ambos casos. Las condiciones iniciales fueron las indicadas en el cuadro 9.

3) Corrida del tramo con efluente indicado en la figura 19, suponiendo las ecuaciones simplificadas (lineales con coeficientes constantes) usadas en la sección 11 de la segunda parte de este trabajo, o sea

$$\frac{\partial u}{\partial t} + \frac{\partial h}{\partial x} = 0$$

$$\frac{\partial h}{\partial t} + \frac{\partial u}{\partial x} = 0$$

Se puede observar que el teorema de existencia de solución de este sistema con condiciones iniciales (51), condiciones de contorno (52) en cada extremo abierto, y condiciones de compatibilidad (53) en cada nodo de confluencia, demostrado en la sección 11 de la segunda parte, se generaliza inmediatamente para redes deltaicas, dado que en ningún momento de dicha demostración se usa que la red es arborescente.

Las coordenadas espaciales de los puntos de discretización de la red (en este caso, es una red arborescente dirigida desde la raiz, que es el extremo abierto aguas arriba) se indican en el cuadro 18.

Cuadro 18

Modelo con efluente para prueba de ecuaciones simplificadas

Coordenadas espaciales

Punto	Coordenada espacial	Punto	Coordenada espacial
1	0	9	7Π/6
2	Π/6	10	4Π/3
3	Π/3	11	3Π/2
4	Π/2	12	Π
5	2Π/3	13	7Π/6
6	5Π/6	14	4Π/3
7	Π	15	3Π/2
8	Π		

Las condiciones iniciales consideradas son las siguientes:

En el tramo 1-7:

$$u_o(x) = 10 + \cos x$$
$$h_o(x) = 7 + \text{sen } x$$

En los tramos 8-11 y 12-15:

$$u_o(x) = 5 + 0{,}5 \cos x$$
$$h_o(x) = 7 + 0{,}5 \text{ sen } x$$

Las condiciones de contorno dadas serán las siguientes:

$$u(x_1,t) = 10 + \cos t - \text{sen } t$$
$$h(x_{11},t) = 7 - 0{,}5 (\cos t + \text{sen } t) = h(x_{15},t)$$

La solución analítica del sistema diferencial planteado es

En el tramo 1-7:

$$u(x,t) = 10 + (\cos t - \text{sen } t) \cos x$$
$$h(x,t) = 7 + (\cos t + \text{sen } t) \text{ sen } x$$

En los tramo 8-11 y 12-15:

$$u(x,t) = 5 + 0{,}5 (\cos t - \text{sen } t) \cos x$$
$$h(x,t) = 7 + 0{,}5 (\cos t + \text{sen } t) \text{ sen } x$$

De acuerdo a lo ya mencionado, para $\Theta=1/2$ y $\Delta x=\Delta t$ (en este caso ambas son iguales a Π/6), la solución numérica deberá coincidir con la solución analítica. Esto es exactamente lo que sucede. En el cuadro 19 se indica el estado del modelo en el instante 2Π, que

coincide completamente con el estado en el instante inicial, como debe suceder por periodicidad, y en el cuadro 20 se indica el estado del modelo en el instante Π, que coincide también con los resultados teóricos, como se ve efectuando las operaciones correspondientes.

Cuadro 19

Estado del modelo teórico con efluente en el instante 2Π
($\Theta=1/2$, $\Delta t=\Delta x=\Pi/6$)

Punto	1	2	3	4	5	6	7	8
h	7,000	7,500	7,866	8,000	7,866	7,500	7,000	7,000
u	11,000	10,866	10,500	10,000	9,500	9,134	9,000	4,500

Punto	9	10	11	12	13	14	15
h	6,750	6,567	6,500	7,000	6,750	6,567	6,500
u	4,567	4,750	5,000	4,500	4,567	4,750	5,000

Cuadro 20

Estado del modelo teórico con efluente en el instante Π
($\Theta=1/2$, $\Delta t=\Delta x=\Pi/6$)

Punto	1	2	3	4	5	6	7	8
h	7,000	6,500	6,134	6,000	6,134	6,500	7,000	7,000
u	9,000	9,134	9,500	10,000	10,500	10,866	11,000	5,500

Punto	9	10	11	12	13	14	15
h	7,250	7,433	7,500	7,000	7,250	7,433	7,500
u	5,433	5,250	5,000	5,500	5,433	5,250	5,000

Posteriormente, en forma análoga a la de dicha sección, se corrió el modelo con $\theta=1/4$, para comprobar la inestabilidad. El modelo se interrumpió en el instante $7\Pi/6$, debido a que en el punto de discretización 1 la variable h tomó un valor negativo. En el cuadro 21 se indican los valores tomados por dicha variable en ese punto hasta el momento de la interrupción:

Cuadro 21

Valores de h en el punto 1 hasta la desestabilización

Tiempo	0	$\Pi/6$	$\Pi/3$	$\Pi/2$	$2\Pi/3$	$5\Pi/6$	Π
h_1	7,000	6,963	7,026	7,183	7,463	7,041	10,177

Ya en el instante Π se observa la inestabilidad.

Se corrió el modelo ahora con $\theta=3/4$ y $\theta=1$ para $\Delta t=\Delta x=\Pi/6$, para comprobar la atenuación de la onda con θ mayores. Los resultados obtenidos en el tiempo 2Π para los puntos de discretización 1 a 7 se indican en el cuadro 22 y en la figura 20:

Cuadro 22

Comparación de alturas teóricas con calculadas en el modelo simplificado para $\theta=3/4$ y $\theta=1$ ($\Delta t=\Delta x=\Pi/6, t=2\Pi$) para los puntos 1 a 7

Punto	1	2	3	4	5	6	7
Teórica	7,000	7,500	7,866	8,000	7,866	7,500	7,000
$\theta=2/3$	7,523	7,897	8,090	8,085	7,899	7,568	7,144
$\theta=1$	7,948	8,204	8,256	8,147	7,922	7,615	7,243

Como se ve en el cuadro (y más claramente en la figura 20) se produce una amortiguación de la onda. Esto se nota más tomando el "histograma" de h en función del tiempo para los tres casos, como se ve en el cuadro 23 y en la figura 21, ambos con datos en el punto de discretización 4:

Cuadro 23

Comparación de alturas teóricas con calculadas en el modelo simplificado para $\theta=3/4$ y 1 en el punto 4 para $\Delta t=\Delta x=\Pi/6$

Tiempo	0	Π/6	Π/3	Π/2	2Π/3	5Π/6	Π
Teórica	8,000	8,366	8,366	8,000	7,366	6,634	6,000
$\theta=3/4$	8,000	8,276	8,203	7,836	7,281	6,662	6,129
$\theta=1$	8,000	8,193	8,071	7,705	7,190	6,645	6,205

Tiempo	7Π/6	4Π/3	3Π/2	5Π/3	11Π/6	2Π
Teórica	5,634	5,634	6,000	6,634	7,366	8,000
$\theta=3/4$	5,821	5,833	6,177	6,775	7,475	8,085
$\theta=1$	5,991	6,072	6,440	7,009	7,634	8,147

Con esto se da por terminado el análisis teórico, pues se considera superfluo repetir todos los experimentos de la sección 10, dada la consistencia de los resultados.

18. EXPERIMENTOS NUMERICOS CON REDES DELTAICAS

Los experimentos numéricos que se indican en esta sección fueron realizados con el modelo de red fluvial deltaica completo indicado en la figura 22, cuya descripción se da en el cuadro 24.

Cuadro 24

Descripción del modelo de red fluvial deltaico

Punto de disc.	1	2	3	4	5	6
Coordenada (km)	0	1	2	0	0,5	1
Nivel de fdo (m)	100,60	100,20	99,80	100,10	99,85	99,60
Ancho (*) (m)	100	100	100	100	100	100
D (**)(1000m^3/s)	41,1	41,1	41,1	41,1	41,1	41,1
Punto de disc.	7	8	9	10	11	12
Coordenada (km)	0	1	2	2	3	4
Nivel de fdo (m)	100,00	99,90	99,80	99,80	99,70	99,60
Ancho (m)	100	100	100	200	200	200
D (1000m^3/s)	41,1	41,1	41,1	43,6	43,6	43,6
Punto de disc.	13	14	15	16	17	18
Coordenada (km)	4	5	6	6	7	8
Nivel de fdo (m)	99,60	99,50	99,40	99,40	99,30	99,20
Ancho (m)	300	300	300	150	150	150
D (1000m^3/s)	44,5	44,5	44,5	42,7	42,7	42,7
Punto de disc.	19	20	21	22	23	24
Coordenada (km)	9	9	10	11	11	12
Nivel de fdo (m)	99,10	99,10	99,00	98,90	98,90	98,80
Ancho (m)	150	250	250	250	150	150
D (1000m^3/s)	42,7	44,1	44,1	44,1	42,7	42,7

Punto de disc.	25	26	27	28	29	30
Coordenada (km)	13	13	14	15	15	16
Nivel de fdo (m)	98,70	98,70	98,60	98,50	98,50	98,40
Ancho (m)	150	250	250	250	180	180
D ($1000m^3/s$)	42,7	44,1	44,1	44,1	43,3	43,3

Punto de disc.	31	32	33	34	35	36
Coordenada (km)	17	6	6,6	7,2	7,8	7,8
Nivel de fdo (m)	98,30	99,40	99,34	99,28	99,22	99,22
Ancho (m)	180	150	150	150	150	100
D ($1000m^3/s$)	43,3	42,7	42,7	42,7	42,7	41,1

Punto de disc.	37	38	39	40	41	42
Coordenada (km)	8,4	9	11	12	13	7,6
Nivel de fdo (m)	99,16	99,10	98,90	98,80	98,70	99,22
Ancho (m)	100	100	100	100	100	50
D ($1000m^3/s$)	41,1	41,1	41,1	41,1	41,1	37,1

Punto de disc.	43	44	45	46	47	48
Coordenada (km)	8,2	8,6	15	15,5	16	16,5
Nivel de fdo (m)	99,14	99,06	98,50	98,35	98,20	98,05
Ancho (m)	50	50	70	70	70	70
D ($1000m^3/s$)	37,1	37,1	39,3	39,3	39,3	39,3

(*) El ancho de cada sección transversal es constante, es decir, las secciones transversales son rectangulares.

(**) D indica el coeficiente de conducción. Se dan, para cada punto de discretización, dos valores de coeficiente de conducción: uno a nivel del fondo, que vale siempre cero, y otro a 10 m de altura respecto del correspondiente nivel del fondo, cuyo valor se indica en el cuadro 24. Los coeficientes de conducción intermedios, como siempre, se interpolan linealmente.

En primer lugar se llevó a cabo una corrida de estabilización, tomando como condiciones iniciales las indicadas en el cuadro 25. Las condiciones de contorno se indican en el cuadro 26.

Cuadro 25

Condiciones iniciales para corridas de estabilización de los modelos deltaicos

Pto	Q (m^3/s)	Z (m)	Pto	Q (m^3/s)	Z (m)	Pto	Q (m^3/s)	Z (m)
1	1000	110,60	17	1500	109,30	33	1500	109,34
2	1000	110,20	18	1500	109,20	34	1500	109,28
3	1000	109,80	19	1500	109,10	35	1500	109,22
4	1000	110,10	20	2500	109,10	36	1000	109,22
5	1000	109,85	21	2500	109,00	37	1000	109,16
6	1000	109,60	22	2500	108,90	38	1000	109,10
7	1000	110,00	23	1500	108,90	39	1000	108,90
8	1000	109,90	24	1500	108,80	40	1000	108,80
9	1000	109,80	25	1500	108,70	41	1000	108,70
10	2000	109,80	26	2500	108,70	42	500	109,22
11	2000	109,70	27	2500	108,60	43	500	109,14
12	2000	109,60	28	2500	108,50	44	500	109,06
13	3000	109,60	29	1800	108,50	45	700	108,50
14	3000	109,50	30	1800	108,40	46	700	108,35
15	3000	109,40	31	1800	108,30	47	700	108,20
16	1500	109,40	32	1500	109,40	48	700	108,05

Cuadro 26

Condiciones de contorno para corrida de estabilización del modelo deltaico (Δt=1 día)

Día	Q_1(m^3/s)	Q_4(m^3/s)	Q_7(m^3/s)	Z_{31}(m)	Z_{44}(m)	Z_{48}(m)
0	1000	1000	1000	108,30	109,06	108,05
30	1000	1000	1000	108,30	109,06	108,05

A partir de estas condiciones iniciales y de contorno originales se realizó una corrida de estabilización (las condiciones iniciales originales son sumamente inestables, y por eso el modelo

demora tanto en la estabilización). El estado estacionario obtenido se indica en el cuadro 27. Se usó θ=1 y el tiempo de computadora para cada intervalo de tiempo de cálculo es de alrededor de 1,24 segundos.

Cuadro 27

Estado estacionario del modelo deltaico

Punto	1	2	3	4	5	6	7	8
Z (m)	121,48	121,27	121,06	119,78	119,66	119,53	121,48	121,27
Q (m^3/s)	1000	1000	1000	1000	1000	1000	1000	1000
Punto	9	10	11	12	13	14	15	16
Z (m)	121,06	121,06	120,32	119,53	119,53	117,55	114,94	114,94
Q (m^3/s)	1000	2000	2000	2000	3000	3000	3000	1320
Punto	17	18	19	20	21	22	23	24
Z (m)	114,30	113,60	112,85	112,85	112,14	111,36	111,36	111,12
Q (m^3/s)	1320	1320	1320	1268	1268	1268	651	651
Punto	25	26	27	28	29	30	31	32
Z (m)	110,89	110,89	109,95	108,85	108,85	108,58	108,30	114,94
Q (m^3/s)	651	1268	1268	1268	588	588	588	1680
Punto	33	34	35	36	37	38	39	40
Z (m)	114,30	113,61	112,84	112,84	112,84	112,85	111,36	111,12
Q (m^3/s)	1680	1680	1680	-52	-52	-52	617	617
Punto	41	42	43	44	45	46	47	48
Z (m)	110,89	112,84	111,03	109,06	108,85	108,59	108,32	108,05
Q (m^3/s)	617	1732	1732	1732	680	680	680	680

A partir de estas condiciones estacionarias se llevaron a cabo diversos experimentos, que se detallan a continuación, y en los cuales se usó θ=0,85.

a) Caudal constante (igual al de la estabilización) ingresando aguas arriba y oscilaciones de altura (debidas, por ejemplo, a mareas), en los extremos aguas abajo. Las condiciones de contorno usadas se indican en el cuadro 28 para un día y se repiten (tienen período de un día) durante cuatro días. El intervalo Δt es de 6 horas.

Cuadro 28

Condiciones de contorno para análisis de influencia de mareas

Tiempo (días)	Punto 1 Q (m^3/s)	Punto 4 Q (m^3/s)	Punto 7 Q (m^3/s)	Punto 31 Z (m)	Punto 44 Z (m)	Punto 48 Z (m)
0	1000	1000	1000	108,30	109,06	108,05
0,25	1000	1000	1000	110,30	111,06	110,05
0,50	1000	1000	1000	108,30	109,06	108,05
0,75	1000	1000	1000	106,30	107,06	106,05
1	1000	1000	1000	108,30	109,06	108,05

En este experimento se produce el caso interesante de que de acuerdo a la marea el caudal en el tramo 36-38 cambia de signo (cabe recordar que las condiciones iniciales "poco factibles" indicadas en el cuadro 25 tenían asignados caudales positivos para ese tramo, y los caudales se convirtieron en negativos para las condiciones estacionarias; esto es un ejemplo de que a veces en tramos interiores de un delta complicado -y, por ejemplo, el delta del río Paraná es mucho más complicado que este ejemplo teórico- el real sentido de flujo del caudal no se conoce, y es crucial que las ecuaciones de Saint-Venant incluyan el signo - representado por el factor $Q|Q|$ en vez de Q^2- en el término no homogéneo para poder modelizar estos casos). En el cuadro 29 se indican los valores de caudales y niveles en el punto 37 para todos los intervalos temporales de cálculo.

Cuadro 29

Caudales (m^3/s) y niveles (m) en el punto 37 para corrida de mareas

(Tiempo en días)

t	0,00	0,25	0,50	0,75	1,00	1,25	1,50	1,75	2,00
Q	-52	26	-223	51	-60	40	-247	83	-82
Z	112.84	113.95	112.31	112.26	112.93	113.89	112.33	112.25	112.90

t		2,25	2,50	2,75	3,00	3,25	3,50	3,75	4,00
Q		59	-271	113	-104	77	-294	142	-123
Z		113,90	112,30	112,27	112,86	113,92	112,27	112,28	112,81

En el cuadro 30 se indica el estado del sistema en el tiempo t=4 días; puede observarse que los valores difieren muy poco de los iniciales, lo cual indica que estas condiciones iniciales son muy adecuadas:

Cuadro 30

Estado del sistema después de cuatro días de marea

Punto	1	2	3	4	5	6	7	8
Z (m)	121.41	121.20	120.99	119.72	119.59	119.47	121.40	121.19
Q (m^3/s)	1000	1001	1002	1000	1000	1001	1000	1001
Punto	9	10	11	12	13	14	15	16
Z (m)	120,99	120,99	120,25	119,47	119,47	117,50	114,90	114,90
Q (m^3/s)	1002	2004	2006	2008	3008	3009	3008	1293
Punto	17	18	19	20	21	22	23	24
Z (m)	114,26	113,58	112,83	112,83	112,10	111,29	111,29	111,05
Q (m^3/s)	1291	1288	1285	1160	1152	1142	594	587
Punto	25	26	27	28	29	30	31	32
Z (m)	110,81	110,81	109,88	108,81	108,81	108,55	108,30	114,90
Q (m^3/s)	579	1118	1103	1081	548	529	508	1715
Punto	33	34	35	36	37	38	39	40
Z (m)	114,26	113,57	112,80	112,80	112,81	112,83	111,29	111,05
Q (m^3/s)	1714	1712	1711	-122	-123	-125	548	544
Punto	41	42	43	44	45	46	47	48
Z (m)	110,81	112,80	111,15	109,06	108,81	108,56	108,30	108,05
Q (m^3/s)	539	1833	1831	1829	533	529	525	521

b) Para testear la consistencia del cambio de condiciones de contorno con un modelo más complicado que el esquema "cauce principal con una bifurcación" usado en la Sección anterior, se corrió nuevamente la situación de marea, pero ahora tomando como

condiciones de contorno en los tres extremos abiertos aguas arriba los niveles calculados por el modelo en la corrida anterior, y como condiciones de contorno en los tres extremos abiertos aguas abajo los caudales calculados por el modelo en la corrida anterior. Dichas condiciones de contorno se exhiben en el cuadro 31. Se tomó también Δt=6 horas.

Cuadro 31

Corrida de marea con condiciones de contorno intercambiadas

Tiempo (días)	Punto 1 Z (m)	Punto 4 Z (m)	Punto 7 Z (m)	Punto 31 Q (m^3/s)	Punto 44 Q (m^3/s)	Punto 48 Q (m^3/s)
0,00	121,48	119,78	121,48	588	1732	680
0,25	121,96	120,33	121,95	506	1600	642
0,50	121,03	119,30	121,02	631	2004	759
0,75	121,47	119,72	121,46	716	1586	688
1,00	121,47	119,79	121,46	508	1734	599
1,25	121,93	120,30	121,93	517	1588	685
1,50	121,05	119,32	121,04	635	2029	708
1,75	121,46	119,71	121,45	700	1550	753
2,00	121,46	119,78	121,45	510	1762	565
2,25	121,94	120,30	121,93	512	1562	712
2,50	121,04	119,30	121,03	644	2063	671
2,75	121,47	119,72	121,46	682	1509	806
3,00	121,44	119,75	121,43	511	1795	541
3,25	121,95	120,32	121,94	508	1534	733
3,50	121,01	119,28	121,00	647	2098	641
3,75	121,49	119,74	121,48	671	1466	851
4,00	121,41	119,72	121,40	508	1829	521

Los resultados reproducen los de la corrida anterior con una precisión notable: en el cuadro 32 se indican los valores complementarios obtenidos en los extremos (caudales donde se dan niveles y niveles donde se dan caudales). Comparados con los valores del cuadro 28, que son los tomados en la corrida original como condiciones de contorno, la precisión es muy alta: en todos

los casos el error no llega al 1% ni en los caudales ni en los niveles (para calcular el error en este caso hay que considerar las alturas desde el fondo del cauce):

Cuadro 32

Condiciones complementarias a las de contorno en los extremos

Tiempo (días)	Punto 1 Q (m^3/s)	Punto 4 Q (m^3/s)	Punto 7 Q (m^3/s)	Punto 31 Z (m)	Punto 44 Z (m)	Punto 48 Z (m)
0,00	1000	1000	1000	108,30	109,06	108,05
0,25	1000	1000	1000	110,30	111,07	110,05
0,50	1000	1003	998	108,30	109,07	108,05
0,75	1001	998	1001	106,30	107,06	106,05
1,00	1001	1003	998	108,30	109,06	108,05
1,25	990	1001	1008	110,30	111,06	110,05
1,50	1008	999	993	108,30	109,06	108,05
1,75	995	1001	1005	106,30	107,07	106,05
2,00	1005	1001	993	108,31	109,06	108,06
2,25	998	998	1004	110,30	111,06	110,05
2,50	1006	996	998	108,29	109,06	108,04
2,75	997	1002	1001	106,29	107,05	106,04
3,00	1005	996	999	108,29	109,05	108,04
3,25	995	1009	997	110,30	111,06	110,05
3,50	1004	996	1001	108,31	109,07	108,06
3,75	998	1003	999	106,31	107,08	106,06
4,00	1004	995	1001	108,31	109,08	108,06

Para comparar los estados de ambos modelos en el tiempo 4 se indican en el cuadro 33 los valores obtenidos en esta corrida:

Cuadro 33

Estado del modelo a los cuatro días con condiciones de contorno intercambiadas

Punto	1	2	3	4	5	6	7	8
Z (m)	121,41	121,20	120,99	119,72	119,59	119,47	121,40	121,20
Q (m^3/s)	1004	1005	1006	995	996	996	1001	1002
Punto	9	10	11	12	13	14	15	16
Z (m)	120,99	120,99	120,25	119,47	119,47	117,50	114,90	114,90
Q(m^3/s)	1003	2008	2010	2012	3008	3009	3008	1293
Punto	17	18	19	20	21	22	23	24
Z (m)	114,26	113,58	112,83	112,83	112,10	111,29	111,29	111,06
Q (m^3/s)	1291	1288	1285	1160	1152	1142	593	587
Punto	25	26	27	28	29	30	31	32
Z (m)	110,82	110,82	109,89	108,82	108,82	108,55	108,31	114,90
Q (m^3/s)	579	1118	1103	1081	548	529	508	1715
Punto	33	34	35	36	37	38	39	40
Z (m)	114,26	113,57	112,80	112,80	112,82	112,83	111,29	111,06
Q (m^3/s)	1714	1712	1710	-122	-123	-125	548	544
Punto	41	42	43	44	45	46	47	48
Z (m)	110,82	112,80	111,15	109,08	108,82	108,56	108,31	108,06
Q (m^3/s)	539	1832	1831	1829	533	529	525	521

c) El experimento siguiente consiste en cerrar los extremos 1, 4, 44 y 48, de modo que el flujo es en un tramo simple (con dos bifurcaciones alrededor de islas). Esta simulación es útil cuando se quiere analizar la posibilidad de elegir en un delta un tramo principal de río, y eventualmente rectificarlo. Las condiciones de contorno son las indicadas en el cuadro 34 (puede observarse que en este experimento se usan como condiciones de contorno aguas abajo niveles y caudales). El paso de tiempo fue de 1 día.

Cuadro 34

Condiciones de contorno para corrida de cierre de extremos abiertos.

Tiempo (días)	Punto 1 Q (m^3/s)	Punto 4 Q (m^3/s)	Punto 7 Q (m^3/s)	Punto 31 Z (m)	Punto 44 Q (m^3/s)	Punto 48 Q (m^3/s)
0	1000	1000	1000	108.30	1732	680
10	0	0	3000	111.30	0	0
20	0	0	3000	111.30	0	0

El modelo entra rápidamente en estado estacionario: el cierre finaliza el día 10, y ya el día 13 los resultados difieren de los indicados en el cuadro 35 (día 16) en a lo sumo 1 m^3/s o 1 cm. Obsérvese el remanso muerto que se forma en los tramos 1-3, 4-6, 42-44 y 45-48 en el día 10 (cuadro 36):

Cuadro 35

Modelo estabilizado con sólo un extremo abierto aguas arriba y otro aguas abajo

Punto	1	2	3	4	5	6	7	8
Z (m)	128,92	128,92	128,92	127,10	127,10	127,10	130,83	129,90
Q (m^3/s)	0	0	0	0	0	0	3000	3000
Punto	9	10	11	12	13	14	15	16
Z (m)	128,92	128,92	128,04	127,10	127,10	126,15	125,14	125,14
Q (m^3/s)	3000	3000	3000	3000	3000	3000	3000	1525
Punto	17	18	19	20	21	22	23	24
Z (m)	124,84	124,53	124,22	124,22	123,04	121,74	121,74	121,35
Q (m^3/s)	1525	1525	1525	3000	3000	3000	1540	1540
Punto	25	26	27	28	29	30	31	32
Z (m)	120,95	120,95	119,39	117,56	117,56	115,12	111,30	125,14
Q (m^3/s)	1540	3000	3000	3000	3000	3000	3000	1475
Punto	33	34	35	36	37	38	39	40
Z (m)	124,97	124,80	124,62	124,62	124,42	124,22	121,74	121,35
Q (m^3/s)	1475	1475	1475	1475	1475	1475	1460	1460
Punto	41	42	43	44	45	46	47	48
Z (m)	120,95	124,62	124,62	124,62	117,56	117,56	117,56	117,56
Q (m^3/s)	1460	0	0	0	0	0	0	0

Cuadro 36

Estado del modelo inmediatamente después del cierre (día 10)

Punto	1	2	3	4	5	6	7	8
Z (m)	128,70	128,70	128,70	126,86	126,86	126,86	130,59	129,67
Q (m^3/s)	0	-1	-2	0	-1	-1	3000	2999
Punto	9	10	11	12	13	14	15	16
Z (m)	128,70	128,70	127,81	126,86	126,86	125,88	124,83	124,83
Q (m^3/s)	2998	2996	2994	2992	2991	2987	2984	1516
Punto	17	18	19	20	21	22	23	24
Z (m)	124,53	124,21	123,90	123,90	122,72	121,42	121,42	121,03
Q (m^3/s)	1514	1512	1511	2973	2970	2967	1523	1521
Punto	25	26	27	28	29	30	31	32
Z (m)	120,63	120,63	119,08	117,25	117,25	114,87	111,30	124,83
Q (m^3/s)	1520	2961	2958	2956	2955	2953	2952	1468
Punto	33	34	35	36	37	38	39	40
Z (m)	124,65	124,47	124,29	124,29	124,09	123,90	121,42	121,03
Q (m^3/s)	1467	1466	1465	1464	1463	1462	1444	1443
Punto	41	42	43	44	45	46	47	48
Z (m)	120,63	124,29	124,29	124,29	117,25	117,25	117,25	117,25
Q (m^3/s)	1442	1	0	0	1	1	0	0

Usando el estado del sistema con extremos cerrados (cuadro 35) como condición inicial, se simuló entonces una operación normal de una represa aguas arriba (punto 7) con nivel fijo aguas abajo (debido a la existencia, por ejemplo, de otro embalse en el punto 31) y con extracción de caudal (para riego, por ejemplo) en el punto 44. Las condiciones de contorno se indican en el cuadro 37, y tienen período diario. El paso de tiempo fue de 6 horas.

Cuadro 37

Funcionamiento diario de un sistema regulado

Tiempo (días)	Punto 1 Q (m^3/s)	Punto 4 Q (m^3/s)	Punto 7 Q (m^3/s)	Punto 31 Z (m)	Punto 44 Q (m^3/s)	Punto 48 Q (m^3/s)
0	0	0	3000	111,30	0	0
0,25	0	0	5000	111,30	500	0
0,75	0	0	5000	111,30	500	0
1	0	0	3000	111,30	0	0

Se puede ver en el cuadro 38 que los cambios de caudal son absorbidos naturalmente por el modelo, notándose una gran sensibilidad de los niveles a dichos cambios. En particular, los remansos en las ramas "muertas" suben y bajan en función del nivel en las respectivas desembocaduras, manteniéndose cada remanso con nivel uniforme (altura del agua horizontal, tipo efecto "lago"). El nivel fijo en la sección 31 implica un considerable aumento de la velocidad V, como se indica.

Cuadro 38

Resultados de corrida con sistema regulado

Tiempo (días)	Z_1 (m)	Z_4 (m)	Z_7 (m)	Q_{31} (m^3/s)	V_{31} (m/s)	Z_{44} (m)	Z_{48} (m)
0	128,92	127,10	130,83	3000	1,28	124,62	117,56
0,25	134,44	131,60	137,43	3731	1,59	127,76	119,13
0,50	137,89	134,75	141,19	4028	1,72	130,28	120,97
0,75	138,42	135,53	141,37	4374	1,87	131,35	121,90
1,00	133,69	131,72	135,69	3655	1,56	128,87	120,67
1,25	135,31	132,91	137,74	4220	1,80	129,51	120,69
1,50	138,58	135,49	141,86	4108	1,76	131,12	121,75
1,75	138,49	135,66	141,39	4502	1,92	131,55	122,12
2,00	133,85	131,89	135,85	3634	1,55	129,05	120,82
2,25	135,25	132,86	137,66	4269	1,82	129,48	120,69
2,50	138,63	135,54	141,91	4083	1,74	131,17	121,79
2,75	138,46	135,62	141,35	4524	1,93	131,52	122,10
3,00	133,87	131,91	135,87	3619	1,55	129,07	120,83

d) Por último, se realizó, con las mismas condiciones iniciales de antes, la simulación de una crecida importante en un sistema con control en todos los extremos, aguas arriba y aguas abajo. En las primeras doce horas el caudal aguas arriba crece linealmente de 3000 a 6000 m^3/s, para lo cual el caudal del punto 1 pasa de 0 a 3000 m^3/s, pues se considera que la capacidad de ingreso de caudal (controlable) esta colmada en el punto 7 con los 3000 m^3/s que ya fluyen. En las segundas doce horas del día el caudal crece linealmente de 6000 a 9000 m^3/s, y ahora se "abre" el punto 4, pues se considera agotada también la capacidad de flujo controlable a través del punto 4. El nivel en el punto 31 se mantiene constante (se supone que mantener ese nivel es una consigna), y se permite el paso de caudal por el punto 44 y, cuando el caudal de entrada supera los 3000 m^3/s, por el punto 48, en este caso hasta 1000 m^3/s. Con estos caudales se opera un día más. Entre el segundo y el tercer día se vuelve a la situación inicial, pero ahora decreciendo el flujo en los puntos 1 y 4 en forma simultánea, y se mantiene esta situación un día más. Esto queda indicado en el cuadro 39:

Cuadro 39

Condiciones de contorno para crecida importante

Tiempo (días)	Q_1 (m^3/s)	Q_4 (m^3/s)	Q_7 (m^3/s)	Z_{31} (m)	Q_{44} (m^3/s)	Q_{48} (m^3/s)
0	0	0	3000	111,30	0	0
0,50	3000	0	3000	111,30	3000	0
1,00	3000	3000	3000	111,30	5000	1000
2,00	3000	3000	3000	111,30	5000	1000
3,00	0	0	3000	111,30	0	0
4,00	0	0	3000	111,30	0	0

Los resultados de la corrida se indican en el cuadro 40. Puede observarse lo siguiente:

i) Durante un corto período el caudal cambia de sentido en el tramo entre los puntos 37 y 39, debido a la brusquedad de la crecida.

ii) El considerable caudal que llega a pasar por el punto 44 lleva la velocidad a valores muy altos, pero sin que por ello el régimen deje de ser subcrítico. Si ésta fuera una simulación de un caso real, debería probablemente pensarse en otro tipo de distribución de caudales aguas abajo.

iii) Los niveles en los puntos 1 y 4 son extremadamente sensibles a los cambios de caudal: pasar de caudal nulo a 3000 m^3/s cambia el nivel casi 17 m en el punto 1 y 14 m en el punto 4.

iv) El sistema se reestabiliza rápidamente: el día 4 los valores de las variables ya son muy similares a los iniciales, pese a que hace apenas un día que las condiciones de contorno no varían.

Cuadro 40

Resultados de la corrida con crecida importante (Δt=6 horas)

Tiempo (días)	Z_1 (m)	Z_4 (m)	Z_7 (m)	Q_{31} (m^3/s)	V_{31} (m/s)	Q_{38} (m^3/s)	Z_{44} (m)	V_{44} (m/s)	Z_{48} (m)
0	128,92	127,10	130,83	3000	1,28	1475	124,62	0,00	117,56
0,25	131,07	128,24	132,79	2926	1,25	992	124,15	1,20	117,40
0,50	136,68	131,09	137,05	2817	1,20	206	122,12	2,60	117,06
0,75	140,62	135,13	140,55	2687	1,15	77	120,26	3,77	116,67
1,00	143,86	139,91	143,87	2596	1,11	-178	118,99	5,02	116,04
1,25	145,14	141,68	145,13	2803	1,20	-5	118,28	5,20	116,10
1,50	145,28	141,80	145,27	2924	1,25	12	119,74	4,84	116,74
1,75	145,38	141,92	145,37	2976	1,27	51	120,62	4,64	116,95
2,00	145,38	141,93	145,37	2992	1,28	31	120,91	4,58	117,03
2,25	142,84	139,71	143,09	3156	1,35	551	123,12	3,12	117,55
2,50	138,84	136,16	139,53	3256	1,39	960	125,78	1,87	118,20
2,75	134,92	132,65	136,06	3290	1,41	1329	126,38	0,91	118,45
3,00	131,09	129,28	132,66	3293	1,41	1621	126,28	0,00	118,53
3,25	129,56	127,84	131,36	3157	1,35	1508	125,49	0,00	118,22
3,50	129,22	127,43	131,09	3073	1,31	1510	124,97	0,00	117,84
3,75	129,04	127,23	130,92	3030	1,29	1469	124,78	0,00	117,68
4,00	128,97	127,16	130,87	3014	1,29	1490	124,69	0,00	117,61

19. CONCLUSIONES

Se ha probado que:

a) Las redes fluviales unidimensionales con estructura arborescente, cualquiera sea su complejidad, se pueden modelizar con un método implícito de primer orden con la misma eficiencia, en tiempo de cálculo y memoria de computadora disponible, que los modelos de tramos simples con igual número de puntos de discretización. En efecto, la memoria necesaria, tanto para tramos simples como para redes arborescentes, es de alrededor de 6n palabras para almacenar los sucesivos **A** y **b** de los sistemas **Ax=b** que es necesario resolver en cada paso de tiempo, y la cantidad de operaciones es de alrededor de 18n operaciones. Los experimentos numéricos con modelos simplificados para los cuales hay solución analítica dieron resultados con el mismo grado de precisión que los realizados para tramos simples; se exhiben también numerosos experimentos con redes fluviales arborescentes similares a las que se pueden encontrar en la realidad.

b) Las redes fluviales unidimensionales deltaicas, cualquiera sea su complejidad,se pueden modelizar con un método implícito de primer orden con una eficiencia que, si bien es menor que en a), es de todos modos muy aceptable. En efecto, la memoria necesaria para almacenar la matriz **A** y el vector **b** es del orden de $6n+t^2$; como t es bastante menor que n, eso significa una sensible disminución de la memoria necesaria para almacenar el sistema completo, que es n^2. Y en cuanto al número de operaciones, es del orden de $28n + t^3/3$, que también indica una mejoría considerable respecto de los $n^3/3$ necesarios para el sistema completo. También se realizaron experimentos con modelos simplificados de los cuales se conocen soluciones analíticas, con resultados similares a los anteriores, y experimentos con redes fluviales complejas análogas a redes reales.

c) Se hicieron estudios heurísticos de la estabilidad numérica de la matriz **A**, que muestran su buen condicionamiento en circunstancias habituales; se planteó figurosamente el esquema de Preissmann en el marco de la teoría de Lax y Richtmyer; en el

apéndice 1 se muestra una implementación del método iterativo de las proyecciones de Kaczmarz, de convergencia asegurada, y una modificación del mismo; en este caso, los resultados obtenidos obligan a desaconsejar el uso de estos métodos en la resolución numérica de las ecuaciones de aguas poco profundas.

d) Los programas computacionales elaborados, que se adjuntan, no son programas válidos solamente para experimentos numéricos teóricos: son programas elaborados, para uso inmediato en modelos reales, tanto para ajuste y validación de los mismos como para su explotación numérica.

Las investigaciones realizadas se pueden proseguir en varias direcciones; por ejemplo:

a) Modelos de redes fluviales con fondo móvil. Para esto hay que analizar, desde el punto de vista matemático y también desde el punto de vista físico, las ecuaciones de compatibilidad de los niveles del fondo del lecho, y su discretización numérica.

b) Modelos de redes fluviales en régimen supercrítico. Para esto parecería en primera instancia necesario asegurarse de que el número de condiciones de contorno no esté sobredimensionado: el número de extremos abiertos aguas abajo (en los cuales no se da ninguna condición de contorno) debería ser igual al de extremos aguas arriba (en los cuales se dan dos condiciones de contorno) o debería valer alguna restricción equivalente. Probablemente la condición de Stoker (12.2) deba ser reemplazada por las ecuaciones (13), pues no se pueda aceptar, en régimen supercrítico, la simplificación (12.2).

c) Propagación de discontinuidades en redes fluviales (por ejemplo, rotura brusca de un dique). Este enfoque incluye, como caso particular pero muy importante y significativo, el análisis del problema de Riemann para redes fluviales.

d) Modelos de red fluvial con cambio de régimen (subcrítico a supercrítico o supercrítico a subcrítico) durante la simulación, en algunos tramos. Uno de los problemas importantes en estos

modelos es la asignación dinámica de condiciones de contorno: en el momento en que hay un cambio de régimen, deberá haber un cambio de condiciones de contorno.

f) Modelos de red fluvial en donde hay un "cruce" de canales, es decir, donde hay 4 puntos i,j,k,l, en que valen las condiciones ampliadas

$$Z_i = Z_j = Z_k = Z_l$$

$$sg(i)Q_i + sg(j)Q_j + sg(k)Q_k + sg(l)Q_l = 0$$

donde sg(i), sg(j), sg(k), sg(l) vale más uno o menos uno dependiendo del sentido que se asigna al flujo en cada uno de los cuatro cauces que confluyen (ver figura 23). Estos cruces se pueden ver, por ejemplo, en ciertas zonas del delta del río Paraná; en particular, este autor diseñó e implementó en 1977 un modelo que fue aplicado al estudio de un tramo del río Paraná de las Palmas por un grupo de trabajo dirigido por M. Gradowczyk. Los experimentos realizados, sin datos ajustados, permitieron detectar inconsistencias en los ceros de los hidrómetros, que fueron ratificados después en una campaña ad-hoc. El modelo no tiene optimización de tiempo de cálculo ni de memoria.

g) Análisis de otros algoritmos de resolución eficiente del sistema Ax=b . Los algoritmos diseñados en este trabajo se basan en el estudio de las redes fluviales y su estructura como grafos, es decir, la matriz A se arma en función de una determinada distribución de los puntos de discretización sobre la red fluvial. Es interesante probar algoritmos que arman la matriz A en forma automática - por ejemplo, el algoritmo de Cuthill-McKee revertido o métodos de disección anidada - tratando de que sea lo más eficiente posible y sin saber de donde proviene la matriz; al respecto, puede verse, por ejemplo, el libro de George y Liu [1981], con la salvedad de que sus algoritmos deben adaptarse a matrices no simétricas. Esto no es un inconveniente grave, pues no es necesario buscar otro pivote que el diagonal en las eliminaciones de Gauss que se utilizan.

APENDICES

APENDICE 1

ANALISIS DE UN METODO ITERATIVO (DE LAS PROYECCIONES)

A1.1. EL METODO DE LAS PROYECCIONES

El método de las proyecciones fue ideado por Kaczmarz en la década del treinta, y permaneció más o menos ignorado hasta que comenzó a ser usado en aplicaciones del álgebra lineal a tomografía computada (la llamada "técnica de reconstrucción algebraica" o ART - algebraic reconstruction technique), y se extendió a aplicaciones en optimización. Una descripción muy completa de generalizaciones del método de las proyecciones, los llamados "métodos de acción sobre filas" (row action methods) puede verse en Censor [1981]. En cuanto al método iterativo de Kaczmarz propiamente dicho, su idea es muy simple: dada una solución inicial x_0 del sistema $Ax = b$, la siguiente aproximación x_1 se obtiene proyectando la primera aproximación sobre un hiperplano $a_i x_i = b_i$, donde a_i es la i-sima fila de la matriz A, y así sucesivamente, eligiendo de alguna manera las filas i para la proyección en los hiperplanos que determinan, de modo que al terminar un ciclo (de n iteraciones, si la matriz es de orden n) se haya proyectado sobre todos los hiperplanos. Una demostración (muy elegante, por cierto), de la convergencia de este algoritmo puede verse en el libro de Gastinel [1966][4].

En la implementación del algoritmo de Kaczmarz para los experimentos que hemos realizado se ha compactado la matriz A mediante un sistema de punteros eficiente y de implementación sencilla, que además se presta naturalmente para su uso en estos algoritmos de acción por filas , y se describirá más adelante. Se podrá observar que la lentitud del método hace impracticable el uso del mismo, excepto con propósitos de investigación teórica. Tampoco mejorará la situación mediante una modificacion sugerida por el tipo de ecuaciones de la matriz A correspondiente a la discretización de Preissmann de las ecuaciones de Saint-Venant: si se consideran, en lugar de los hiperplanos determinados por cada fila de la matriz A, los subespacios de dimensión n-2 determinados

por los *pares* de hiperplanos correspondientes a la discretización de dicho par de ecuaciones de Saint-Venant en el intervalo entre puntos de discretización contiguos, se podría tal vez esperar que estos subespacios estén más "separados" (angularmente) uno de otro, y si el tiempo necesario para calcular la intersección fuera pequeño en comparación con la iteración de un hiperplano al par, se podría reducir el tiempo de iteraciones hasta casi la mitad (pues ahora se harían n/2 iteraciones en vez de n). Los experimentos realizados indican que también esta modificación debe rechazarse: no se nota ninguna mejoría.

Pasemos entonces a analizar los métodos de las proyecciones.

Sea $Ax=b$ el sistema que se quiere resolver. Sea a_i la i-sima fila de la matriz A (de orden n), y sea $b=(b_1,\ldots,b_n)$ el lado derecho de la ecuación. Entonces, dada una aproximación inicial $x^{(0)}$ de la solución del sistema, la iteración típica será

$$x^{(k)} = x^{(k-1)} + \frac{b_{i_k} - \langle a_{i_k}, x^{(k-1)} \rangle}{\| a_{i_k} \|^2} a_{i_k}$$

donde

$i_k \in [1,\ldots,n]$

a_{i_k} es la i_k-sima fila de la matriz cuadrada A

$\langle , \rangle$ indica el producto escalar en $\mathbb{R}^n$

$\| a \|^2 = \langle a,a \rangle$

$x^{(k)}$ será la proyección de $x^{(k-1)}$ sobre el hiperplano de ecuación $a_{i_k}.x = b_{i_k}$.

La sucesión de índices $\{i_k\}_{k=0}^{\infty}$ según la cual se eligen las filas de la matriz A se denomina sucesión de control (en el paso iterativo k → k+1 se usa la fila i_k). La sucesión de control más natural es por supuesto $i_k = k \pmod n + 1$ (control cíclico), que es la que hemos utilizado. Se puede observar que los $\|a_{i_k}\|^2$ se calculan una sola vez cada uno y, si la matriz A es rala (que es justamente el caso que interesa) son muy pocas las coordenadas en que $x^{(k)}$ diferirá de $x^{(k-1)}$. Por consiguiente es engañoso terminar la iteración cuando dos iteraciones sucesivas difieren entre sí en menos de un ε dado; es más razonable asegurarse de que la diferencia es pequeña después de que se modificaron todas las coordenadas de x, y por consiguiente se toma como criterio para terminar la iteración el de que

$$\| x^{n(k+1)} - x^{nk} \| \leq \varepsilon$$

para un ε dado suficientemente pequeño.

La matriz A se representa por medio del siguiente esquema de punteros, que puede verse en la amplia reseña de Pooch y Nieder [1973] de descripción de matrices ralas mediante punteros. Es "orientado por filas", como corresponde a este caso, y de fácil y eficiente implementación:

Bastan tres vectores: $v \in \mathbb{N}^n$, $w \in \mathbb{N}^J$, y $z \in \mathbb{R}^J$, donde J indica el número total de elementos no nulos de la matriz A, y cuyos significados son los siguientes:

v[i] es el número de elementos no nulos de la fila i.

w[j] es la ubicación del j-simo elemento no nulo de la matriz A en la fila a la cual pertenece, es decir en la fila k tal que

$$\Sigma_{i<k} v[i] < j \qquad \Sigma_{i\leq k} v[i] \geq j$$

z[j] es el valor del elemento no nulo correspondiente.

La matriz A se construirá del siguiente modo: la incógnita ΔQ_j, cuando v_j pertenece al nodo de confluencia $\{v_{j-1}, v_k, v_j\}$ se reemplaza por $\Delta Q_{j-1}+\Delta Q_k$ (si se trata de un nodo de afluencia),o por $\Delta Q_{j-1}-\Delta Q_k$ (si se trata de un nodo de efluencia). Los niveles ΔZ_j y ΔZ_k se reemplazan por ΔZ_{j-1}.

De tal modo, si n es el número total de puntos, t el de nodos de confluencias, y s el de extremos abiertos, el orden de la matriz A será

$$2n-s-3t$$

Es decir, no se introducen entre las ecuaciones de la matriz las ecuaciones

$$\Delta Q_{i-1} \pm \Delta Q_k = \Delta Q_i$$

$$\Delta Z_{i-1} = \Delta Z_i = \Delta Z_k$$

pues se usan directamente para disminuir el tamaño de la matriz A. Interesa (y en particular será necesario en el uso del método modificado de Kacmarcz, que se verá en el parágrafo siguiente) asegurarse de que el orden de A es par. Sea ahora m el número total de tramos simples. Demostraremos por tanto el siguiente

LEMA: s es par si y sólo si t es par.

DEMOSTRACION: El número total de todos los puntos extremos de los tramos de la red fluvial es s+3t (pues a cada punto de confluencia corresponden tres puntos extremos cerrados), y éste es un número par pues hay dos puntos extremos por tramo simple: s+3t=2m. (Este resultado ya se vio en la sección 15). Como la paridad de t coincide con la de 3t, queda demostrado. O sea, el orden de la matriz es 2(n-m). ■

En la programación de la estructura de punteros de A, así como en la del método de Kacmarcz, usaremos los conocimientos que tenemos de la estructura de A, y que podemos resumir en el siguiente

LEMA: Cada fila de A tiene tres, cuatro o cinco elementos distintos de cero. Cuando una fila tiene tres elementos distintos de cero, son contiguos. Si tiene cuatro elementos distintos de cero, al menos tres de ellos son contiguos. Y si tiene cinco elementos distintos de cero, exactamente uno es aislado (no contiguo de ninguno de los otros cuatro). Además, las filas tienen esa estructura de a pares.

DEMOSTRACION:

Las filas de tres elementos corresponden a intervalos (i,i+1) para los cuales i o i+1 es un punto extremo abierto (y por consiguiente una de las incógnitas ΔQ_i, ΔZ_i, ΔQ_{i+1}, ΔZ_{i+1}, con su correspondiente coeficiente, pasa al lado derecho). Queda una fila de la forma

$$A_{2i}\Delta Z_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1}=A_{5i}-A_{1i}\Delta Q_i$$

$$A_{1i}\Delta Q_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1}=A_{5i}-A_{2i}\Delta Z_i$$

$$A_{1i}\Delta Q_i+A_{2i}\Delta Z_i \quad +A_{4i}\Delta Z_{i+1}=A_{5i}-A_{3i}\Delta Q_{i+1}$$

$$A_{1i}\Delta Q_i+A_{2i}\Delta Z_i \quad +A_{3i}\Delta Q_{i+1}=A_{5i}-A_{4i}\Delta Z_{i+1}$$

según sea conocido como condición de contorno el caudal en i, el nivel en i, el caudal en i+1 o el nivel en i+1, respectivamente. Y análogamente otra fila con los coeficientes $B_{k,i}$, k=1,..5.

Las de cuatro elementos, si son contiguos, corresponden a intervalos (i,i+1) interiores de un tramo simple, de la forma

$$A_{1i}\Delta Q_i+A_{2i}\Delta Z_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1}=A_{5i}$$

y si alguno no lo es, a intervalos (i,i+1) en los cuales uno de los puntos es un extremo cerrado, y el otro un punto interior.

Si el extremo cerrado es i, se tratará del primer intervalo de un tramo efluente, y la fila será

$$A_{2i}\Delta Z_{j-1}+ \quad A_{1i}\Delta Q_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \quad =A_{5i} \quad \text{o}$$

$$A_{1i}\Delta Q_i+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \quad +A_{2i}\Delta Z_{j-1}=A_{5i}$$

donde el punto de confluencia al cual pertenece i es {j-1,i,j}, según sea en la numeración j<i o i<j. El modelo no admite, para evitar complicaciones computacionales relacionadas con un caso muy inusual, la posibilidad de que i e i+1 sean extremos cerrados, (ni uno cerrado y otro abierto, es decir que no admite tramos simples de solamente dos puntos de discretización). Análogamente, si el extremo cerrado es i+1, la fila será

$$A_{1i}\Delta Q_i+A_{2i}\Delta Z_i+A_{3i}\Delta Q_{i+1} \qquad +A_{4i}\Delta Z_{j-1}=A_{5i} \qquad \text{o}$$

$$A_{4i}\Delta Z_{j-1}+ \qquad A_{1i}\Delta Q_i+A_{2i}\Delta Z_i+A_{3i}\Delta Q_{i+1} \qquad =A_{5i}$$

donde el punto de confluencia al cual pertenece i+1 es {j-1,i+1,j}, según sea en la numeración j>i+1 o j<i+1. Lo mismo vale, por supuesto, para la fila con los $B_{k,i}$, k=1,...,5.

Finalmente, las filas con cinco elementos distintos de cero tienen siempre un elemento aislado y cuatro elementos contiguos. Son las correspondientes a los tramos (i,i+1) tales que i es un extremo cerrado. Son de la forma

$$A_{1i}\Delta Q_j+ \qquad A_{1i}\Delta Q_{i-1}+A_{2i}\Delta Z_{i-1}+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \qquad =A_{5i}$$

$$A_{1i}\Delta Q_{i-1}+A_{2i}\Delta Z_{i-1}+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \qquad +A_{1i}\Delta Q_j \quad =A_{5i}$$

según j<i o j>i, respectivamente, donde j es el último punto de discretización del tramo afluente cuya confluencia con el tramo principal es el nodo de afluencia {i-1,j,i}, o

$$-A_{1i}\Delta Q_j+ \qquad A_{1i}\Delta Q_{i-1}+A_{2i}\Delta Z_{i-1}+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \qquad =A_{5i}$$

$$A_{1i}\Delta Q_{i-1}+A_{2i}\Delta Z_{i-1}+A_{3i}\Delta Q_{i+1}+A_{4i}\Delta Z_{i+1} \qquad -A_{1i}\Delta Q_j \quad =A_{5i}$$

según j<i o j>i, respectivamente, donde j es el primer punto de discretización del tramo efluente cuya confluencia con el tramo principal es el nodo de efluencia {i-1,j,i}. Naturalmente, existe la ecuación análoga con los B_{ki}, k=1,...5.

A1.2. EXPERIMENTOS CON EL METODO DE LAS PROYECCIONES

Para comparar los tiempos de cálculo del método de Kaczmarz con los de los otros métodos usados, se hicieron dos tipos de prueba: se corrió el modelo con la red arborescente de la sección 11, para comparar este método con el usado en la primera parte de este trabajo, y luego se corrió con la red fluvial indicada en la figura 22 y descripta en el cuadro 24, para comparar los distintos métodos usados para redes deltaicas.

Para la corrida con la red fluvial arborescente de 14 puntos se usó un límite de 32.000 iteraciones. En primer lugar se realizó una corrida de estabilización con ε=0,001, partiendo de las condiciones iniciales dadas en el cuadro 8. Esta corrida no convergió, es decir, para 32000 iteraciones todavía el error entre dos ciclos de iteraciones es mayor que 0,001. Para realizar las 32000 iteraciones el modelo empleó 3 minutos, aproximadamente. Esto es ya muy preocupante, pues 0,001 no es una exigencia desmedida, y 3 minutos es mucho tiempo para un intervalo de cálculo. Se decidió entonces testear la bondad de los resultados con un ε menor; se tomó ε=0,01 y se volvió a correr el modelo para estabilizarlo. En este caso el promedio por intervalo de tiempo de cálculo fue de 3 segundos, para una corrida de 20 días con un intervalo de cálculo de 1 día (es importante indicar cuántos días se corrieron, pues en esta y otras corridas se notó que puede haber cambios muy bruscos en la cantidad de iteraciones necesarias para resolver el sistema, sin que haya ningún cambio especial de las condiciones de contorno que lo justifique). Se obtuvieron los resultados indicados en el cuadro 41, que, comparados con los valores estacionarios del cuadro 9 indican excesiva tolerancia, pues no se llegó aún al estado estacionario:

Cuadro 41

Aproximación al estado estacionario en 20 días con ε=0,01

Punto	1	2	3	4	5	6	7	8
Cota (m)	112,28	110,32	108,46	106,61	112,86	111,57	110,29	109,10
Punto	9	10	11	12	13	14		
Cota (m)	107,90	106,61	106,61	104,98	102,90	100,00		

Cabe observar que en un método iterativo los tiempos de cálculo pueden variar, dependiendo de la bondad de la iteración inicial en cada paso de tiempo. Así, puede convenir disminuir el paso temporal Δt, para tener un valor inicial para la iteración siguiente "más cercano" a la solución (pues se supone que en "menos tiempo" la solución se modificó menos). Eso se comprueba, realizando, por ejemplo, la corrida con las condiciones iniciales estacionarias indicadas en el cuadro 9, y las condiciones de contorno de la corrida 1 indicada en el cuadro 12: con Δt=1 día no hay convergencia (para el límite de iteraciones dado) pero con Δt=0,25 de días se obtienen resultados con poca precisión, pero con un tiempo medio por paso de tiempo de 15 segundos por paso de tiempo Δt. Si se reduce ahora el intervalo Δt a 0,1 de día, el paso de tiempo necesario para obtener resultados para los 18 días del experimento se reduce en promedio a alrededor de 3 segundos.

Intuitivamente, el tomar Δt menor indica que la variación de resultados entre dos pasos de tiempo sucesivos será menor, y por consiguiente, tomar el resultado del instante anterior como iteración inicial es tomar una mejor aproximación inicial. Pero esto sirve cuando estamos en una situación "hidrodinámicamente aceptable", lo cual muchas veces no es el caso cuando se parte de condiciones iniciales no conocidas perfectamente, y se quiere llegar a partir de ellas a situaciones aceptables (en general, a regímenes estacionarios). Además, no hemos podido resolver el problema de la falta de precisión de los resultados: para Δt=0,1 día, y ε=0,001 el modelo no converge de entrada para la corrida con las condiciones de contorno del cuadro 12. Y por otra parte,

las variaciones bruscas en el número de iteraciones necesarias impiden estimar tiempos medios de corridas cortas con suficiente precisión, lo que añade incertidumbre al método.

Por consiguiente, queda claro que este método no es para nada recomendable: en algunos casos es totalmente impracticable, en otros debería ser necesario cambiar la tolerancia en una misma corrida, en otros la tolerancia posible permite muy poca precisión, etc. En principio, el método es muy lento, y no parece haber mayores ventajas en usarlo respecto de métodos explícitos.

Para los restantes experimentos se usó la red fluvial definida en el cuadro 24 e indicada en la figura 22. En primer lugar, se trató de estabilizar el modelo a partir de las condiciones iniciales del cuadro 25 con las condiciones de contorno del cuadro 26.

Se pudo comprobar lo siguiente: si la tolerancia ε utilizada para interrumpir la iteración en cada paso de tiempo es de 0,02, el método de las proyecciones es tan lento que supera la cantidad máxima de iteraciones empleada (32000). Pasando a una tolerancia ε de 0,05, se llega a valores estabilizados, que resultan inadmisibles por lo cual no puede aceptarse como aproximación.

Considerando como mínima tolerancia aceptable ε=0,001, se decidió evaluar si se podía estabilizar el modelo (como ya se indicó, las condiciones iniciales son muy poco satisfactorias, como se puede comprobar observando la lentitud de la convergencia en el modelo directo). Pues bien, corriendo el modelo con un intervalo temporal de Δt=0,1 día, y llevando el número de iteraciones antes de la interrupción por falta de convergencia a 100000, *no se pudo llegar a la convergencia:* al llegar a la iteración 100000, la diferencia entre valores sucesivos $x^{100000-N}$ y x^{100000}, donde N es el tamaño de la matriz,es de 0,01572. El programa emplea 11 minutos en llevar a cabo las 100000 iteraciones. Usando ahora 200000 iteraciones (para las que el

programa emplea 22 minutos), la diferencia entre valores aproximados es aún de 0,01113, lo cual indica la extrema lentitud del método y obliga a descartarlo totalmente.

Cabe acotar que con las mismas condiciones de contorno, el modelo, tomando las condiciones iniciales del cuadro 27, es decir, *ya estabilizado*, es prácticamente instantáneo, es decir, la convergencia es inmediata: la aproximación inicial coincide con la solución.

Por otra parte, tomando como condiciones iniciales las de dicho cuadro 27, se efectuó la corrida de mareas cuyas condiciones iniciales se indican en el cuadro 28. Primeramente se intentó usar una tolerancia ε=0,01, con Δt=0,25 de día, y el procedimiento no convergió (con 32000 iteraciones). Con ε=0,01 y Δt=0,125 de día tampoco convergió; sólo se pudo efectuar la corrida con ε=0,02 y Δt=0,125. Se notan errores de precisión importantes, debido a la tolerancia tan permisiva, como se indica en el cuadro 42, correspondiente al estado del sistema el día 4 (comparar con el cuadro 30, en que se usó el método directo). El tiempo promedio necesario fue de 1 minuto por intervalo temporal. Cabe indicar que, con este modelo, para realizar las 32000 iteraciones admitidas antes de discontinuar se requieren 3,5 minutos aproximadamente, por lo cual no se consideró razonable aumentar el número de iteraciones: si para un modelo con 48 puntos de discretización fueran necesarios más de 3 minutos por intervalo de cálculo (sin por eso garantizarse una precisión adecuada) el modelo es claramente inservible para uso industrial.

Cuadro 42

Resultados de la corrida de marea para el día 4

Punto	1	2	3	4	5	6	7	8
Z (m)	121,49	121,28	121,07	119,80	119,68	119,57	121,48	121,28
Q (m^3/s)	1000	1000	1000	1000	1000	1000	1000	1000
Punto	9	10	11	12	13	14	15	16
Z (m)	121,07	121,07	120,34	119,57	119,57	117,60	114,96	114,96
Q (m^3/s)	1000	2000	2000	2000	3000	3000	3000	1320
Punto	17	18	19	20	21	22	23	24
Z (m)	114,45	113,71	112,83	112,83	112,05	111,42	111,42	111,25
Q (m^3/s)	1320	1320	1320	1267	1269	1269	652	650
Punto	25	26	27	28	29	30	31	32
Z (m)	111,07	111,07	109,72	108,12	108,12	108,10	108,30	114,96
Q (m^3/s)	649	1264	1265	1272	591	594	582	1680
Punto	33	34	35	36	37	38	39	40
Z (m)	114,23	113,58	112,91	112,91	112,86	112,83	111,42	111,31
Q (m^3/s)	1680	1681	1680	-52	-52	-53	617	616
Punto	41	42	43	44	45	46	47	48
Z (m)	111,07	112,91	111,16	109,06	108,12	108,06	108,16	108,05
Q (m^3/s)	615	1732	1732	1731	681	681	680	677

A1.3. VARIANTE DEL METODO DE LAS PROYECCIONES

Tomemos el par de ecuaciones lineales correspondientes al intervalo entre puntos de discretización (i,i+1).

En este caso $x^{(k)}$ deberá ser la proyección de $x^{(k-1)}$ sobre la variedad lineal $\mathbb{L}$ de dimensión n-2 formado por la intersección de los hiperplanos $\langle a_{2i-1}.y\rangle = b_{2i-1}$ y $\langle a_{2i}.y\rangle = b_{2i}$, es decir $x^{(k)}-x^{(k-1)}$ debe ser perpendicular a los respectivos hiperplanos homogéneos,

$$x^{(k)} - x^{(k-1)} \perp \langle a_{2i-1}.y\rangle = 0 \qquad x^{(k)} - x^{(k-1)} \perp \langle a_{2i}.y\rangle = 0$$

es decir,

$x^{(k)}-x^{(k-1)}=\lambda_{1k}a_{2i-1}+\lambda_{2k}a_{2i}y$ $x^{(k)} \in \mathbb{L}$. Por consiguiente

$$\langle a_{2i-1}.x^{(k-1)}\rangle + \langle a_{2i-1}.(\lambda_{1k}a_{2i-1} + \lambda_{2k}a_{2i})\rangle = b_{2i-1}$$

$$\langle a_{2i}.x^{(k-1)}\rangle + \langle a_{2i}.(\lambda_{1k}a_{2i-1} + \lambda_{2k}a_{2i})\rangle = b_{2i}$$

donde n es par y $k \equiv i \bmod n/2$. O sea

$$\lambda_{1k}\|a_{2i-1}\|^2 + \lambda_{2k}\langle a_{2i-1}, a_{2i}\rangle = b_{2i-1} - \langle a_{2i-1}.x^{(k-1)}\rangle$$

$$\lambda_{1k}\langle a_{2i-1}, a_{2i}\rangle + \lambda_{2k}\|a_{2i}\|^2 = b_{2i} - \langle a_{2i}.x^{(k-1)}\rangle$$

y λ_{1k}, λ_{2k} se obtienen resolviendo el sistema lineal $B_i\lambda_k = d_k$, siendo

$$B_i = \begin{bmatrix} \|a_{2i-1}\|^2 & \langle a_{2i-1}, a_{2i}\rangle \\ \langle a_{2i-1}, a_{2i}\rangle & \|a_{2i}\| \end{bmatrix}$$

$$d_k = (b_{2i-1} - < a_{2i-1} . x^{(k-1)} > \quad , \quad b_{2i} - < a_{2i} . x^{(k-1)} >)^t$$

En cada iteración se obtiene entonces $x^{(k)}$ como

$$x^{(k)} = x^{(k-1)} + \lambda_{1k} a_{2i-1} + \lambda_{2k} a_{2i}$$

Las n/2 matrices B_i se calculan una vez sola al principio del algoritmo, y también se triangulan una sola vez.

Es necesario asegurarse de que el método converge siempre. Para ello generalizaremos la demostración, que puede verse en Gastinel [1966], de convergencia (de orden lineal) del método de las proyecciones de Kaczmarz. La línea de la demostración es similar, y no pretendemos originalidad pues es muy simple.

TEOREMA: El método de Kaczmarz modificado con proyecciones sobre variedades lineales de dimensión n-2 (siendo n el orden de la matriz A) generados por pares caonsecutivos de ecuaciones converge siempre, y el orden de convergencia es lineal.

DEMOSTRACION: Sea $x^{(k)}$ la k-sima iteración del método.
$x^{(k)}=x^{(k-1)} + \lambda_{1k} a_{2i-1} + \lambda_{2k} a_{2i}$

con $x^{(k)}$ proyección de $x^{(k-1)}$ sobre la variedad lineal $\mathbb{L}$ de dimensión n-2 formada por la intersección de los hiperplanos $a_{2i-1}x=b_{2i-1}$ y $a_{2i}x=b_{2i}$, con $k \equiv i \bmod n/2$. Sea z la solución del sistema Ax=b, o sea, en particular, $z \in \mathbb{L}$. Por consiguiente, $x^{(k)}-z$ es perpendicular a $\lambda_{1k} a_{2i-1} + \lambda_{2k} a_{2i}$, y como

$$x^{(k-1)}-z = x^{(k)}-z- (\lambda_{1k} a_{2i-1}+\lambda_{2k} a_{2i})$$

por el teorema de Pitágoras será

$$\| x^{(k-1)}-z \|^2 = \| x^{(k)}-z \|^2 + \| \lambda_{1k} a_{2i-1} + \lambda_{2k} a_{2i} \|^2$$

o sea $$\| x^{(k)} - z \| \leq \| x^{(k-1)} - z \|$$

Es decir, la sucesión $\| x^{(k)} - z \|$ es una sucesión monótona no creciente, y en particular los $x^{(k)} - z$ están todos contenidos en un compacto.

Ahora bien, sea $r^{(k)} = Ax^{(k)} - b$. Entonces será

$\lambda_k^t = B_i^{-1} d_k$, donde $\lambda_k^t = (\lambda_{1k}, \lambda_{2k})^t$

o sea $\lambda_k^t = B_i^{-1} d_k = -B_i^{-1}(b_{2i-1} - \langle a_{2i-1}, x^{(k-1)} \rangle, b_{2i} - \langle a_{2i}, x^{(k-1)} \rangle)^t$

$$= -B_i^{-1} W_i r^{(k)}$$

siendo W_i la matriz de 2 x n

$$W_i = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

con todos sus elementos nulos salvo el i-1-simo sobre la fila 1 y el i-simo sobre la fila 2.

Por consiguiente podemos escribir

$$r^{(1)} = Ax^{(1)} - b = A(x^{(0)} - (a_1, a_2) B_1^{-1} W_1 r^{(0)}) - b =$$

$$r^{(0)} - A(a_1, a_2) B_1^{-1} W_1 r^{(0)} = (I - M_1) r^{(0)}$$

con $M_1 = A(a_1, a_2) B_1^{-1} W_1 r^{(0)}$

y análogamente

$$r^{(2)} = (I - M_2)\, r^{(1)} = (I - M_2)\,(I - M_1) r^{(0)}$$

. . . . . .

$$r^{(n/2)} = (I - M_{n/2})\, r^{(n/2-1)}$$

$$= (I - M_{n/2})\,(I - M_{n/2-1}) \ldots (I - M_1)\, r^{(0)}$$

$$= N\, r^{(0)}$$

o sea $$x^{(n/2)} - z = A^{-1} N A\, (x^{(0)} - z)$$

Se tiene entonces que, si hay convergencia, ésta es lineal. Veamos que hay efectivamente convergencia. Por empezar, $r^{(np/2)} = N^p r^{(0)}$. Por otra parte, como dado q existe p tal que

$$np/2 \leq q \leq n(p+1)/2$$

vale $\| x^{(n(p+1)/2)} - z \| \leq \| x^{(q)} - z \| \leq \| x^{(np/2)} - z \|$

Consideremos el conjunto de proyecciones $x^{(n/2)}$, $x^{(n)}$, $x^{(3n/2)}$,...,sobre la variedad lineal $\mathbb{L}_{n/2}$ generada por las ecuaciones $a_{n-1}x=b_{n-1}$, $a_n x=b_n$. Como los números $\| x^{(n/2)} - b \|$ $\| x^{(n)} - z \|$, $\| x^{(3n/2)} - z \|$,... forman una sucesión monótona no creciente, tienen un límite $s \geq 0$ (que es a su vez límite de los $\| x^{(q)} - z \|$), y existe una subsucesión $x^{(np'/2)}$ que converge a un punto $y^{(n/2)}$ tal que $\| y^{(n/2)} - z \| = s$. Pero entre los puntos $x^{(np'/2+1)}$ (que pertenecen a la variedad lineal $\mathbb{L}_1$ generada por las ecuaciones $a_1x=b_1$, $a_2x=b_2$) existe una subsucesión convergente a un punto $y^{(1)}$ tal que $\| y^{(1)} - z \| = s$. Como por continuidad del operador lineal de proyección sobre una variedad lineal vale que $y^{(1)}$ es la proyección de $y^{(n/2)}$ sobre $\mathbb{L}_1$, la única posibilidad de que $\| y^{(n/2)} - z \| = \| y^{(1)} - z \| = s$ es que $y^{(n/2)} = y^{(1)}$. Se toma ahora una subsucesión de la subsucesión usada para encontrar $y^{(1)}$, y repitiendo el procedimiento se llega a un límite de esta subsucesión $y^{(2)}=y^{(1)}=y^{(n/2)}$. Continuando así se obtiene que existe un punto y que pertenece a todas las variedades lineales $\mathbb{L}_1$, $\mathbb{L}_2$,...,$\mathbb{L}_{n/2}$. Pero eso es equivalente a decir que $Ay=b$. O sea $s=0$, y el procedimiento iterativo converge. ■

A1.4. EXPERIMENTOS NUMERICOS CON EL METODO DE LAS PROYECCIONES MODIFICADO

Se realizó un ensayo con la red fluvial indicada en la figura 22 y definida en el cuadro 24, con las condiciones iniciales del cuadro 25 y las condiciones de contorno del cuadro 26, con $\Delta t=0,01$, hasta tratar de alcanzar una tolerancia de $\varepsilon=0,001$, como en el caso anterior. El programa no convergió después de 100000 iteraciones, para las que empleó 30 minutos (casi tres veces más que con el método sin modificar) observándose además una diferencia entre iteraciones sucesivas de 0,02403, mucho mayor, por otra parte, que la observada en el método sin modificar.

Por tal motivo, se conviene que, para el modelo hidrodinámico de red fluvial deltaica, el método de Kaczmarz modificado es aún más inaceptable que el método de Kaczmarz original.

APENDICE 2.

ALGUNAS CARACTERISTICAS DEL METODO DE PREISSMANN

En este apéndice se indican los resultados de algunos experimentos numéricos efectuados con distintos valores de θ para comprobar las propiedades conservativas del método de Preissmann. es decir, para verificar que el flujo de agua que entra aguas arriba del modelo y no sale aguas abajo se acumula. No suelen publicarse estos experimentos, por lo cual consideramos útil su inclusión en este trabajo. Tomaremos como modelo de prueba un tramo simple, pero el resultado vale para redes arborescentes y deltaicas debido a que la conservación de la masa está expresamente incluida en las ecuaciones de compatibilidad de Stoker.

El tramo simple consta de 10 secciones transversales de discretización, separadas por un intervalo espacial Δx de 10 km cada una de la siguiente; las secciones transversales son rectangulares e iguales, de ancho 100 m constante. El coeficiente de conducción varía linealmente entre 0 y 80000 m^3/s para alturas desde el fondo de 0 a 10 m, respectivamente. La pendiente de fondo es de 1 en 10000, y el nivel del fondo del punto extremo aguas arriba es 100 m. Tomando condiciones iniciales estacionarias (ya calculadas antes) dadas por el cuadro 43, las condiciones de contorno consisten en mantener fijo el nivel en la sección aguas abajo, y pasar linealmente, en 20 días, el caudal aguas arriba de 500 m^3/s a 750 m^3/s (situación similar a la que puede presentarse al modelizar un tramo de río que termina en un embalse mantenido a nivel constante)[5].

Cuadro 43

Condiciones iniciales estacionarias para tramo simple

Punto	Nivel (m)	Caudal (m^3/s)	Punto	Nivel (m)	Caudal (m^3/s)
1	107,61	500	6	102,18	500
2	106,57	500	7	100,99	500
3	105,51	500	8	99,66	500
4	104,43	500	9	98.15	500
5	103,33	500	10	96.00	500

Deberá cumplirse, teóricamente, que

$$\int_0^T (Q(x_1,t)-Q(x_L,t))dt = \int_{x_1}^{x_L} (B(Z(x,T))Z(x,T))-B(Z(x,0))Z(x,0))dx$$

siendo 0 y T los instantes inicial y final de la simulación, y x_1 y x_L los extremos aguas arriba y aguas abajo; es decir, la conservación en su expresión más simple: lo que entra, si no sale, se acumula. Naturalmente, será necesario integrar numéricamente ambas expresiones, tomando como puntos para las cuadraturas respectivas los intervalos espaciales y temporales donde se conocen las variables; se usará directamente la regla de integración del trapecio. El error (o sea el agua que "aparece" o "desaparece" por razones numéricas) se calculará como porcentaje respecto de la variación del volumen debida a la diferencia entre el caudal entrante y el saliente, es decir,

$$\varepsilon = 100[\Delta V - (Q_E-Q_S)\Delta t]/[(Q_E-Q_S)\Delta t]$$

donde ΔV es el valor (calculado por integración numérica) del segundo miembro de la ecuación de conservación, y $(Q_E-Q_S)\Delta t$ es el valor (también calculado numéricamente) del primer miembro de la ecuación de conservación. Los resultados obtenidos (con el modelo de red fluvial arborescente) son los que se muestran en el cuadro 44 siguiente:

Cuadro 44

Conservación numérica de la masa en un tramo unidimensional

θ	ΔV (Hm^3)	$(Q_E-Q_S)\Delta t$ (Hm^3)	ε (%)
0,75	23,83	23,80	0,126
0,85	23,79	23,20	2,543
0,90	23,77	23,11	2,856
0,85*	49,65	49,08	1,161

*: Corrida manteniendo la derivada de Q aguas arriba hasta el día 40 (o sea hasta Q=1000), para observar si la generación artificial de masa líquida en el río va aumentando o no (no va aumentando).

Si bien la cantidad de experimentos realizados es escasa, podemos observar que la generación artificial de masa líquida se mantiene dentro de una proporción bastante razonable.

Cabe indicar que para θ=0,66 la simulación se hace inestable, tal como se anticipó en la seccion 10. Esto puede verse en el cuadro 45, en que se notan las oscilaciones del caudal aguas abajo. Para observar mejor la inestabilidad se corrió 30 días en vez de 20, llevando entonces el caudal aguas arriba, manteniendo la derivada del crecimiento, hasta 875 m^3/s.

Cuadro 45

Desestabilización del caudal aguas abajo con θ=0,66

Día	1	2	3	4	5	6	7	8	9	10
Q_{10} (m^3/s)	504	512	524	535	549	560	574	584	600	609
Día	11	12	13	14	15	16	17	18	19	20
Q_{10} (m^3/s)	626	632	652	655	681	675	712	692	749	701
Día	21	22	23	24	25	26	27	28	29	30
Q_{10} (m^3/s)	795	698	861	670	964	600	1139	462	1435	259

Si bien ya habíamos observado problemas de inestabilidad al intentar obtener un estado estacionario de la red fluvial arborescente (ver Cuadro 10), en ese caso partíamos de condiciones iniciales físicamente dudosas. Se exhibe aquí este ejemplo pues tanto las condiciones iniciales como las de contorno son muy razonables, y el tramo modelizado es además muy sencillo.

Con el objeto de comparar los resultados del modelo de red fluvial arborescente y los obtenidos con el método directo para redes deltaicas, se corrieron ambos modelos con las condiciones de contorno indicadas. Los resultados fueron exactamente iguales, y se indican en el cuadro 46 para el tiempo = 10 días (θ=0,85).

Cuadro 46

Estado del tramo simple con datos antes indicados para t=10 días

Punto	1	2	3	4	5	6	7	8
Z (m)	109,27	108,18	107,06	105,91	104,73	103,50	102,18	100,72
Q (m^3/s)	625	623	621	619	617	616	614	612

Punto	9	10
Z (m)	99,00	96,00
Q (m^3/s)	611	611

Posteriormente, se utilizó este tramo sencillo para realizar las pruebas mencionadas en la sección 17 de este trabajo: con el fin de verificar la consistencia de los modelos para cambios en las condiciones de contorno, se corrieron ambos reemplazando como condiciones de contorno las anteriores (caudal aguas arriba y nivel aguas abajo) por las complementarias (nivel aguas arriba y caudal aguas abajo), asignando a éstas los valores calculados en la corrida anterior. La buena coincidencia de los resultados con los obtenidos antes (y en particular la coincidencia entre los resultados obtenidos en los caudales aguas arriba y los niveles aguas abajo con los datos de la primera corrida) es condición necesaria para la consistencia. Dicha coincidencia es excelente, como puede observarse comparando el cuadro anterior con el cuadro 47 siguiente, en donde se exponen los resultados, también para el tiempo = 10 días, de la corrida efectuada con las condiciones de contorno cambiadas:

Cuadro 47

Estado del tramo simple el día 10 para condiciones de contorno cambiadas

Punto	1	2	3	4	5	6	7	8
Z (m)	109,27	108,17	107,05	105,91	104,73	103,49	102,17	100,71
Q (m^3/s)	625	623	621	619	617	616	614	613

Punto	9	10
Z (m)	98,98	95,97
Q (m^3/s)	611	611

Las condiciones de contorno usadas, tomadas de la corrida indicada en el cálculo de la conservación numérica de la masa, se dan en el cuadro 48:

Cuadro 48

Condiciones de contorno intercambiadas

Tiempo	Z_1 (m)	Q_{10} (m^3/s)	Tiempo	Z_1 (m)	Q_{10} (m^3/s)
0	107,61	500	6	108,58	561
1	107,72	505	7	108,75	573
2	107,89	513	8	108,93	586
3	108,06	524	9	109,10	598
4	108,23	536	10	109,27	611
5	108,41	548			

APENDICE 3.

COMENTARIOS SOBRE PROGRAMAS COMPUTACIONALES Y MODELOS

En este apéndice se describirán someramente los programas computacionales utilizados en este trabajo, que fueron todos desarrollados por el autor. La computadora usada fue una Toshiba 3100, con 640 Kb de memoria, disco rígido de 10 Mb, unidad de disco flexible de 3,5 pulgadas y como unidades periféricas (no pudiéndose usar ambas simultáneamente) disquetera de discos flexibles de 5 1/4 pulgadas e impresora Epson 86b. Los programas fueron escritos en TURBOPASCAL, con la intención de que fueran lo más modulares y estructurados posibles, y para aprovechar la extraordinaria eficiencia en la compilación en computadora personal de dicho lenguaje en su compilador Turbo. Sin embargo, no se utilizó todo el potencial de PASCAL: en particular, pese a que la descripción de matrices ralas mediante punteros se presta naturalmente para el uso del tipo de dato pointer de PASCAL, se prefirió usar vectores punteros en forma clásica, pensando en facilitar así una posible traducción a FORTRAN. La estructura de los programas es la misma para todos: en realidad, es la estructura natural para modelos de simulación en "tiempo continuo discretizado", que se utiliza en general con mayor o menor precisión en la mayoría de los grandes modelos matemáticos computacionales dinámicos en existencia (siempre y cuando los modelistas actualicen sus versiones manteniendo la filosofía original). Curiosamente, no es habitual mencionar la estructura computacional *general* en los libros y artículos sobre modelos matemáticos (aunque sí, por supuesto, adjuntar los listados). En ese sentido, aunque no se refiere a modelos de resolución numérica de ecuaciones en derivadas parciales con valores iniciales sino a modelos de optimización en planificación y control, es interesante e ilustrativo el trabajo de Matrosov *et al.* [1979], donde utiliza el término (creo que muy adecuado) "tecnología matemática" para referirse a técnicas de modelización.

La estructura general de los programas es la siguiente: Existen tres clases de datos:

a) Datos de constitución del modelo matemático. Estos son los datos que definen las características físicas, topológicas, geométricas, etc., del modelo considerado (los ríos que lo forman, los puntos de discretización, la distancia entre ellos, los anchos o áreas mojadas para distintos niveles, los coeficientes de conducción para distintos niveles, etc.).

b) Las condiciones iniciales del sistema. Son simplemente los valores de Z y Q en el instante inicial en los puntos de discretización (dado que el método numérico usado usa valores de un paso de tiempo solamente para calcular el siguiente). Para métodos numéricos de paso múltiple es necesario dar n-uplas de condiciones iniciales (siendo n el conjunto de pasos en los tiempos $t, t-\Delta t_1, \ldots, t-\Delta t_1-\ldots-\Delta t_{n-1}$ usados para calcular la solución en el instante $t+\Delta t$, o calcularlos mediante procedimientos auxiliares.

c) Las condiciones de contorno. En regímenes subcríticos, que son los que estamos tratando, deberá darse una condición de contorno en cada extremo abierto (aguas arriba o aguas abajo) de la red fluvial. En esta tesis, por simplicidad, se trabajó tomando como posibles condiciones de contorno en un extremo abierto exclusivamente el nivel Z o el caudal Q, pero las condiciones de contorno pueden ser de la forma

$$Z = F(Q)$$

o sea, en terminología de hidráulica fluvial, leyes altura/caudal; también se pueden dar en forma implícita:

$$F(Z,Q) = 0$$

Obsérvese que esta clasificación es muy general, y puede ser adaptada sin dificultad a modelos matemáticos de otro tipo, por ejemplo económicos (aunque estos suelen usar otra terminología, con "variables de estado", "variables endógenas", "variables exógenas", etc., ver por ejemplo Naylor [1971]).

Entre los datos de constitución del modelo están aquéllos que pueden medirse o calcularse sin dificultad, como áreas, anchos, etc., y aquéllos que deben ajustarse empíricamente, pues no se

sabe con precisión su valor (para nuestro caso de modelos hidrodinámicos de ríos reales estos datos son los coeficientes de conducción). El ajuste y validación (esto es, confirmación de la bondad del modelo en casos reales en los que hay que estimar parámetros desconocidos), son temas de por sí complicados, que se alejan del tema de este trabajo propiamente dicho. Y en particular los métodos de ajuste automático, sobre los cuales puede verse el trabajo de Becker y Yeh [1972]. Otro problema usual de modelización es el de obtención de las condiciones iniciales. Este problema no es tradicionalmente mencionado en la bibliografía matemática puesto que es muy común que el problema de ecuaciones diferenciales en derivadas parciales a resolver (teórica y/o numéricamente) sea un problema de Cauchy o un problema mixto, para el cual *ya se conocen* las condiciones iniciales y, eventualmente, las condiciones de contorno. En modelización fluvial esto no es así: cuando se quiere modelizar un río, por ejemplo, pueden desconocerse los estados estacionarios del río, o estados físicamente factibles a partir de los cuales se puedan producir los fenómenos que se quieren simular. En particular, en los procesos de ajuste se quieren reproducir episodios históricos de flujo de agua, para lo cual hay que partir de un estado inicial, presuntamente también histórico, pero que no se conoce sino parcialmente (no es común disponer de registros históricos de niveles en todos los puntos de discretización del río o sistema fluvial considerado, pues las escalas hidrométricas no se instalan con tanta densidad, y mucho menos de registros de caudales o velocidades, que solo se miden esporádicamente y en general en campañas especialmente organizadas). Por consiguiente, lo normal es emplear el procedimiento utilizado en este trabajo: partir de condiciones iniciales supuestamente factibles desde el punto de vista físico, y tratar de llegar a estados estacionarios, a partir de los cuales se comienza la experimentación numérica propiamente dicha. Y esas condiciones iniciales tentativas, en los casos de redes arborescentes o deltaicas, deben cumplir las condiciones de compatibilidad de Stoker.

Por otro lado, también debe tenerse cuidado, cuando se desean llevar a cabo simulaciones, en considerar solamente condiciones de contorno factibles. Esto se traduce principalmente en tener que prestar mucha atención cuando en todos los extremos abiertos las condiciones conocidas son caudales: en ese caso los valores dados deberán cumplir la ley de conservación de la masa líquida. El problema es menos grave - o el modelo menos sensible al problema - cuando se combinan entre las condiciones de contorno niveles y caudales; en este caso el modelo puede forzar razonablemente el cumplimiento de las condiciones de contorno, como se vio en este trabajo.

Por consiguiente, la estructura general de los programas es:

a) Armado del modelo físico.

b) Lectura de condiciones iniciales, tiempo inicial de cálculo, intervalo(s) de discretización temporal usado(s), tiempo(s) de cambio de valor de intervalo de discretización temporal, (si los hay), tiempo final de la simulación. Armado de la estructura general (invariante en el tiempo) de la matriz A.

c) Tiempo de cálculo: tiempo inicial + intervalo de discretización temporal.

d) **Mientras** el tiempo de cálculo no es mayor que el tiempo final, leer condiciones de contorno para ese tiempo, armar el sistema Ax=b correspondiente a ese tiempo, resolverlo, asignar los valores de las variables Q y Z correspondientes a ese tiempo, avanzar el tiempo de cálculo.

La diferencia entre los programas se da exclusivamente en el armado de la estructura general de la matriz A y el armado y resolución del sistema Ax=b en cada paso de tiempo. Por otra parte, estos son los puntos fundamentales de los programas, en los cuales se ha usado la teoría desarrollada en este trabajo, dado que el resto de los problemas a encarar en la programación está bastante normalizado.

Los programas usados son:

CUENSIMP: resuelve el modelo de red fluvial arborescente para el sistema hiperbólico simplificado lineal con coeficientes constantes según la teoría de la segunda parte de este trabajo.

CUENCAS: resuelve el modelo de red fluvial arborescente general según la teoría de la segunda parte de este trabajo.

DELTSIMP: resuelve el modelo de red fluvial deltaica para el sistema hiperbólico simplificado lineal con coeficientes constantes según la teoría de la tercer parte de este trabajo.

DELTA: resuelve el modelo de red fluvial deltaica por un método directo según la teoría de la tercer parte de este trabajo.

DELTA1: resuelve el modelo de red fluvial deltaica por el método de las proyecciones de Kaczmarz,según se ve en el Apéndice 1.

DELTA2: resuelve el modelo de red fluvial deltaica por el método de las proyecciones de Kaczmarz modificado, según se ve en el Apéndice 1.

CUENSIMP y DELTSIMP son sencillamente versiones de CUENCAS Y DELTA, respectivamente, con cálculo de coeficientes de las discretizaciones (18) y (20) simplificados y con eliminación de las variables en este caso superfluas (tablas de niveles, de anchos, de áreas, de coeficientes de conducción).

Se acompañan al final de este trabajo los listados de todos los mencionados programas computacionales.

NOTAS

1. Con respecto a régimen subcrítico y régimen supercrítico, es conocida (ver Stoker [1957], capítulo 2, sección 3) la analogía entre las ecuaciones de aguas poco profundas y las ecuaciones isentrópicas de dinámica de gases politrópicos. En efecto, si tomamos en las ecuaciones (3) como variables a la velocidad v=Q/h y a h, y se desprecian los términos a la derecha de la igualdad, debidos a la fricción y a la gravedad, las ecuaciones serán

$$\partial h/\partial t + h\partial v/\partial x + v\partial h/\partial x = 0 \qquad (3.3)$$

$$\partial v/\partial t + v\partial v/\partial x + g\partial h/\partial x = 0 \qquad (3.4)$$

y las ecuaciones de dinámica de gases (es decir, las ecuaciones de fluidos compresibles) son

$$\partial \rho/\partial t + \rho\partial v/\partial x + v\partial \rho/\partial x = 0 \qquad (3.5)$$

$$\partial v/\partial t + v\partial v/\partial x + \rho^{-1}\partial p/\partial x = 0 \qquad (3.6)$$

donde v es la velocidad, como antes, ρ es la densidad del fluido y p la presión. Suponiendo una ecuación de estado del gas de la forma

$$p = K\rho^{\gamma}$$

donde K y γ son constantes ($\gamma = c_p/c_v$, siendo c_p y c_v el calor específico a presión constante y a temperatura constante, respectivamente), y suponiendo $\gamma = 2$ (esto vale sólo para un gas teórico), será

$$\partial p/\partial x = K\gamma\rho^{\gamma-1}\partial \rho/\partial x = 2K\rho\partial \rho/\partial x$$

y la ecuación (3.6) resulta

$$\partial v/\partial t + v\partial v/\partial x + 2K\partial \rho/\partial x$$

y (3.5) y (3.6) equivalen a (3.3) y (3.4) reemplazando g por 2K y h por ρ. Los regímenes subcrítico y supercrítico de la teoría de aguas poco profundas se corresponden entonces con los regímenes subsónico y supersónico de dinámica de gases. Esto suele ser útil desde el punto de vista teórico, pues se tiene un conocimiento mayor sobre fluidos compresibles que sobre fluidos incompresibles.

2. Sin embargo, la tesis de Cunge es útil sobre todo por sus experimentos numéricos, en los cuales ofrece ejemplos de que cuando $r=\Delta t=\Delta x=1$ y $\Theta=1/2$ no se produce atenuación de ondas (para el sistema (49) lineal); de que cuando $\Theta=1/2$ y $r\neq 1$ se observan oscilaciones parásitas que se van eliminando a medida que $\Theta\to 1$. No es tan útil en cuanto a sus demostraciones teóricas: en su demostración de estabilidad del esquema de Preissmann para $\Theta \geq 1/2$ usa el método de Godunov y Riabenkii (ver Godunov y Riabenkii [1964], capítulo 5, sección 2, y capítulo 6), de teoría espectral de operadores en diferencias que permite estudiar problemas (lineales, de todos modos) mixtos, esto es, con condiciones de contorno, pero deduce la acotación uniforme de la familia de operadores de Godunov y Riabenkii usando esencialmente el método de von Neumann de los coeficientes de Fourier y, por otra parte, tiene un error: Godunov y Riabenkii descomponen la matriz A originada por la discretización en la forma $w^{n+1}=R_{(\Delta t(\Delta x),\Delta x)}w^n + \Delta t\rho^n$, donde ρ^n depende exclusivamente de la discretización del término de la derecha de la ecuación diferencial y de las condiciones de contorno, y Cunge lo hace depender también de w^n.

3. La interpolación lineal de valores de anchos y coeficientes de conducción para distintos valores de niveles en cada sección transversal representada por un punto de discretización se utiliza por ser la más sencilla de programar. Es interesante, en modelos con secciones transversales más irregulares, reemplazar la interpolación lineal por otro tipo de representación de la correspondiente curva (por ejemplo, por funciones spline, o polinomios de algún orden), y evaluar la eficiencia relativa de linealizar a intervalos menores o usar un ajuste más elaborado, comparando resultados. Esto es muy claro cuando se trata de las curvas de anchos, pero no tanto cuando se trata de curvas de

coeficientes de conducción, que suelen tener que ser ajustados: cuanto menos sean los puntos de ajuste, mejor (en el sentido de que el ajuste implica menos trabajo).

4. Curiosamente, es muy inusual que los libros de análisis numérico, e incluso de análisis numérico lineal, mencionen el método de las proyecciones de Kaczmarz. En la bibliografía citada en la muy completa reseña de Censor [1981], solamente figura el ya indicado libro de Gastinel [1966] y el (muy denso) de Householder [1964]. Es probable que esto se deba a que en definitiva los métodos directos en álgebra lineal numérica (y otros métodos iterativos) parecen ser siempre más eficientes que el de las proyecciones, como este apéndice no hace más que reafirmar.

5. Se puede observar en este caso particular (y en la estabilización anterior) la conveniencia de un esquema implícito: como se usó Δt = 1 día y Δx = 10 km., se tiene $r = \Delta t/\Delta x$ = 86400/10000=8,64, mientras que la condición de Courant impone una cota superior para un esquema explícito de $r \leq 1/|\lambda|$ para cada autovalor de la matriz A, o sea aproximadamente

$$r \leq 1/(0,5+\sqrt{gS/B}\ ) \approx 1/(0,5+\sqrt{10\times 5} \approx 0,13$$

Es decir, se necesitarían alrededor de 66 veces más intervalos de cálculo para simular el mismo tiempo.

Ahora bien, este cálculo debe ser ponderado teniendo en cuenta el número total de operaciones necesarias para pasar de un instante t a un instante t+Δt. Para simplificar el análisis que se describe a continuación, consideraremos multiplicaciones y divisiones con el mismo peso, y despreciaremos las sumas (en esencia, como hay un número bastante similar de sumas que de multiplicaciones, estamos igualando el "costo" de una división al de una suma más una multiplicación). Para el esquema de Preissmann se necesitan 18n operaciones, donde n es el tamaño de la matriz A, más el número de operaciones necesarias para calcular los coeficientes indicados en (18), (20). Para asumir una hipótesis pesimista, supongamos que son 75 (se pueden hacer menos cuentas que 75). En total, 93 operaciones. Supongamos un esquema explícito

ideal en el cual el número de operaciones será el necesario para obtener los valores del vector de la derecha b en la fórmula Ax=b (en realidad todo esquema explícito será más complicado), es decir, asignémosle la cantidad de operaciones necesaria para calcular A_{5i} y B_{5i}, más una multiplicación (por Δt). Se requeriran 14 operaciones. En el intervalo Δt implícito para el cual se requieren 93 operaciones del método de Preissmann, serán necesarias 14 x 66 operaciones con el método explícito "ideal", o sea 924 operaciones; es decir, unas 10 veces más.

BIBLIOGRAFIA

Amein, M. y C. S. Fang, Implicit flood routing in natural channels, *Journal of the Hydraulic Division*, ASCE, 96, 2481-2500, 1970.

Amein, M. y H. L. Chu, Implicit numerical modeling of unsteady flows, *Journal of the Hydraulic Division*, ASCE, 101, 717-731, 1975.

Becker, L. y W. W-G. Yeh, Identification of parameters in unsteady open channel flows, *Water Resources Research*, 8, 956-965, 1972.

Censor, Y., Row-action methods for huge and sparse systems and their applications, *SIAM Review*, 23, 444-466, 1981.

Chow, V.-T., *Open channel hydraulics*, Mc Graw-Hill, Nueva York, 1959.

Courant, R. y K. O. Friedrichs, *Supersonic flow and shock waves*, Interscience, Nueva York, 1948.

Courant, R. y D. Hilbert, *Methods of mathematical physics*, Vol II, Interscience, Nueva York, 1962.

Cunge, J. A, Un schéma de différences finies pour les équations hyperboliques (une et deux variables spaciales), Tesis presentada a la Facultad de Ciencias de la Universidad de Grenoble, 1966.

Cunge, J. A. y M. Wegner, Intégration numérique des équations d´écoulement de Barré de Saint-Venant par un schéma implicite de différences finies, *La Houille Blanche*, 1, 33-39, 1964.

Christofides, N., *Graph theory. An algorithmic approach*, Academic Press, Nueva York, 1975.

Dahlquist, G., y A. Björk, *Numerical methods*, Prentice-Hall, Englewood Cliffs, Nueva Jersey, 1974.

DiPerna, R. J., Global solutions to a class of nonlinear hyperbolic systems of equations, *Comm. Pure Applied Math.*, 26, 1-28, 1973.

EGASAT (Estudio Gradowczyk y Asociados S. A. T.), Modelo matemático hidrodinámico de fondo móvil de los canales de restitución de las centrales de Pichi Picún Leufú y Michihuao, Informe para Hidronor S. A., Buenos Aires, 1982.

Even, S., *Graph algorithms*, Computer Science Press, Rockville, Maryland, 1979.

Fread, D. L., Techniques for implicit dynamic routing with tributaries, *Water Resources Research*, 9, 918-926,1973.

Friedrichs, K. O., On the derivation of the shallow water theory, *Comm. Pure Applied Math.*, 81-87 (Apéndice a: Stoker, J. J., The formation of breakers and bores, *Comm. Pure Applied Math.*, 1-87), 1948.

Gastinel, N., *Analyse numérique linéaire*, Hermann, Paris, 1966.

Gelfand, I. M. y O. V. Lokutsievsky, The double sweep method for solution of difference equations, en: S. K. Godunov y V. S. Ryabenkii, *Theory of difference schemes*, North Holland, Amsterdam, Apéndice 2, 239-262, 1964.

George, A. y J. W. Liu, *Computer solution of large sparse positive definite systems*, Prentice-Hall, Englewood Cliffs, Nueva Jersey, 1981.

Glimm, J., Solutions in the large for nonlinear hyperbolic systems of equations, *Comm. Pure Appl. Math.*, **18**, 697-715, 1965.

Godunov, S. K., Métodos de cálculo numérico en diferencias para soluciones discontinuas de ecuaciones de la dinámica de fluidos, *Mat. Sb.*, **47**, 271-295, 1959 (en ruso).

Godunov, S. K., *Ecuaciones de la física matemática*, MIR, Moscú, 1978.

Godunov, S. K. y V. S. Riabenkii, *Theory of difference schemes*, North Holland, Amsterdam, 1964.

Golub, G. H. y C. F. van Loan, *Matrix computations*, Johns Hopkins University Press, Baltimore, Maryland, 1983.

Gradowczyk, M. H., Wave propagation and boundary instability in erodible-bed channels, *J. Fluid Mech.*, **33**, 93-112, 1968.

Gradowczyk, M. H. y P. M. Jacovkis, Un modelo matemático para regímenes impermanentes en redes fluviales complejas, 2° Jornadas Latinoamericanas de Computación, UTN, Buenos Aires, 1974.

Holt, M., *Numerical methods in fluid dynamics*, Springer-Verlag, Berlin, 1977.

Householder, A. S., *The theory of matrices in numerical analysis*, Dover, Nueva York, 1964.

Joliffe, I. B., Computation of dynamic waves in channel networks, *Journal of Hydraulic Engineering*, **110**, 1358-1370, 1984.

Knuth, D., *The art of computer programming*, Vol.1: *Fundamental algorithms*, Addison-Wesley, Reading, Massachusetts, 1975.

Lax, P. D., Hyperbolic systems of conservation laws II, *Comm. Pure Applied Math.*, **10**, 537-566, 1957.

Lax, P. D., Development of singularities of solutions of nonlinear hyperbolic partial differential equations, *J. Mathematical Phys.*, 5, 611-613, 1964.

Lax, P. D., The formation and decay of shock waves, *Amer. Math. Monthly*, 79, 227-241, 1972.

Lax, P. D., *Hyperbolic systems of conservation laws and the mathematical theory of shock waves*, SIAM, Philadelphia, 1973.

Lax, P. D. y R. D. Richtmyer, Survey of the stability of linear finite difference equations, *Comm. Pure Appl. Math.*, 9, 267-293, 1956.

Leendertse, J. J., Aspects of a computational model for long-period water-wave propagation, *RAND Corporation Memorandum RM-5294-PR*, Santa Monica, California, 1967.

Li, Z..C., L. J. Zhan y H. L. Wang, Difference methods of flow in branch channel, *Journal of Hydraulic Engineering*, 109, 424-446, 1983.

Liggett, J .A. y J. A. Cunge, Numerical methods of solution of the unsteady flow equations, en: K.Mahmood and V.Yevjevich (eds), *Unsteady flow in open channels*, Water Resources Publications, Fort Collins, Colorado, 89-182, 1975.

Liu, T.-P., Quasilinear hyperbolic systems, *Commun. Math. Phys.*, 68, 141-172, 1979.

Liu, T.-P., Quasilinear hyperbolic partial differential equations and the mathematical theory of shock waves, *Contemporary Mathematics*, 17, 9-20, 1983.

Liu, T.-P, y J. A. Smoller, The vacuum state in isentropic gas dynamics, *Adv. Appl. Math.*, 1, 345-359, 1980.

Mahmood, K., y V. Yevjevich (eds.), *Unsteady flow in open channels*, Volume III, Water Resources Publications, Fort Collins, Colorado, 1975.

Majda, A., *Compressible fluid flow and systems of conservation laws in several space variables*, Springer-Verlag, Nueva York, 1984.

Marshall, G., y R. Méndez, Computational aspects of the random choice method for shallow water equations, *J. Comput. Phys.*, 39, 1-21,1981.

Marshall, G. y A. N. Menéndez, Numerical treatment of nonconservation forms of the equations of shallow water theory, *J. Comput. Phys.*, 44, 167-188, 1981.

Matrosov, V. M., S. N. Vassilyev, O. G. Divakov y A. I. Tyatushkin, On technology of modelling and optimization of complex systems, en: G. I. Marchuk (ed.), *Modelling and optimization of complex systems*, (Proceedings of the IFIP-TC 7 Working Conference, Novosibirsk, 3-9 July 1978), Springer Verlag, Berlin, 1979.

Meis,T. y U. Markowitz, *Numerical solution of partial differential equations*, Springer-Verlag, Nueva York, 1981.

Naylor, T. H. (ed.), *Computer simulation experiments with models of economic systems*, Wiley, Nueva York, 1971.

Ockendon, H. y A. B. Tayler, *Inviscid fluid flows*, Springer-Verlag, Nueva York, 1983.

Oleinik, O. A., Discontinuous solutions of non-linear differential equations, *American Mathematical Society Translations*, Series 2, 26, 95-172, 1963.

Peyret, R. y T. D. Taylor, *Computational methods for fluid flow*, Springer-Verlag, Nueva York, 1983.

Pooch, U. W., y A. Nieder, A survey of indexing techniques for sparse matrices, *Computing Surveys*, 5, 109-133, 1973.

Preissmann, A., Propagation des intumescences dans les canaux et rivières, *Premier Congrès de l'Assoc. Française de Calcul*, Grenoble, 1961.

Quinn, F., y B. Wylie, Transient analysis of the Detroit River by the implicit method, *Water Resources Research*, 8, 1461-1469, 1972.

Richtmyer, R. D., *Difference methods for initial-value problems*, Interscience, Nueva York, 1957 (Primera edición).

Richtmyer, R. D. y K. W. Morton, *Difference methods for initial value problems*, Interscience, Nueva York, 1967 (Segunda edición).

Roache, P., *Computational fluid dynamics*, Hermosa Publishers, Albuquerque, Nuevo México, 1976.

Rozdestvenskiĭ, B. L. y N. N. Janenko, *Systems of quasilinear equations and their application to gas dynamics*, Translations of Mathematical Monographs, Vol. 55, American Mathematical Society, Providence, 1983.

Schmitz, G. y J. Edenhofer, Flood routing in the Danube River, by the new implicit method of characteristics (IMOC), *Third International Appl. Math. Modelling Conference*, Hamburg, 1-13, 1980.

Smoller, J., *Shock waves and reaction-diffusion equations*, Springer-Verlag, Berlin, 1983.

Stoker, J. J., The formation of breakers and bores. The theory of nonlinear wave propagation in shallow water and open channels, *Comm. Pure Applied Math.*, 1, 1-87, 1948.

Stoker, J. J., *Water waves*, Interscience, Nueva York, 1957.

Wood, E. F., B. M. Harley y F. G. Perkins, Transient flow routing in channel network, Research Report 75-1, International Institute for Applied Systems Analysis (IIASA), Laxenburg, 1975.

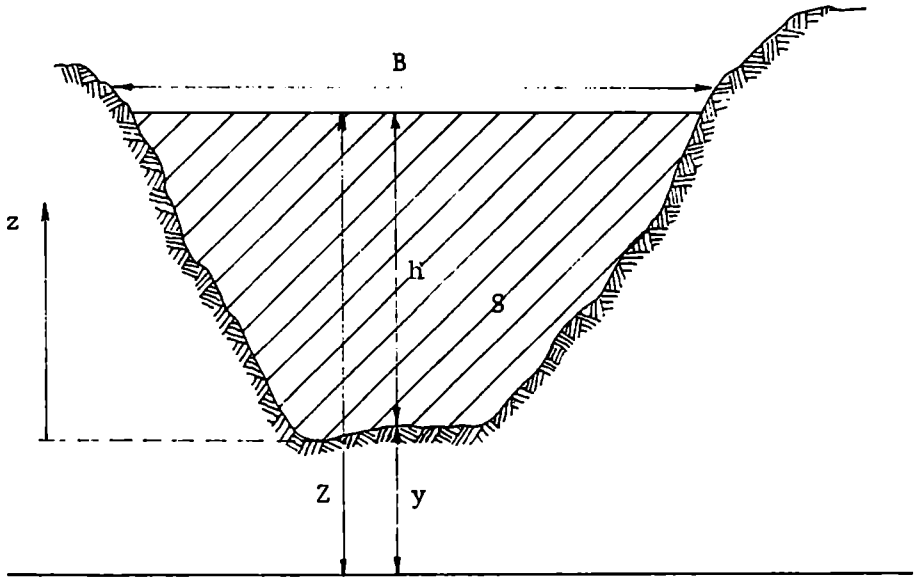


Figura 1: Sección transversal de un cauce fluvial

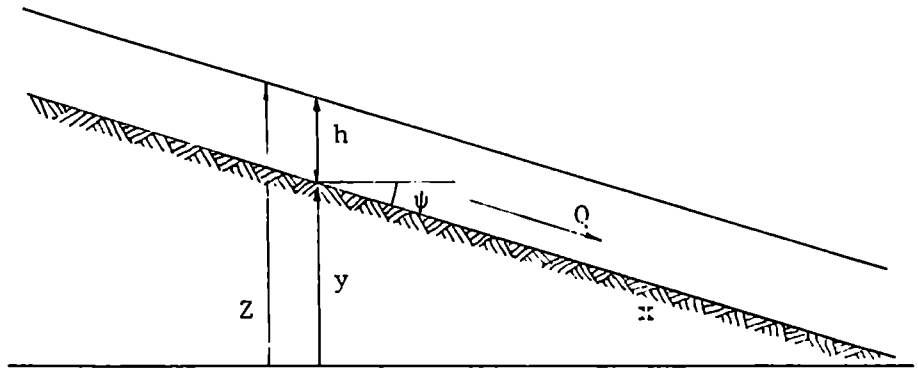


Figura 2: Corte longitudinal de un cauce fluvial

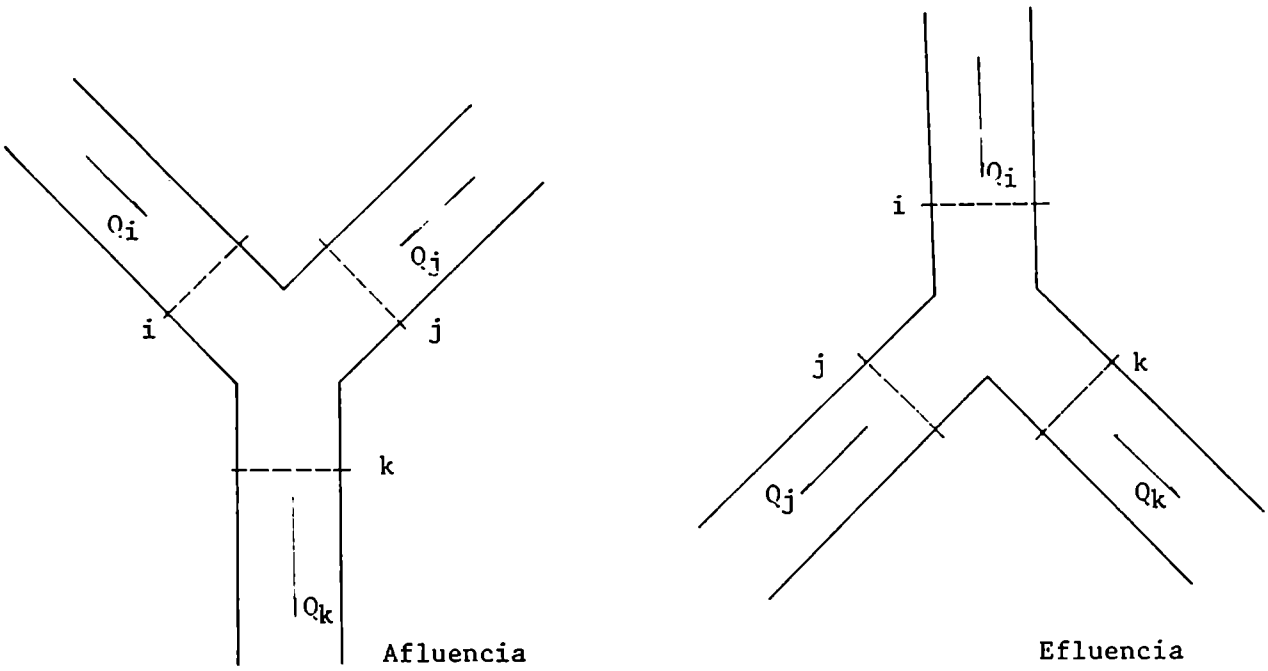


Figura 3: Confluencia de dos cauces fluviales

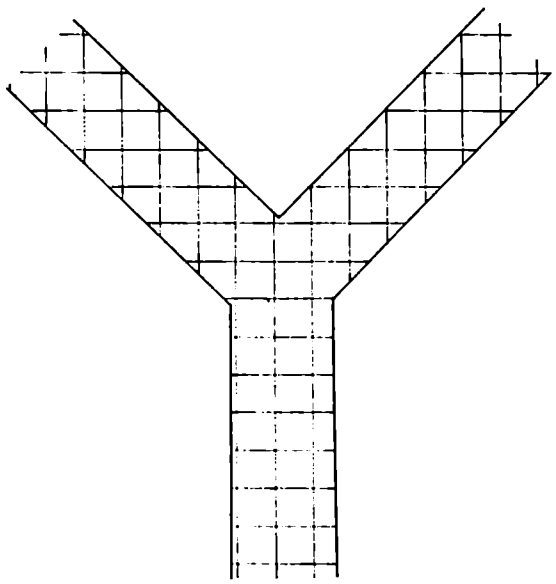


Figura 4: Discretización bidimensional de una confluencia

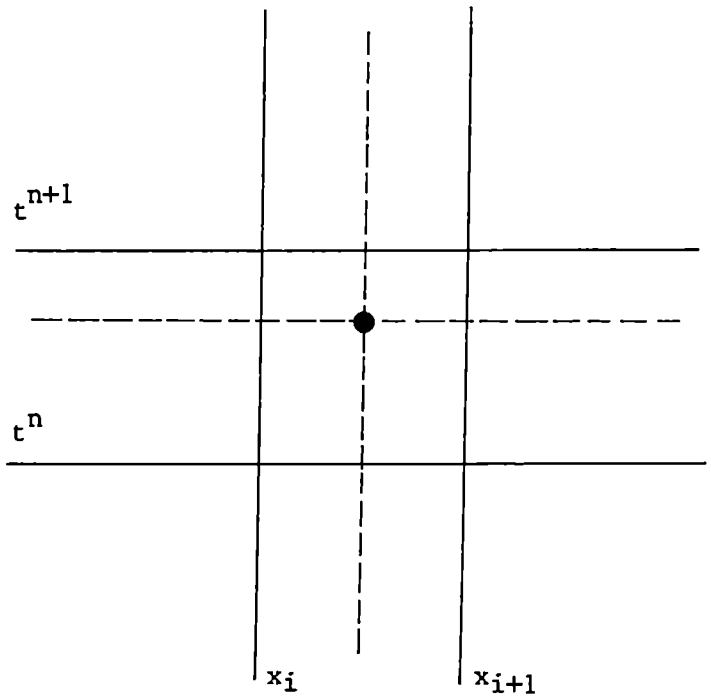


Figura 5: Esquema de cuatro puntos

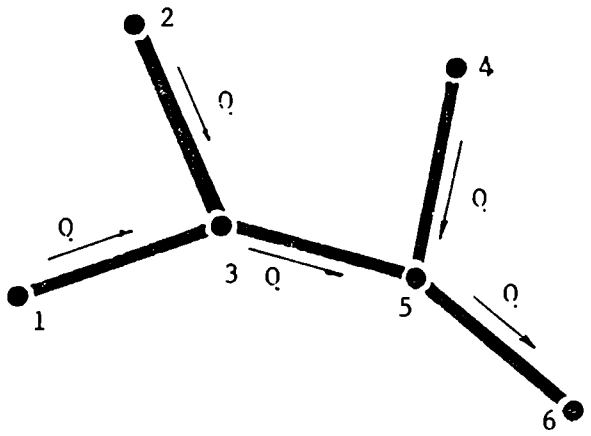


Figura 6: Red arborescente

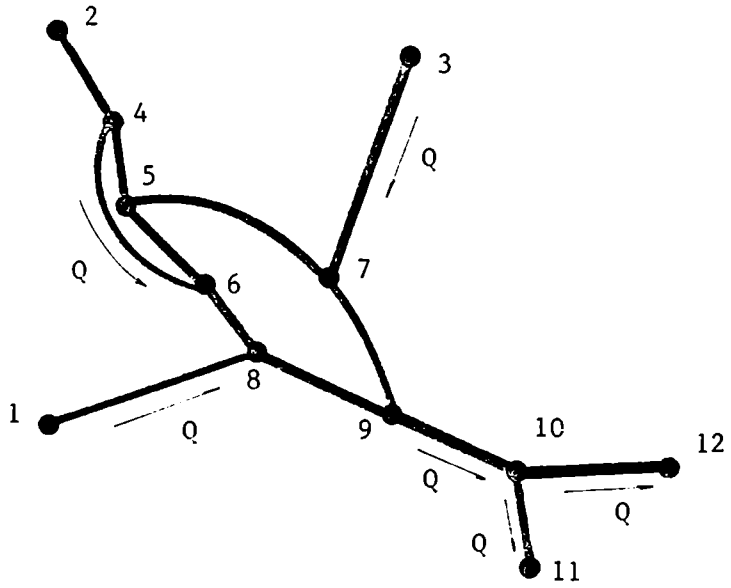


Figura 7: Red deltaica

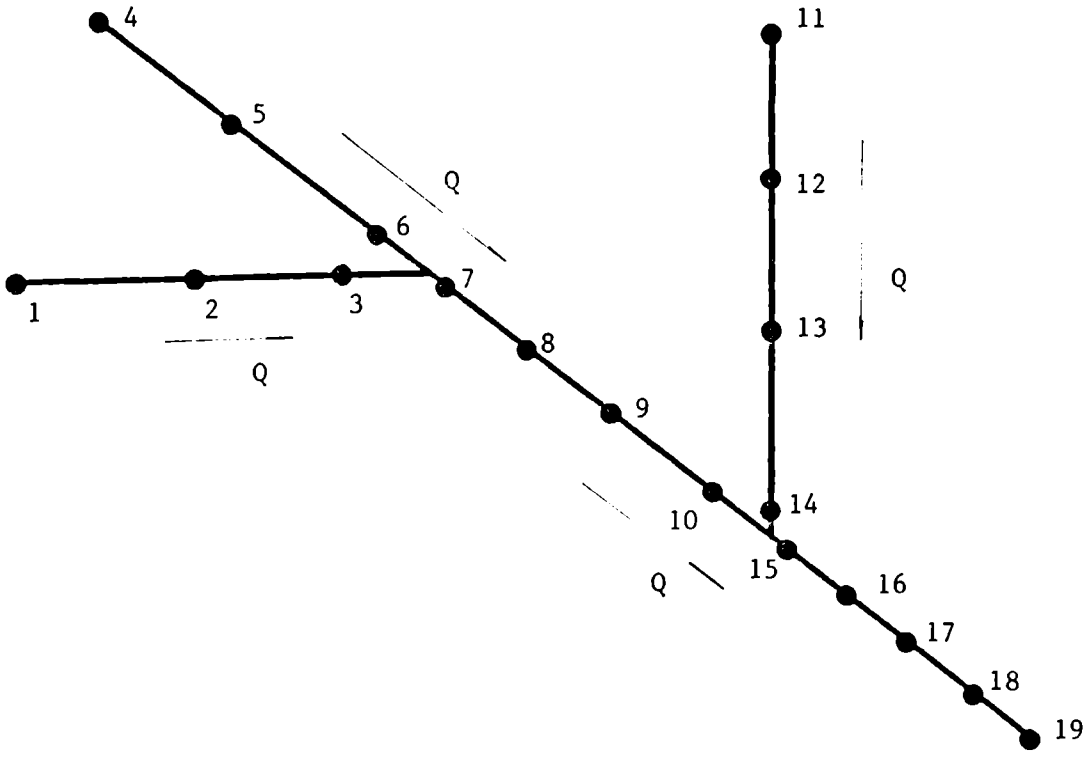


Figura 8: Numeración de los puntos de una red arborescente

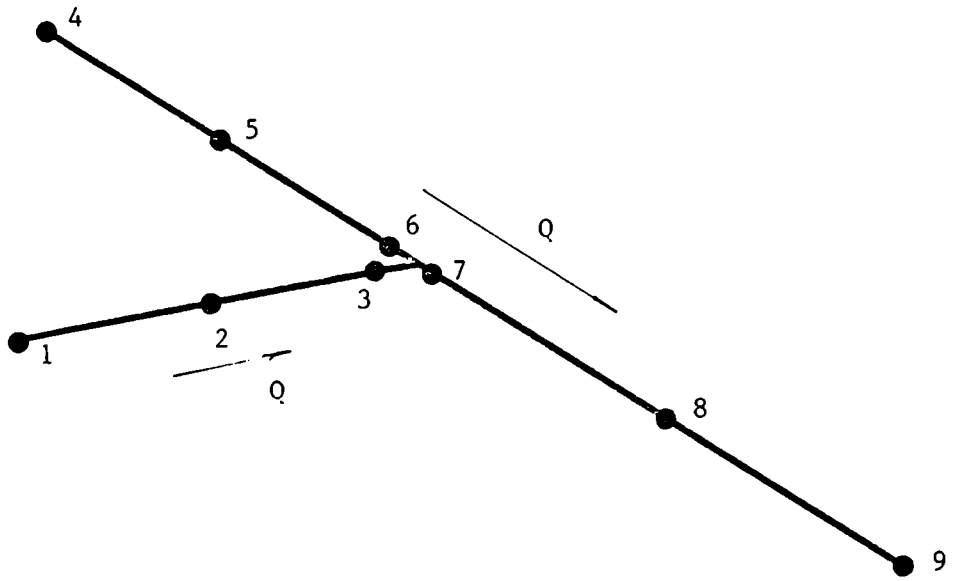


Figura 9: Red arborescente usada en el ejemplo

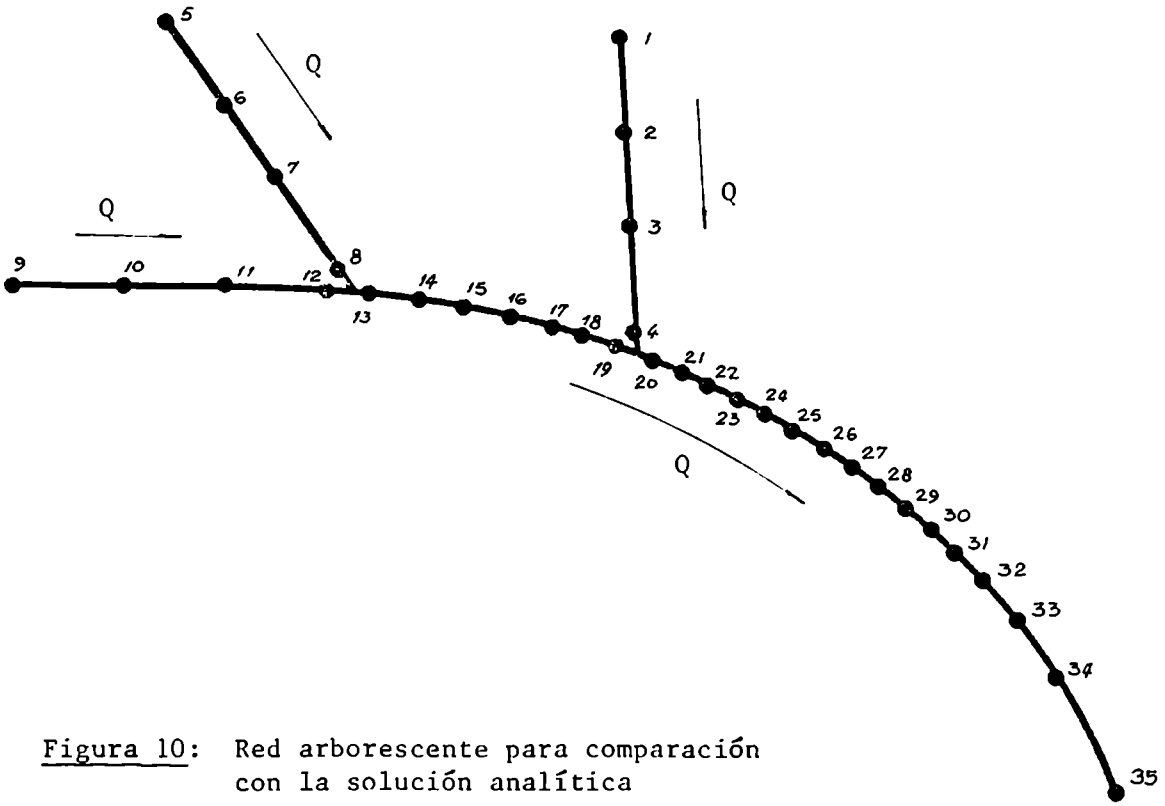


Figura 10: Red arborescente para comparación con la solución analítica

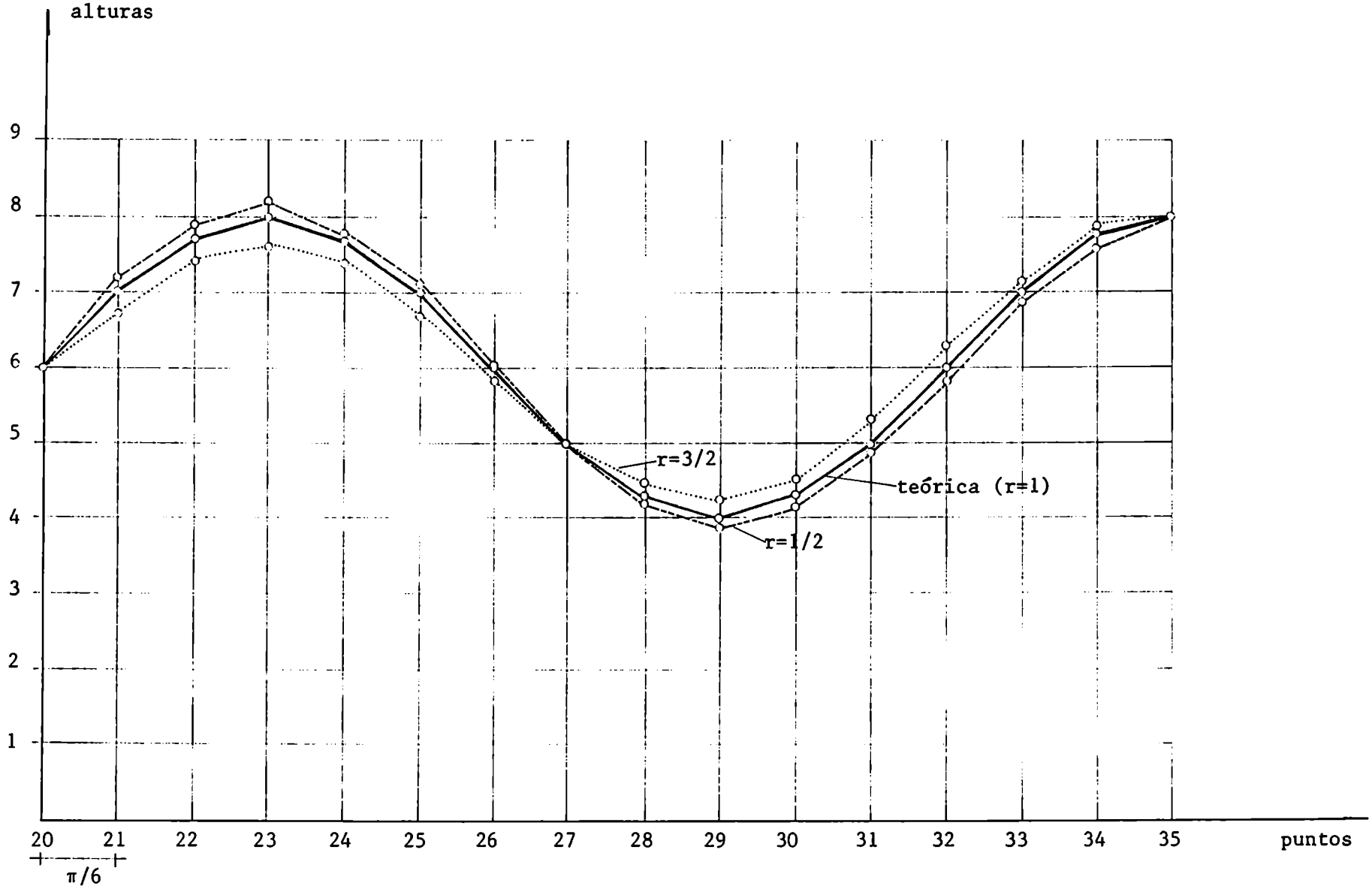


Figura 11: Comparación de alturas teóricas y calculadas para el modelo simplificado con distintos $\Delta t/\Delta x$

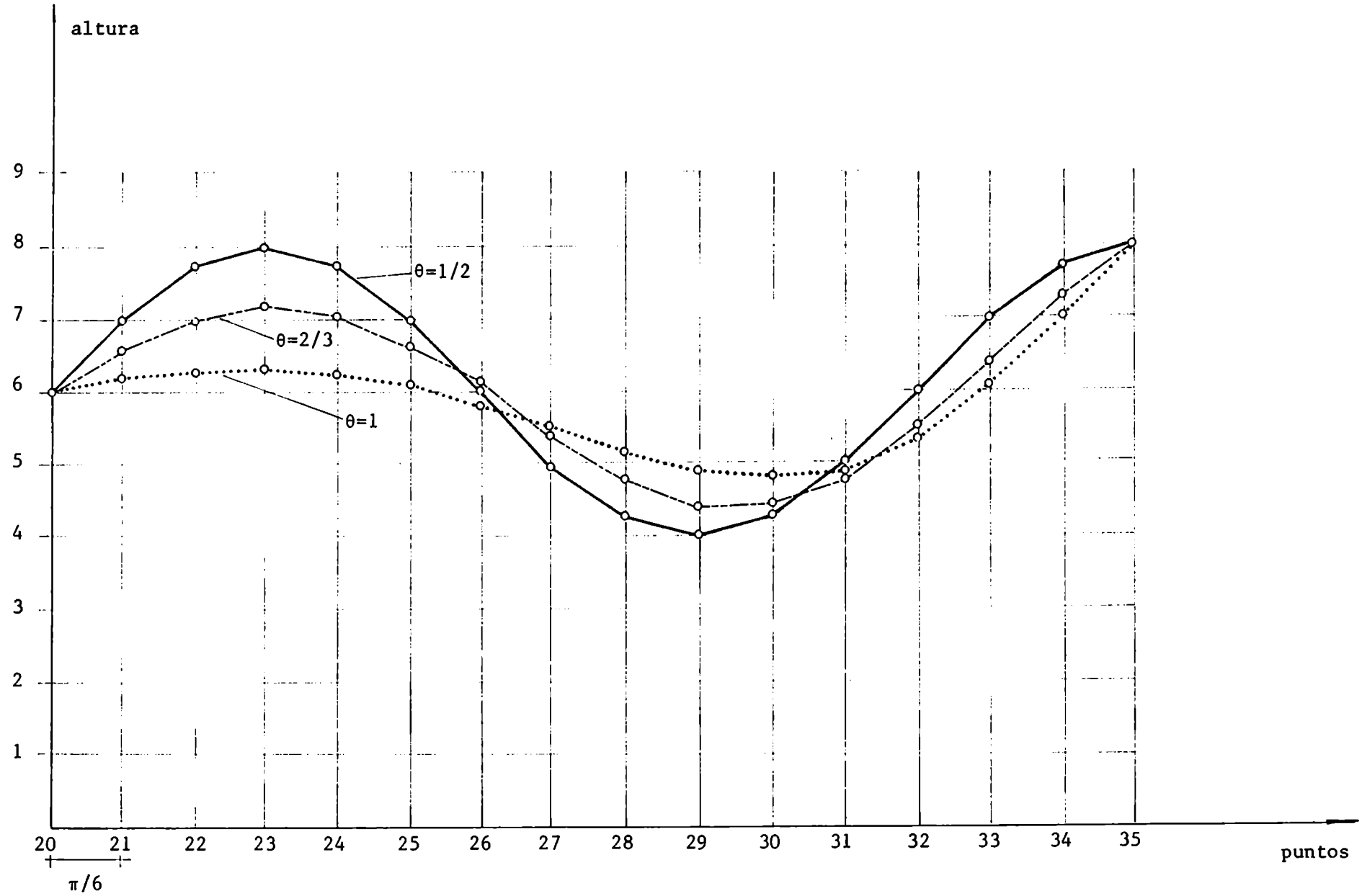


Figura 12: Comparación de alturas teóricas y numéricas con distintos valores de θ para el modelo simplificado

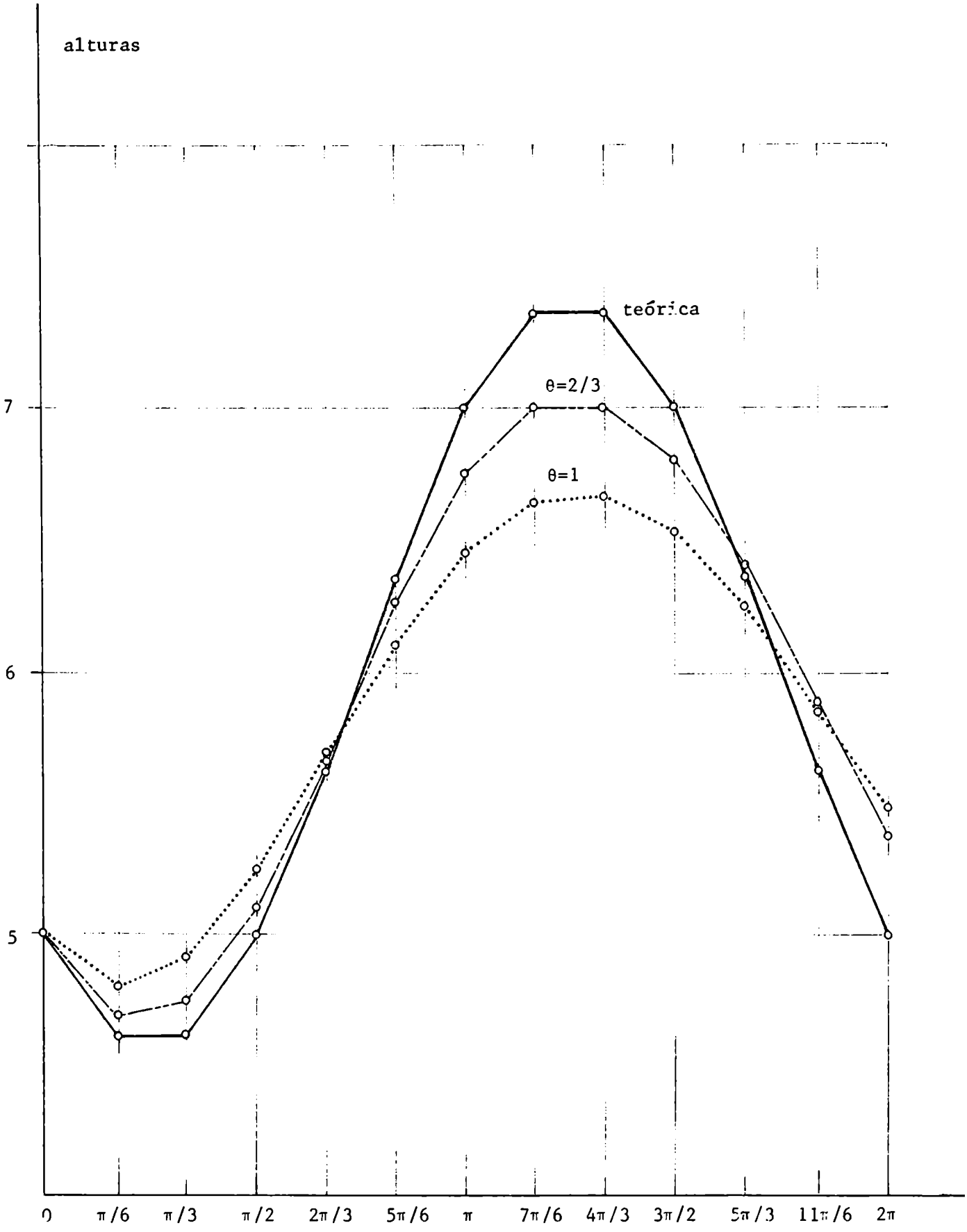


Figura 13: Atenuación de alturas en el punto 27 del modelo simplificado para distintos valores de θ

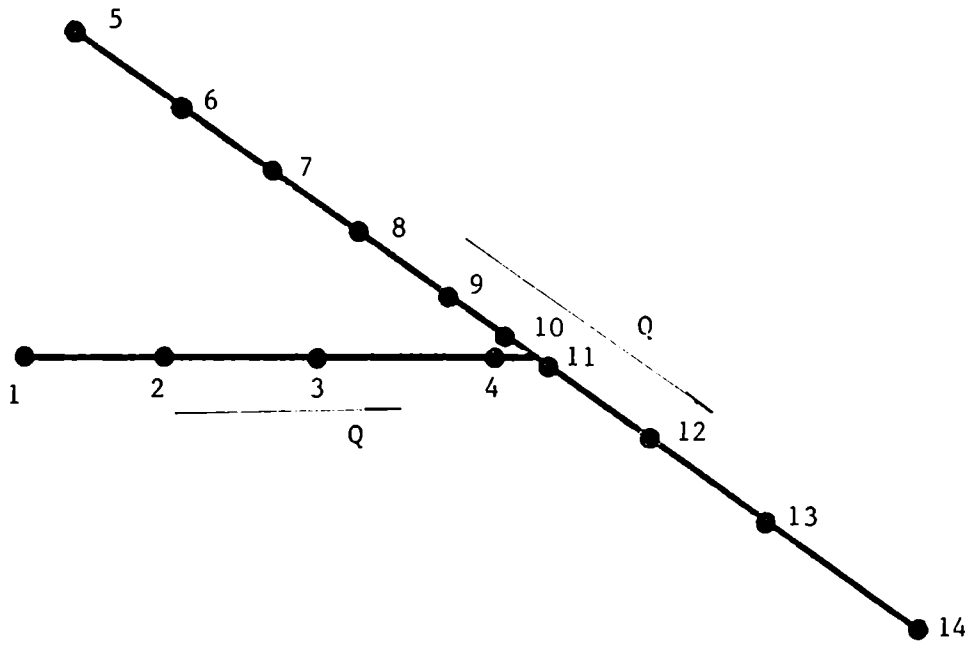


Figura 14: Red fluvial para pruebas del sistema arborescente

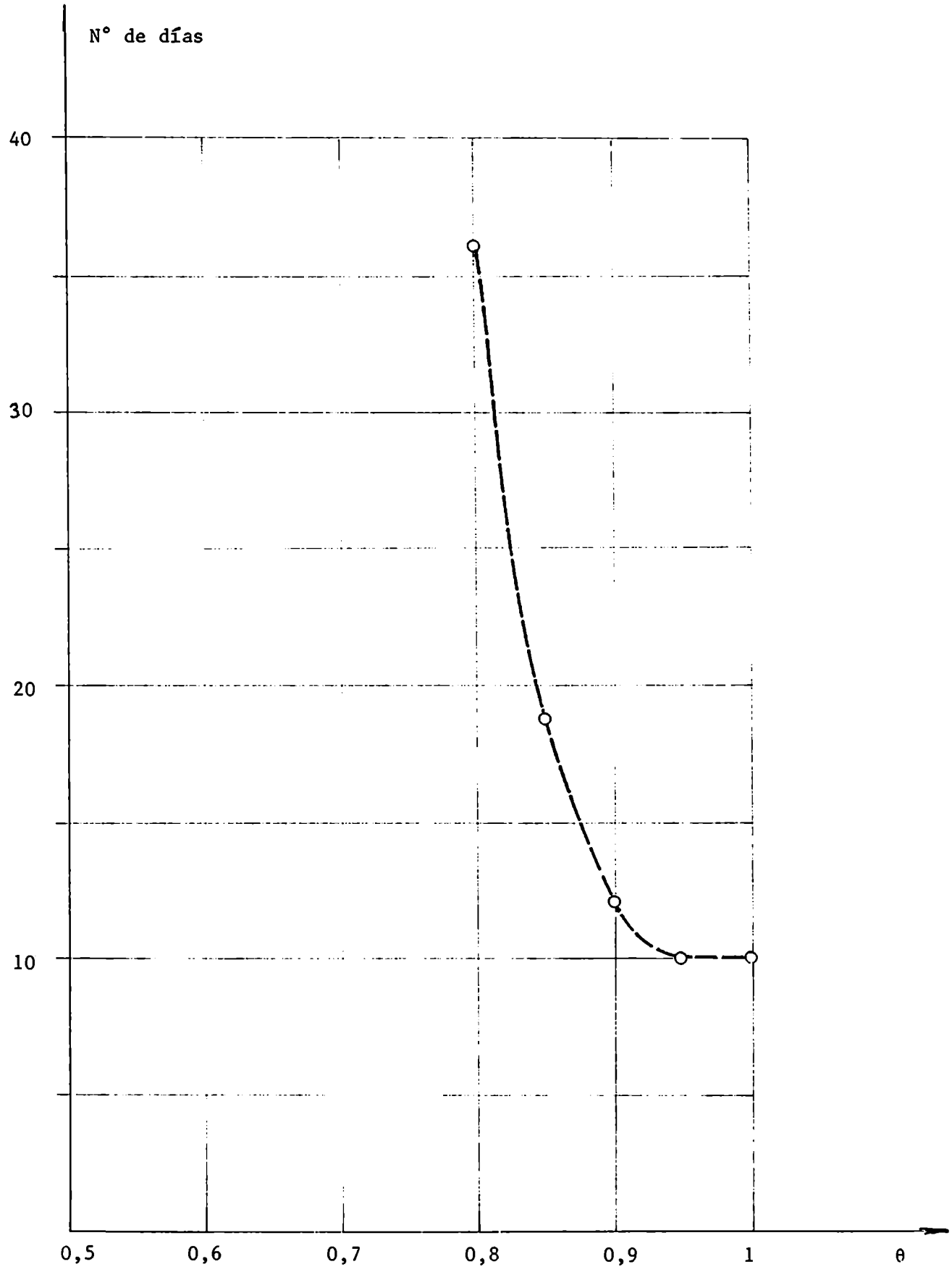


Figura 15: Número de días necesarios para estabilización en función de θ

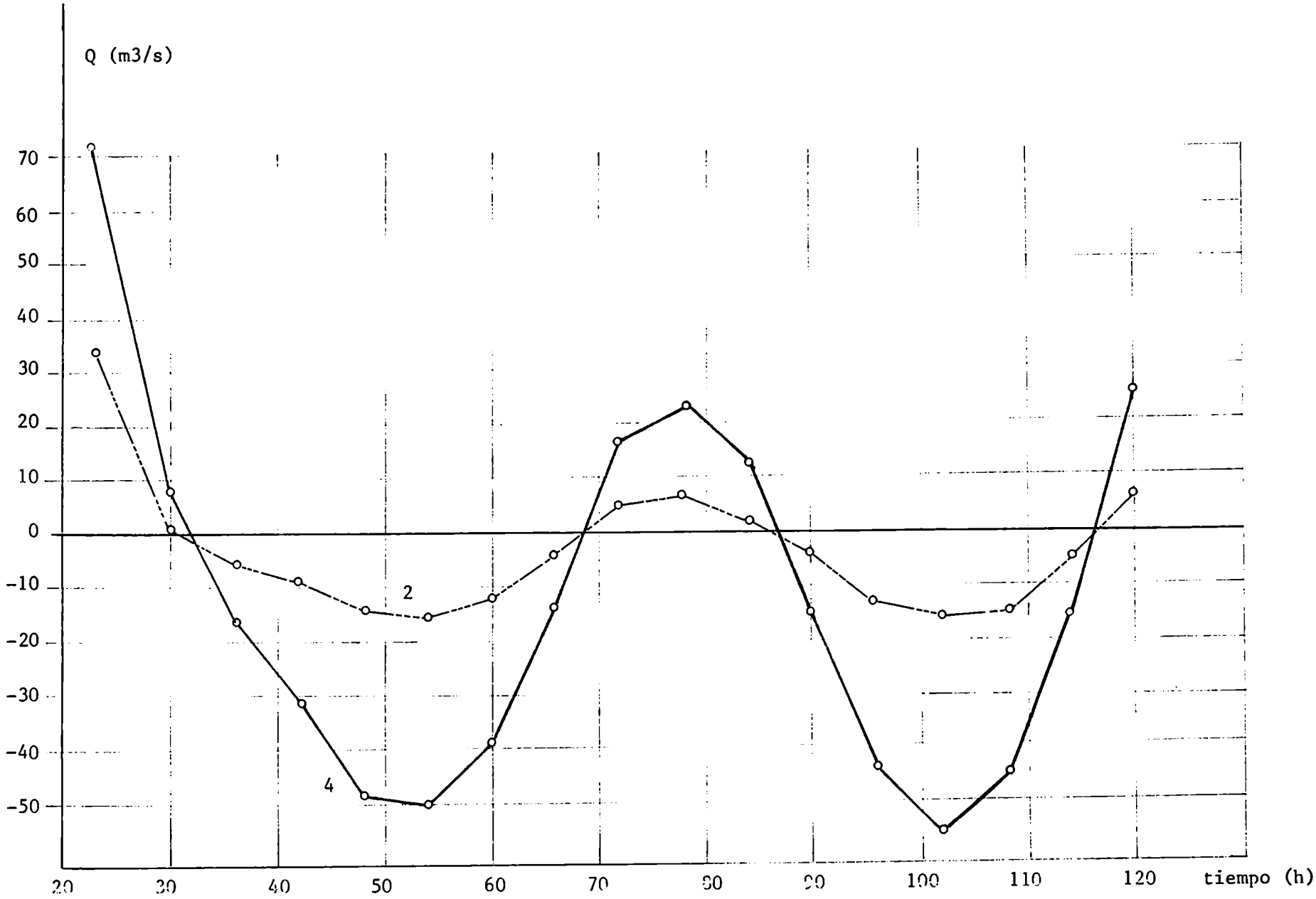


Figura 16: Caudales en los puntos 2 y 4 del afluente cerrado

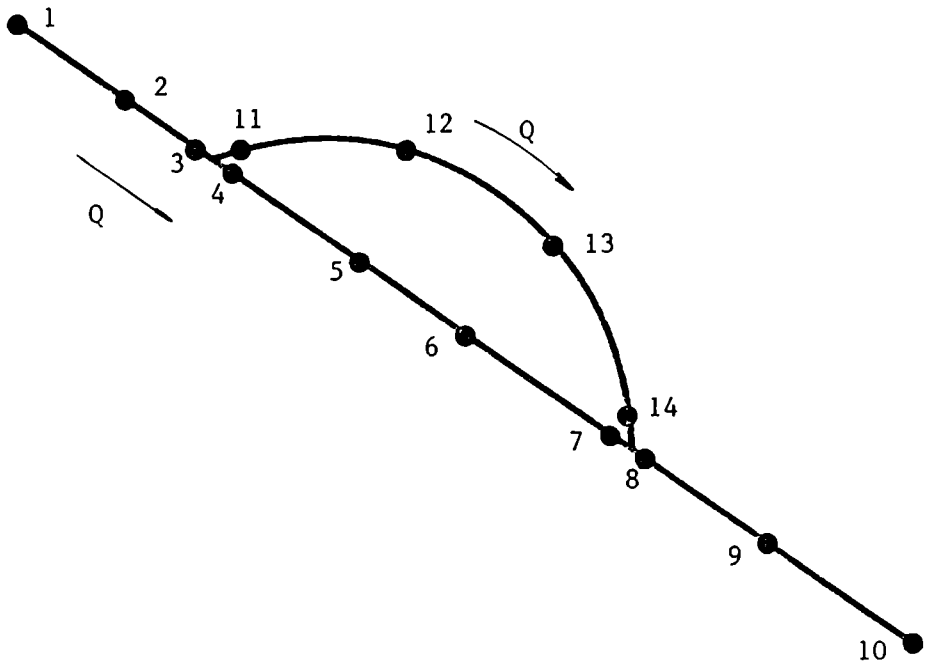


Figura 17 - Red deltaica elemental

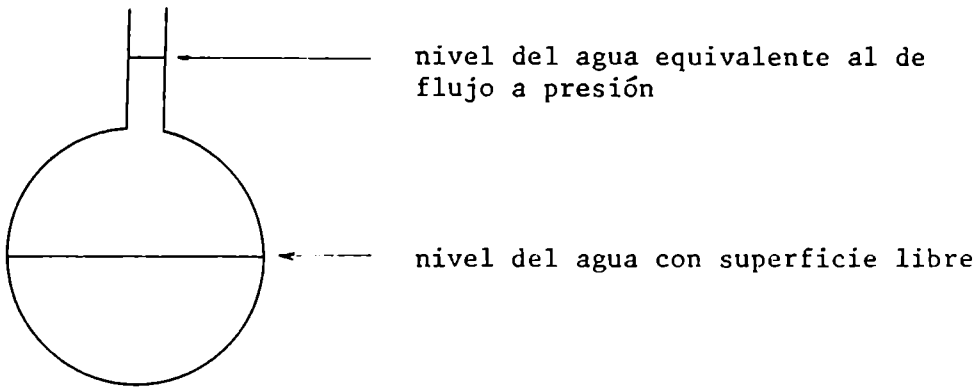


Figura 18: Sección transversal de una cañería cerrada

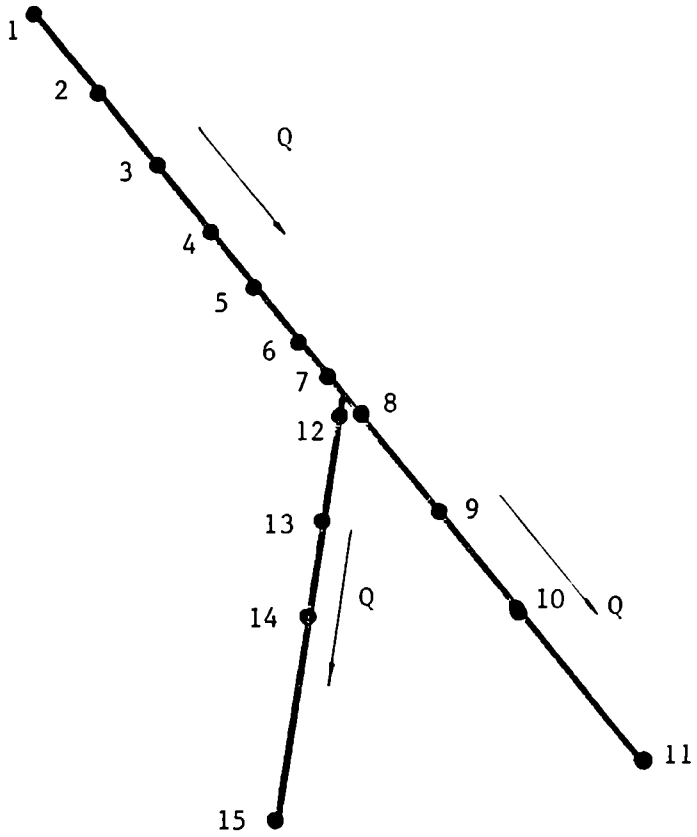


Figura 19: Sistema con efluente

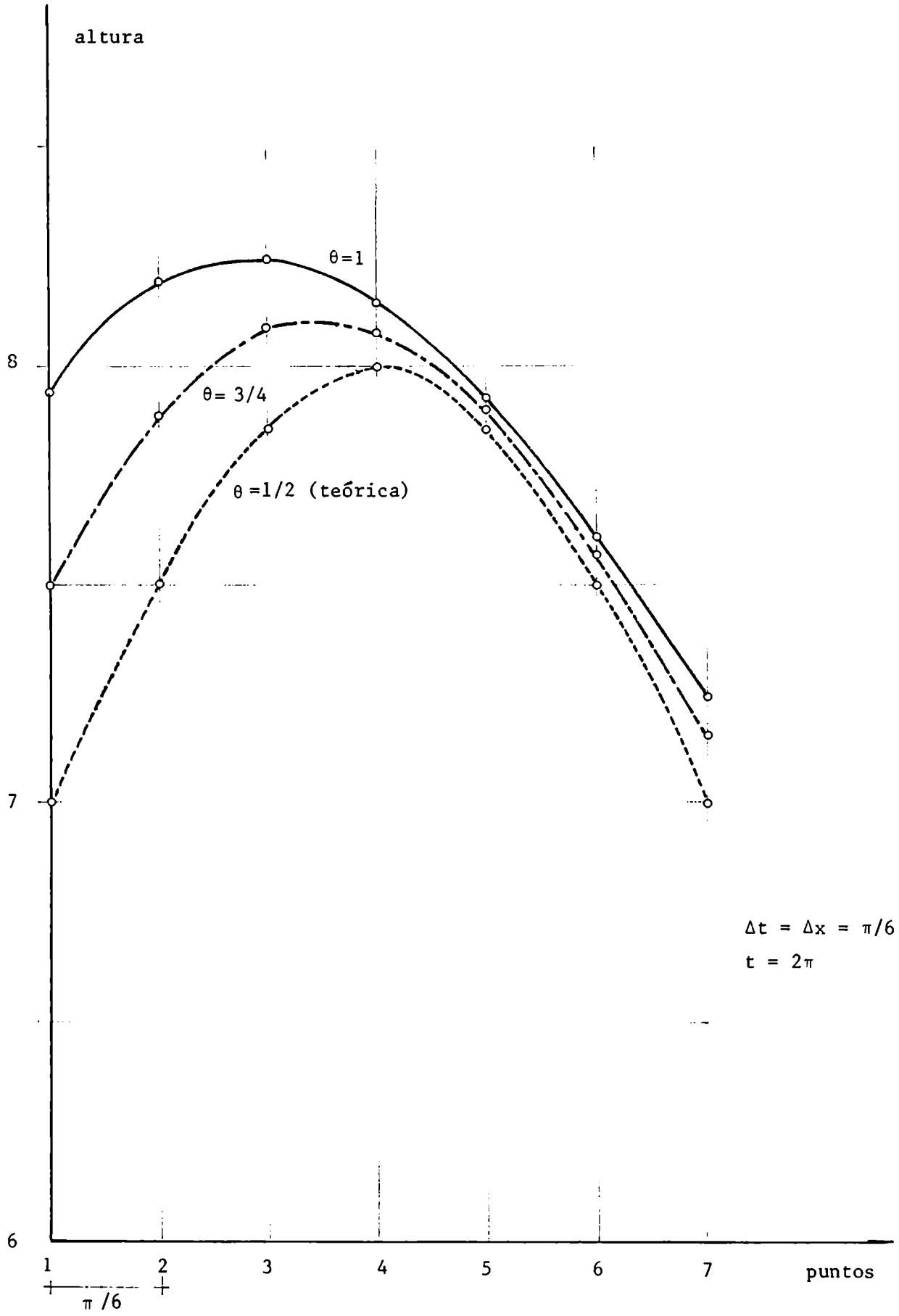


Figura 20: Comparación de alturas teóricas con calculadas con $\theta = 3/4$ y $\theta=1$ para el modelo simplificado en los puntos 1 a 7

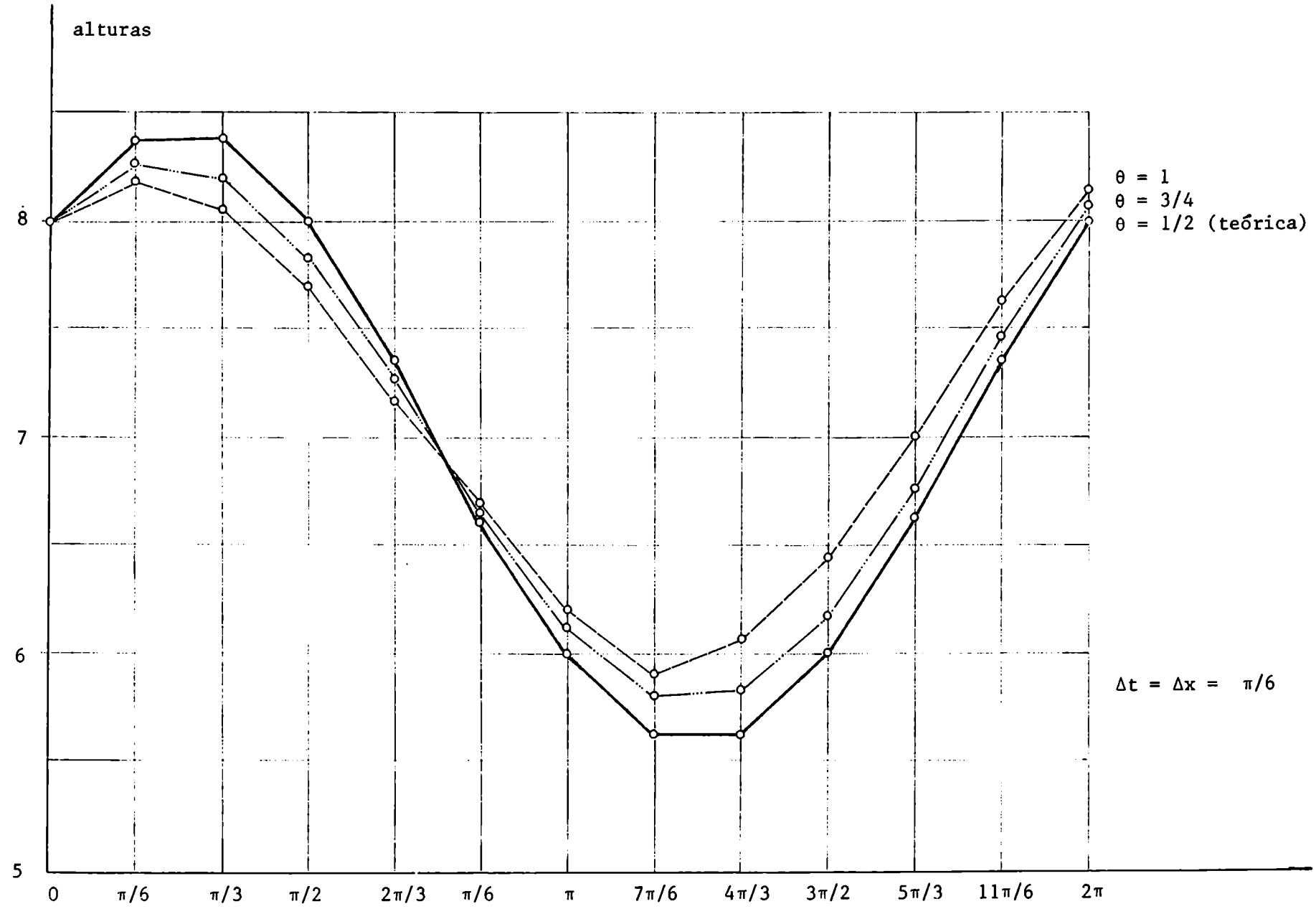


Figura 21: Comparación de alturas teóricas con calculadas con θ = 3/4 y θ = 1 en el punto 4 para el modelo simplificado

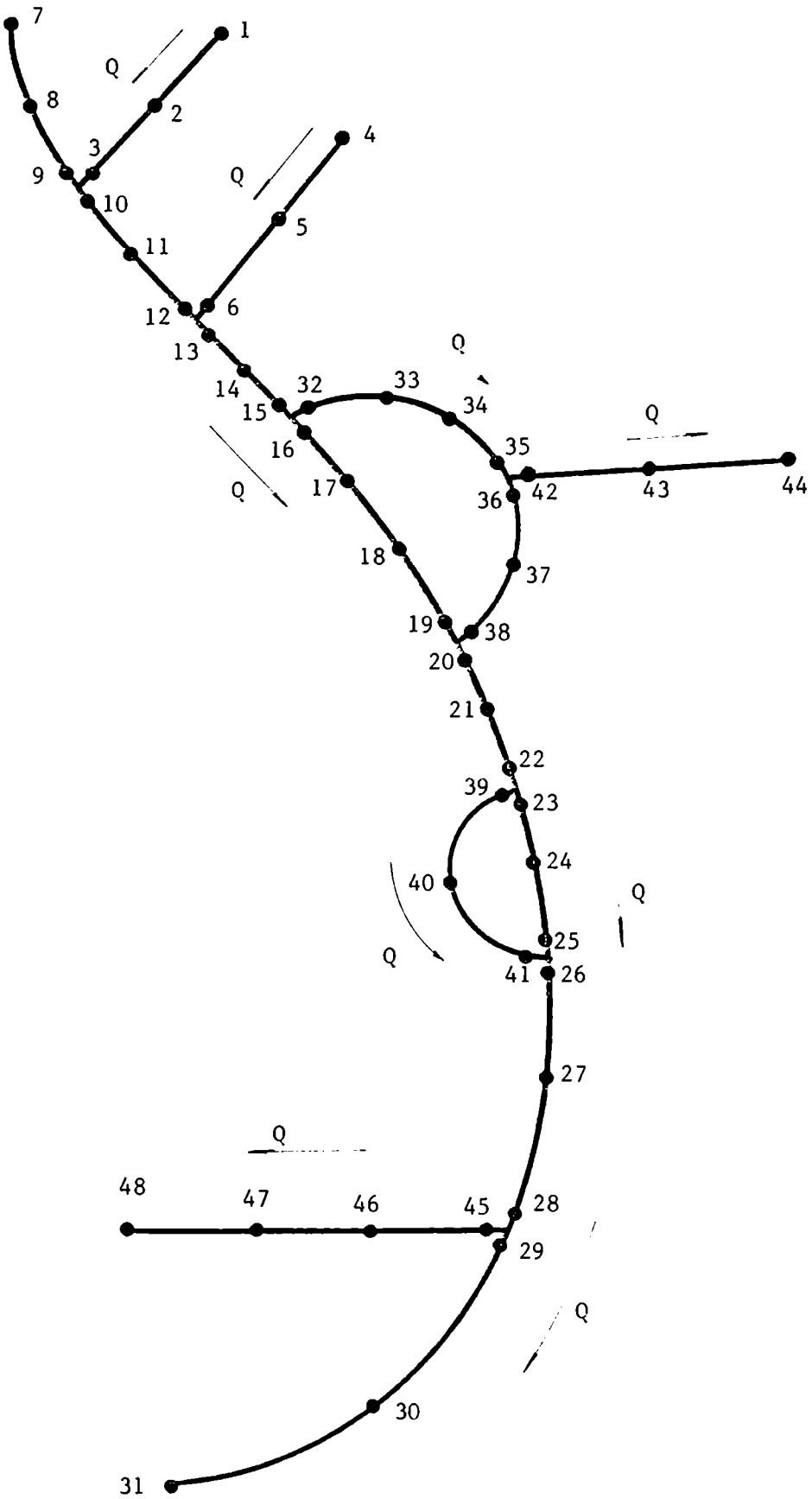


Figura 22: Red fluvial deltaica para experimentos

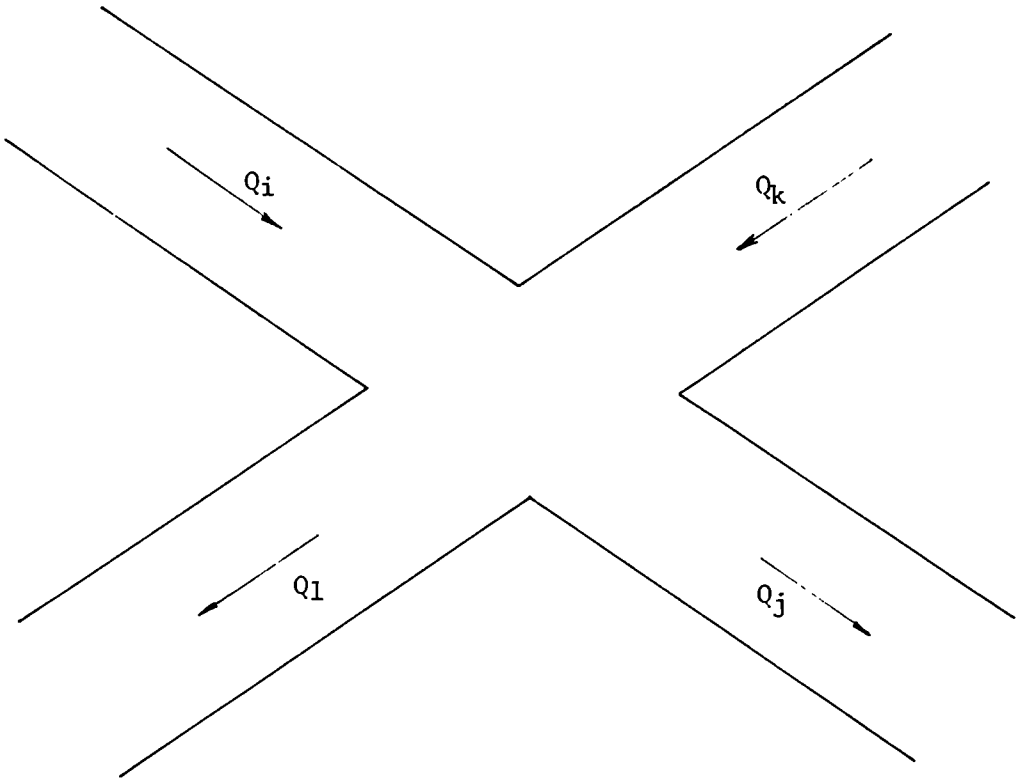


Figura 23: Cruce de canales

LISTADO DE PROGRAMAS

CUENSIMP
CUENCAS
DELTSIMP
DELTA
DELTA1
DELTA2

```
program cuensimp;

(* este programa resuelve el modelo de red fluvial con estructura
   arborescente para las ecuaciones diferenciales linealizadas que
   componen la ecuacion de la onda *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.5;      (* parametro de "implicitud" *)
        imprix=10;        (* maximo numero de intervalos entre impresiones *)
  type intnpmx=1..npmx;
       intndmx=1..ndmx;
       intimpr=0..imprix;
       array25=array[1..2,1..5] of real;
       arrayd=array[intndmx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;
       arraypx3=array[intnpmx,1..3] of real;
   var contorno,salida:text;unidad:string7;
       m,npuntos: intnpmx;  imp,impri:intimpr;
       ndat:intndmx; t,tfin,dt,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,Q,Z: arraypx;
       nombre: arrayp;
       codigo,ubica: arraypxi;
       contor: string[12] ;
       c1,c2:arraypx3;


function min(i1,i2:integer):integer;
begin
 if(i1<i2) then min:=i1 else min:=i2
end;
```

```
procedure leemodelo(var npuntos:intnpmx;var nombre:arrayp;
                    var xcoor:arraypx;
                    var codigo,ubica:arraypxi);

(* lee los datos fisicos,geometricos y topograficos de la red fluvial


  var red:text;modelo:string[12];titulo:string[80];
      m:intnpmx;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    read(red,titulo);
    writeln(salida,titulo);
    read(red,npuntos);
    readln(red);
    for m:=1 to npuntos do
      begin
        read(red,nombre[m],xcoor[m],codigo[m],ubica[m]);
        readln(red)
     end;
   close(red)
  end;

procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales en los puntos de discretizacion *)

  var iniciales: text;m:intnpmx;
      inicial:string[12];

  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:= 1 to npuntos do read(iniciales,Q[m]);
    for m:= 1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;
```

```
procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real;var unidad:string7);

(* este procedimiento indica cuales seran las condiciones de contorno

  var borde:integer; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= vel. dada');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/vel.');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Numero de condiciones de contorno ');
       readln(ndat);
       unidad:=' seg.  ';
       writeln(salida,
          ' Seccion  Coordenada (km)       Codigo
            ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
       writeln(salida, nombre[m]:8,xcoor[m]:12:3,
                        codigo[m]:16,ubica[m]:11);
       write('Tiempo inicial de calculo ');
       readln(tinic);
       write('Intervalo dt de calculo ');
       readln(dt);
       write('Tiempo final de calculo ');
       readln(tfin);
       write('Cada cuantos intervalos se imprime? ');
       readln(impri);
       if(impri=0) then impri:=1;
       writeln(salida,'Tiempo inicial ',tinic:9:2,unidad);
       writeln(salida,'Intervalo dt de calculo ',dt:7:2,unidad);
       writeln(salida,'Tiempo final ',tfin:9:2 ,unidad);
       writeln(salida,'Numero de condiciones de contorno ',ndat)
  end;
```

```
procedure precoefis(npuntos:intnpmx;Z:arraypx);

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
        if(Z[m]<0) then fracaso(nombre[m]);
   end;

procedure imprime(unidad:string7;
                  t:real;npuntos:intnpmx;nombre:arrayp;Z,Q:arraypx);

   (* imprime resultados en determinados instantes *)

  var m,m1,m2,nn: intnpmx;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:2, unidad);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Altura(m)');
        for m:= m1 to m2 do write(salida,Z[m]:7:3);
        writeln(salida);
        write(salida,'Vel.(m/s)');
        for m:= m1 to m2 do write(salida,Q[m]:7:3);
        writeln(salida);
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

   (* lee datos de contorno en instantes t *)

  var nd:intndmx;
  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;
```

```
procedure interdat(t:real;
                   ndat:intndmx;var dat,dat1,dat2:arrayd;var t1,t2:real

(* interpola linealmente datos de contorno en instantes t en funcion
   de los datos de que dispone en instantes t1 y t2 con t1<t<=t2  *)

  var nd:intndmx;tet1,tet2:real;

  begin
        while(t > t2+0.0001) do
           begin
             t1:=t2;
             dat1:=dat2;
             leedat(ndat,dat2,t2)
           end;
        tet1:= (t2-t)/(t2-t1);
        tet2:=1-tet1;
        for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;


procedure barrido(npuntos:intnpmx;teta,dts:real;
                  codigo,ubica:arraypxi;dat:arrayd;
                  var Q,Z:arraypx);

(* barrido descendente *)

  var l:integer;a:array25;c1,c2:arraypx3;m:intnpmx;

procedure coefis(m:intnpmx;dts,teta:real;
                  xcoor,Q,Z:arraypx; var a:array25);

(* calcula los coeficientes de las ecuaciones linealizadas
    y discretizadas para dos filas consecutivas *)

  var      dx,r,x1:real;i:integer;

  begin
    dx:=abs(xcoor[m+1]-xcoor[m]);
    r:=dts/dx;
    x1:=2*r*teta;
    a[1,1]:=1;
    a[1,2]:=-x1;
    a[1,3]:=1;
    a[1,4]:=x1;
    a[1,5]:=2*r*(Z[m]-Z[m+1]);
    a[2,1]:=-x1;
    a[2,2]:=1;
    a[2,3]:=x1;
    a[2,4]:=1;
    a[2,5]:=2*r*(Q[m]-Q[m+1])
  end;
```

```
procedure desciende(i:integer;a:array25;X1,X2,X3:real;var Y1,Y2,Y3:real);

  (* triangula la matriz fila a fila *)

var coef:real;

begin
  coef:=a[1+i,1+i]/X1;
  Y1:=a[1+i,2+i]-coef*X2;
  Y2:=a[1+i,3+i];
  Y3:=a[1+i,5]-coef*X3
end;

procedure comienzo(i:integer;dd:real;a:array25;var X1,X2,X3,Y1,Y2,Y3:real);

   (* arma las primeras ecuaciones del tramo *)

begin
  X1:=a[1,2-i];
  X2:=a[1,3];
  X3:=a[1,5]-dd*a[1,1+i];
  if(i=0) then  desciende(1,a,X1,X2,X3,Y1,Y2,Y3)
          else
                begin
                    Y1:=a[2,3];
                    Y2:=a[2,4];
                    Y3:=a[2,5]
                end
end;

procedure  banda(var a:array25);

           (* tridiagonaliza la matriz *)

var l:integer;coef:real;

begin
  coef:=a[2,1]/a[1,1];
  for l:= 2 to 5 do   a[2,l]:=a[2,l]-coef*a[1,l];
  coef:=a[1,4]/a[2,4];
  for l:=2 to 3 do a[1,l]:=a[1,l]-coef*a[2,l];
  a[1,5]:=a[1,5]-coef*a[2,5]
end;
```

```
procedure ascenso(codigo,ubica:arraypxi;npuntos:intnpmx;
                  dat:arrayd;c1,c2:arraypx3;var Q,Z:arraypx);

           (* barrido ascendente y asignacion de resultados *)

var dQ,dZ:arraypx;m:intnpmx;

begin
  if(codigo[npuntos]=91) then

                 (* m es punto extremo aguas abajo con caudal Q dado *)

        begin
          dQ[npuntos]:=dat[ubica[npuntos]]-Q[npuntos];
          dZ[npuntos]:=(c2[npuntos-1,3]-c2[npuntos-1,1]*dQ[npuntos])
                              /c2[npuntos-1,2]
        end
                           else
         begin

                 (* m es punto extremo aguas abajo con nivel Z dado *)

           dZ[npuntos]:=dat[ubica[npuntos]]-Z[npuntos];
           dQ[npuntos]:=(c2[npuntos-1,3]-c2[npuntos-1,2]*dZ[npuntos])
                              /c2[npuntos-1,1]
         end;
   for m:=npuntos-1 downto 1 do
        begin
          if(((codigo[m]=0) and (codigo[m+1]<>25) and (codigo[m+1]<>20))
                     or (codigo[m]=25) or (codigo[m]=20)) then

          (* m es punto interior de tramo o punto extremo cerrado
              aguas arriba perteneciente a nodo de afluencia
                {m-1,ubica[m],m} *)

            begin
              dZ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1];
              dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
            end
                             else
          if(codigo[m+1]=20 ) then       (* inyeccion lateral de caudal *)
             begin
               dZ[m]:=dZ[m+1];
               dQ[m]:=dQ[m+1]-dat[ubica[m+1]]+Q[m+1]-Q[m]
             end
                            else
           if(codigo[m+1]=25) then

   (* m es punto extremo cerrado aguas abajo perteneciente
        a nodo de confluencia {m,ubica[m+1],m+1}  *)

               begin
                 dZ[m]:=dZ[m+1];
                 dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
               end
                            else
```

```
        if(codigo[m]=21) then


    (* m es punto extremo cerrado aguas abajo perteneciente a nodo
       de afluencia  {ubica[m]-1,m,ubica[m] }  *)

            begin
              dZ[m]:=dZ[ubica[m]];
              dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
            end
                        else

    (* m es punto extremo abierto aguas arriba con caudal Q dado *)

        if(codigo[m]=11) then
            begin
             dQ[m]:=dat[ubica[m]]-Q[m];
             dZ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1]
            end
                       else
        if(codigo[m]=12) then

    (* m es punto extremo abierto aguas arriba con nivel Z dado *)

            begin
             dZ[m]:=dat[ubica[m]]-Z[m];
             dQ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1]
           end
    end;
    for m:=1 to npuntos do
      begin
        Q[m]:=Q[m]+dQ[m];
        Z[m]:=Z[m]+dZ[m]
      end
end;


  begin                                          (* barrido descendente *)
    for m:=1 to npuntos-1 do
      if((codigo[m+1]<>20) and (codigo[m]<>21) and (codigo[m+1]<>25)) then
          begin
            coefis(m,dts,teta,xcoor,Q,Z,a);
            if(codigo[m+2]=20) then
    for l:=1 to 2 do a[l,5]:=a[l,5]+a[l,3]*(dat[ubica[m+2]]-(Q[m+2]-Q[m+1]));

            banda(a);
            if(codigo[m]=11)then
                     comienzo(0,dat[ubica[m]]-Q[m],a,c1[m,1],c1[m,2],
                                  c1[m,3],c2[m,1],c2[m,2],c2[m,3])
                               else
             if(codigo[m]=12) then
                      comienzo(1,dat[ubica[m]]-Z[m],a,c1[m,1],c1[m,2],
                                   c1[m,3],c2[m,1],c2[m,2],c2[m,3])
                                else
             begin
                       desciende(0,a,c2[m-1,1],c2[m-1,2],c2[m-1,3],
                                     c1[m,1],c1[m,2],c1[m,3]);
                       desciende(1,a,c1[m,1],c1[m,2],c1[m,3],c2[m,1],
                                     c2[m,2],c2[m,3])
              end
```

```
            end


                                        else
          begin
           if(codigo[m+1]=25) then
              begin
                c2[m,1]:=1;
                for l:=2 to 3 do
  c2[m,l]:=c2[m-1,l]/c2[m-1,1]+c2[ubica[m+1]-1,l]/c2[ubica[m+1]-1,1]
              end
                              else
           if(codigo[m+1]=20 ) then for l:= 1 to 3 do c2[m,l]:=c2[m-1,l]
          end;
        ascenso(codigo,ubica,npuntos,dat,c1,c2,Q,Z)
end;

  begin                                                          (* cuensimp *)
    write('Nombre del archivo de resultados ');
    readln(contor);
    assign(salida,contor);
    rewrite(salida);
    leemodelo(npuntos,nombre,xcoor,codigo,ubica);
    leeinic(npuntos,Q,Z);
    leeborde(npuntos,nombre,xcoor,ubica,codigo,ndat,impri,tinic,dt,
               tfin,unidad);
    imp:=0;
    write('Archivo de condiciones de contorno ');
    readln(contor);
    assign(contorno,contor);
    reset(contorno);
    precoefis(npuntos,Z);
    imprime(unidad,tinic,npuntos,nombre,Z,Q);
    leedat(ndat,dat1,t1);
    leedat(ndat,dat2,t2);
    t:= tinic+dt;
    imp:=1;
    while ( t <= tfin+0.0001) do
                    begin
                      interdat(t,ndat,dat,dat1,dat2,t1,t2);
                      barrido(npuntos,teta,dt,codigo,
                              ubica,dat,Q,Z);
                      precoefis(npuntos,Z);
                      if(imp=impri) then
                                      begin
                                        imprime(unidad,t,
                                                npuntos,nombre,Z,Q);
                                        imp:=0
                                      end;
                      t:=t+dt;
                      imp:=imp+1
                    end;
    close(salida);
    close(contorno)
  end.
```

```
program cuencas;

(* este programa resuelve el modelo de red fluvial con estructura
   arborescente *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.85;     (* parametro de "implicitud" *)
        ntabx=2;          (* maximo numero de elementos de las tablas *)
        imprix=1000;        (* maximo numero de intervalos entre impresiones

  type intnpmx=1..npmx;
       intntabx=1..ntabx;
       intndmx=1..ndmx;
       intimpr=0..imprix;
       array25=array[1..2,1..5] of real;
       arrayd=array[intndmx] of real;
       arraynt=array[intnpmx,intntabx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       arrayt=array[intntabx] of real;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;
       arraypx3=array[intnpmx,1..3] of real;
   var contorno,salida:text;unidad:string7;
       m,npuntos: intnpmx;  imp,impri:intimpr; ntab:intntabx;
       ndat:intndmx; t,tfin,dt,dts,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,cota,Q,Z,S,D,B,derD: arraypx;
       nombre: arrayp;
       codigo,ubica: arraypxi;
       contor: string[12] ;tablaH:arrayt;
       tablaB,tablaD:arraynt;
       c1,c2:arraypx3;


function min(i1,i2:integer):integer;
begin
 if(i1<i2) then min:=i1 else min:=i2
end;
```

```
procedure leemodelo(var npuntos:intnpmx;var ntab:intntabx;var nombre:arra
                    var tablaH:arrayt;var xcoor,cota:arraypx;
                    var codigo,ubica:arraypxi;var tablaB,tablaD:arraynt);

(* lee los datos fisicos,geometricos y topograficos de la red fluvial *)

  var red:text;modelo:string[12];titulo:string[80];
      k:intntabx;m:intnpmx;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    read(red,titulo);
    write(salida,titulo);
    read(red,ntab,npuntos);
    for k:=1 to ntab do read(red,tablaH[k]);readln(red);
    writeln(salida,'    Tabla de alturas (en m)');
    for k:= 1 to ntab do writeln(salida,k:2,tablaH[k]:12:2);
    for m:=1 to npuntos do
      begin
        read(red,nombre[m],xcoor[m],cota[m],codigo[m],ubica[m]);
        writeln(salida,'Seccion ',m,' ',nombre[m],
                  '  Coordenada (km)  ',xcoor[m]:8:2);
        writeln(salida,'      Cota de referencia (m) ',cota[m]:8:2,
                  '  Codigo ',codigo[m]:3,' relacionado con ',ubica[m]:3);
        writeln(salida,' Tabla de anchos (en m)  ',
                  '      Tabla de coeficientes de conduccion (en 1000m3/s)')
        xcoor[m]:=xcoor[m]*1000;
        for k:=1 to ntab do read(red,tablaB[m,k]);
        for k:=1 to ntab do
            begin
              read(red,tablaD[m,k]);
              writeln(salida,'       ',tablaB[m,k]:10:2,'
                  tablaD[m,k]:12:3);
              tablaD[m,k]:=tablaD[m,k]*1000.
            end;
        readln(red)
     end;
   close(red)
  end;

procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales en los puntos de discretizacion *)

  var iniciales: text;m:intnpmx;
      inicial:string[12];

  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:= 1 to npuntos do read(iniciales,Q[m]);
    for m:= 1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;
```

```
procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor,cota:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real;var unidad:string7);

(* este procedimiento indica cuales seran las condiciones de contorno *)

  var borde:integer; tiem: char; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= caudal dado');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/caudal');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Unidad de tiempo (d,h,m) ');
       readln(tiem);
       write('Numero de condiciones de contorno ');
       readln(ndat);
       if(tiem='d') then unidad:=' dias'
            else if(tiem='h') then unidad:=' horas'
                 else unidad:=' minutos';
       writeln(salida,'       Resumen del modelo');
       writeln(salida,
         ' Seccion  Coordenada (km)    Cota de referencia (m)   Codigo
           ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
         writeln(salida, nombre[m]:8,xcoor[m]/1000.0:12:3,
                         cota[m]:27:2,codigo[m]:12,ubica[m]:11);
       write('Tiempo inicial de calculo ');
       readln(tinic);
       write('Intervalo dt de calculo ');
       readln(dt);
       write('Tiempo final de calculo ');
       readln(tfin);
       write('Cada cuantos intervalos se imprime? ');
       readln(impri);
       if(impri=0) then impri:=1;
       writeln(salida,'El tiempo se mide en',unidad,'.');
       writeln(salida,'Tiempo inicial ',tinic:9:2,unidad);
       writeln(salida,'Intervalo dt de calculo ',dt:7:2,unidad);
       writeln(salida,'Tiempo final ',tfin:9:2 ,unidad);
       writeln(salida,'Numero de condiciones de contorno ',ndat)
  end;
```

```
procedure precoefis(npuntos:intnpmx;ntab:intntabx;Z,cota:arraypx;
                    tablaH:arrayt;tablaB,tablaD:arraynt;
                    var B,derD,S,D:arraypx);

 (* calcula areas, anchos,coeficientes de conduccion y derivadas
    de coeficientes de conduccion en cada instante t *)

  var h,dz:real;m:intnpmx;k,k1:intntabx;

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
      begin
        S[m]:=0;
        h:= Z[m]-cota[m];
        if(h<0) then fracaso(nombre[m]);
        k:= 1;
        repeat k:=k+1 until (h<tablaH[k]) or (k=ntab);
        dz:= h-tablaH[k-1];
        B[m]:= tablaB[m,k-1]+dz*(tablaB[m,k]-tablaB[m,k-1])/(tablaH[k]-
                                tablaH[k-1]);
                                   (* B es el ancho superficial *)
        derD[m]:= (tablaD[m,k]-tablaD[m,k-1])/(tablaH[k]-tablaH[k-1]);
                       (* derD es la derivada del coeficiente de conduccion
                             respecto del nivel Z *)
        if(k>2)then for k1:=2 to k-1 do
           S[m]:=S[m]+(tablaB[m,k1]+tablaB[m,k1-1])*(tablaH[k1]-tablaH[k1-1]

        S[m]:=0.5*((B[m]+tablaB[m,k-1])*dz+S[m]);
                             (* S es el area mojada *)
        D[m]:=tablaD[m,k-1]+dz*derD[m]
                             (* D es el coeficiente de conduccion *)
      end
   end;
```

```
procedure imprime(unidad:string7;t:real;npuntos:intnpmx;nombre:arrayp;
                  Z,Q,S:arraypx);

      (* imprime resultados en determinados instantes *)

  var m,m1,m2,nn: intnpmx;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:2, unidad);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Cota  (m)');
        for m:= m1 to m2 do write(salida,Z[m]:7:2);
        writeln(salida);
        write(salida,'Q  (m3/s)');
        for m:= m1 to m2 do write(salida,Q[m]:7:0);
        writeln(salida);
        write(salida,'Vel.(m/s)');
        for m:= m1 to m2 do write(salida,Q[m]/S[m]:7:2)
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

     (* lee datos de contorno en determinados instantes *)

  var nd:intndmx;
  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;

procedure interdat(t:real;ndat:intndmx;
             var dat,dat1,dat2:arrayd;var t1,t2:real);

          (* interpola linealmente datos de contorno en instantes t
             en funcion de los datos de que dispone en instantes t1 y t2
             con t1 < t <= t2  *)

  var nd:intndmx;tet1,tet2:real;

  begin
        while(t > t2+0.0001) do
           begin
             t1:=t2;
             dat1:=dat2;
             leedat(ndat,dat2,t2)
           end;
        tet1:= (t2-t)/(t2-t1);
        tet2:=1-tet1;
        for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;
```

```
procedure barrido(npuntos:intnpmx;teta,dts:real;S,B,D,derD:arraypx;
                  codigo,ubica:arraypxi;dat:arrayd;
                  var Q,Z:arraypx);

         (* barrido descendente *)

  var l:integer;a:array25;c1,c2:arraypx3;m:intnpmx;

procedure coefis(m:intnpmx;dts,teta:real;
                  xcoor,Q,Z,S,D,B,derD:arraypx; var a:array25);

              (* calcula los coeficientes de las ecuaciones de
         Saint-Venant discretizadas para dos filas consecutivas *)


  var      dx,xc,x1,x2,x3,x4,efe:real;i:integer;

  begin
    dx:=abs(xcoor[m+1]-xcoor[m]);
    xc:=dx/(19.62*dts);
    x1:=dx*abs(Q[m])/sqr(D[m]);
    x2:=dx*abs(Q[m+1])/sqr(D[m+1]);
    x3:=Q[m]/(9.81*sqr(S[m]));
    x4:=Q[m+1]/(9.81*sqr(S[m+1]));
    efe:=dx/(2*teta*dts);
    a[1,1]:=-1;
    a[1,2]:=efe*B[m];
    a[1,3]:=1;
    a[1,4]:=efe*B[m+1];
    a[1,5]:=(Q[m]-Q[m+1])/teta;
    a[2,1]:=teta*(x1-x3)+xc/S[m];
    a[2,2]:=-teta-teta*x1*Q[m]*derD[m]/D[m]
            +(teta*x3-xc/S[m])*Q[m]*B[m]/S[m];
    a[2,3]:=teta*(x2+x4)+xc/S[m+1];
    a[2,4]:=teta-teta*x2*Q[m+1]*derD[m+1]/D[m+1]-
             (teta*x4+xc/S[m+1])*Q[m+1]*B[m+1]/S[m+1];
    a[2,5]:=Z[m]-Z[m+1]-0.5*(x1-x3)*Q[m]-(x2+x4)*Q[m+1]
  end;

procedure desciende(i:integer;a:array25;X1,X2,X3:real;var Y1,Y2,Y3:real);
var coef:real;

                (* triangula la matriz fila a fila  *)
begin
  coef:=a[1+i,1+i]/X1;
  Y1:=a[1+i,2+i]-coef*X2;
  Y2:=a[1+i,3+i];
  Y3:=a[1+i,5]-coef*X3
end;
```

```
procedure comienzo(i:integer;dd:real;a:array25;var X1,X2,X3,Y1,Y2,Y3:real)

      (* arma las primeras ecuaciones del tramo  *)

begin
  X1:=a[1,2-i];
  X2:=a[1,3];
  X3:=a[1,5]-dd*a[1,1+i];
  if(i=0) then  desciende(1,a,X1,X2,X3,Y1,Y2,Y3)
          else
                begin
                    Y1:=a[2,3];
                    Y2:=a[2,4];
                    Y3:=a[2,5]-dd*a[2,2]
                end
end;

procedure  banda(var a:array25);

          (* tridiagonaliza los pares de ecuaciones  *)

var l:integer;coef:real;

begin
  coef:=a[2,1]/a[1,1];
  for l:= 2 to 5 do   a[2,l]:=a[2,l]-coef*a[1,l];
  coef:=a[1,4]/a[2,4];
  for l:=2 to 3 do a[1,l]:=a[1,l]-coef*a[2,l];
  a[1,5]:=a[1,5]-coef*a[2,5]
end;


procedure ascenso(codigo,ubica:arraypxi;npuntos:intnpmx;
                  dat:arrayd;c1,c2:arraypx3;var Q,Z:arraypx);

              (* barrido ascendente y asignacion de variables *)

var dQ,dZ:arraypx;m:intnpmx;

begin
  if(codigo[npuntos]=91) then

    (* m es punto extremo aguas abajo con condicion de contorno caudal Q

        begin
          dQ[npuntos]:=dat[ubica[npuntos]]-Q[npuntos];
          dZ[npuntos]:=(c2[npuntos-1,3]-c2[npuntos-1,1]*dQ[npuntos])
                          /c2[npuntos-1,2]
        end
                        else

    (* m es punto extremo aguas abajo con condicion de contorno nivel Z *)

         begin
           dZ[npuntos]:=dat[ubica[npuntos]]-Z[npuntos];
           dQ[npuntos]:=(c2[npuntos-1,3]-c2[npuntos-1,2]*dZ[npuntos])
                          /c2[npuntos-1,1]
         end;
```

```
for m:=npuntos-1 downto 1 do
     begin
       if(((codigo[m]=0) and (codigo[m+1]<>25) and (codigo[m+1]<>20))
                  or (codigo[m]=25) or (codigo[m]=20)) then

         (* m es punto interior de tramo o punto extremo
            cerrado aguas arriba perteneciente a nodo
            de afluencia {m-1,ubica[m],m} *)

        begin
          dZ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1];
          dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
        end
                          else
      if(codigo[m+1]=20 ) then    (* inyeccion lateral de caudal *)
         begin
           dZ[m]:=dZ[m+1];
           dQ[m]:=dQ[m+1]-dat[ubica[m+1]]+Q[m+1]-Q[m]
         end
                        else
       if(codigo[m+1]=25) then

                 (* m es punto extremo cerrado aguas abajo
                   perteneciente a nodo de afluencia
                   {m,ubica[m],m+1}  *)

          begin
            dZ[m]:=dZ[m+1];
            dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
          end
                        else
       if(codigo[m]=21) then

                 (* m es punto extremo cerrado aguas abajo
                    perteneciente a nodo de afluencia
                    {ubica[m]-1,m,ubica[m]}  *)

          begin
            dZ[m]:=dZ[ubica[m]];
            dQ[m]:=(c2[m-1,3]-c2[m-1,2]*dZ[m])/c2[m-1,1]
          end
                        else
       if(codigo[m]=11) then

          (* m es punto extremo abierto aguas arriba con condicion
             de contorno caudal Q  *)

          begin
           dQ[m]:=dat[ubica[m]]-Q[m];
           dZ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1]
          end
                       else
       if(codigo[m]=12) then
```

```
              (* m es punto extremo abierto aguas arriba con condicion
                 de contorno nivel Z *)

              begin
               dZ[m]:=dat[ubica[m]]-Z[m];
               dQ[m]:=(c1[m,3]-c1[m,2]*dQ[m+1])/c1[m,1]
              end
     end;
     for m:=1 to npuntos do
       begin
         Q[m]:=Q[m]+dQ[m];
         Z[m]:=Z[m]+dZ[m]
       end
end;


  begin                                              (* barrido descendente *)
    for m:=1 to npuntos-1 do
      if((codigo[m+1]<>20) and (codigo[m]<>21) and (codigo[m+1]<>25)) then
           begin
             coefis(m,dts,teta,xcoor,Q,Z,S,D,B,derD,a);
             if(codigo[m+2]=20) then
    for l:=1 to 2 do a[l,5]:=a[l,5]+a[l,3]*(dat[ubica[m+2]]-(Q[m+2]-Q[m+1]));

             banda(a);
             if(codigo[m]=11)then
                      comienzo(0,dat[ubica[m]]-Q[m],a,c1[m,1],c1[m,2],
                                 c1[m,3],c2[m,1],c2[m,2],c2[m,3])
                               else
              if(codigo[m]=12) then
                       comienzo(1,dat[ubica[m]]-Z[m],a,c1[m,1],c1[m,2],
                                    c1[m,3],c2[m,1],c2[m,2],c2[m,3])
                                else
              begin
                        desciende(0,a,c2[m-1,1],c2[m-1,2],c2[m-1,3],
                                      c1[m,1],c1[m,2],c1[m,3]);
                        desciende(1,a,c1[m,1],c1[m,2],c1[m,3],c2[m,1],
                                      c2[m,2],c2[m,3])
               end
             end
                                     else
          begin
           if(codigo[m+1]=25) then
              begin
                c2[m,1]:=1;
                for l:=2 to 3 do
  c2[m,l]:=c2[m-1,l]/c2[m-1,1]+c2[ubica[m+1]-1,l]/c2[ubica[m+1]-1,1]
              end
                               else
           if(codigo[m+1]=20 ) then for l:= 1 to 3 do c2[m,l]:=c2[m-1,l]
          end;
        ascenso(codigo,ubica,npuntos,dat,c1,c2,Q,Z)
end;
```

```
begin                                                    (* cuencas *)
  write('Nombre del archivo de resultados ');
  readln(contor);
  assign(salida,contor);
  rewrite(salida);
  leemodelo(npuntos,ntab,nombre,tablaH,xcoor,cota,codigo,ubica,tablaB,
            tablaD) ;
  leeinic(npuntos,Q,Z);
  leeborde(npuntos,nombre,xcoor,cota,ubica,codigo,ndat,impri,tinic,dt,
             tfin,unidad);
  imp:=0;
  write('Archivo de condiciones de contorno ');
  readln(contor);
  assign(contorno,contor);
  reset(contorno);
  writeln(salida, 'teta = ',teta:10:3);
  precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,tablaD,B,derD,S,D);
  imprime(unidad,tinic,npuntos,nombre,Z,Q,S);
  leedat(ndat,dat1,t1);
  leedat(ndat,dat2,t2);
  if (unidad=' minutos') then dts:=dt*60.0
                         else if(unidad=' horas') then dts:= dt*3600.0
                         else dts:= dt*86400.0;
   t:= tinic+dt;
   imp:=1;
   while ( t <= tfin+0.0001 ) do
                  begin
                    interdat(t,ndat,dat,dat1,dat2,t1,t2);
                    barrido(npuntos,teta,dts,S,B,D,derD,codigo,
                            ubica,dat,Q,Z);
                    precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,
                            tablaD,B,derD,S,D) ;
                    if(imp=impri) then
                                    begin
                                      imprime(unidad,t,npuntos,nombre,
                                              Z,Q,S);
                                      imp:=0
                                    end;
                    t:=t+dt;
                    imp:=imp+1
                  end;
  close(salida);
  close(contorno)
end.
```

```
program deltsimp;

(* este programa resuelve el modelo de red fluvial con estructura
   deltaica mediante un metodo directo de solucion del sistema Ax=b,
   donde A es una matriz de coeficientes constantes *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.5;       (* parametro de "implicitud" *)
        imprix=10;        (* maximo numero de intervalos de impresion *)
        ntramx=20;        (* maximo numero de tramos *)
        npmtx=200;        (* maximo orden de la submatriz superior izquierda *)

        jmx=20;           (* maximo numero de confluencias *)
        ntr=20;           (* maximo numero de elementos no nulos de cada column

                             del rectangulo superior derecho *)

  type intnpmx=1..npmx;
       intjmx=1..jmx;
       intnpmtx=1..npmtx;
       intntr=1..ntr;
       intndmx=1..ndmx;
       intimpr=0..imprix;
       intntram=1..ntramx;
       arraycf=array[intntram] of 0..jmx;
       arraytr2=array[intntram] of intnpmtx;
       arrayx=array[intnpmx] of intntram;
       arraytr1=array[intntram]  of intnpmx;
       array25=array[1..2,1..5] of real;
       arrayd=array[intndmx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;
       arraypx3=array[intnpmtx,1..3] of real;
       arraysol=array[intnpmtx] of real;
       arrayntr=array[intntram] of 0..ntr;
       arrayntrj=array[intntram,intjmx] of real;
       arraycua=array[intjmx,intjmx] of real;
       arrayrhs=array[intjmx] of real;

   var contorno,salida:text;
       m,npuntos: intnpmx;  imp,impri:intimpr;
       ndat:intndmx; t,tfin,dt,dts,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,cota,Q,Z: arraypx;
       nombre: arrayp;
       codigo,ubica: arraypxi;
       contor: string[12] ;
       np1,np2:arraytr2;comtramo,fintramo:arraytr1;
       li1,li2:arrayntr;j1,j2:arraycf;imax:intnpmtx;jmax:intjmx;
       asig:arrayx;kmax:intntram;cr,Y:arrayrhs;c:arraypx3;
       cd:arraycua;X:arraysol;cz:arrayntrj;
```

```
function min(i1,i2:integer):integer;
begin
 if(i1<i2) then min:=i1 else min:=i2
end;

procedure leemodelo(var npuntos:intnpmx;var nombre:arrayp;
                    var xcoor:arraypx;
                    var codigo,ubica:arraypxi);


(* lee los datos fisicos,geometricos y topograficos de la red fluvial *)


  var red:text;modelo:string[12];titulo:string[80];m:intnpmx;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    readln(red,titulo);
    write(salida,titulo);
    readln(red,npuntos);
    for m:=1 to npuntos do
      begin
        readln(red,nombre[m],xcoor[m],codigo[m],ubica[m]);
        writeln(salida,'Seccion ',m,' ',nombre[m],
                  '  Coordenada (km)  ',xcoor[m]:8:2);
        writeln(salida,
                  '     Codigo ',codigo[m]:3,' relacionado con ',ubica[m]:3);
     end;
   close(red)
  end;

procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales en los puntos de discretizacion *)

  var iniciales: text;m:intnpmx;
      inicial:string[12];

  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:= 1 to npuntos do read(iniciales,Q[m]);
    for m:= 1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;
```

```
procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real);

(* este procedimiento indica cuales seran las condiciones de contorno *)

  var borde:integer; tiem: char; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= vel. dada');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/vel.');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Numero de condiciones de contorno ');
       readln(ndat);
       writeln(salida,'       Resumen del modelo');
       writeln(salida,
          ' Seccion  Coordenada (km)   Codigo   ',
            ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
         writeln(salida, nombre[m]:8,xcoor[m]:12:3,
                         codigo[m]:12,ubica[m]:11);
       write('Tiempo inicial de calculo ');
       readln(tinic);
       write('Intervalo dt de calculo ');
       readln(dt);
       write('Tiempo final de calculo ');
       readln(tfin);
       write('Cada cuantos intervalos se imprime? ');
       readln(impri);
       if(impri=0) then impri:=1;
       writeln(salida,'Intervalo dt de calculo ',dt:7:2);
       writeln(salida,'Tiempo final ',tfin:9:2 );
       writeln(salida,'Numero de condiciones de contorno ',ndat)
  end;
```

```
procedure precoefis(npuntos:intnpmx;Z:arraypx);

  var m:intnpmx;

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
        if(Z[m]<0) then fracaso(nombre[m]);
   end;

procedure imprime(t:real;npuntos:intnpmx;nombre:arrayp;
                  Z,Q:arraypx);

            (* imprime resultados en determinados instantes *)

  var m,m1,m2,nn: intnpmx;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:7);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Z         ');
        for m:= m1 to m2 do write(salida,Z[m]:7:3);
        writeln(salida);
        write(salida,'V         ');
        for m:= m1 to m2 do write(salida,Q[m]:7:3);
        writeln(salida)
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

     (* lee datos de contorno en instantes t *)

  var nd:intndmx;
  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;
```

```
procedure interdat(t:real;ndat:intndmx;var dat,dat1,dat2:arrayd;
            var t1,t2:real);

    (* interpola linealmente datos de contorno en instantes t en funcion
       de los datos de que dispone en instantes t1, t2, con t1 < t <= t2 *)

  var nd:intndmx;tet1,tet2:real;

  begin
      while(t > t2+0.0001)do
           begin
             t1:=t2;
             dat1:=dat2;
             leedat(ndat,dat2,t2)
           end;
        tet1:= (t2-t)/(t2-t1);
        tet2:=1-tet1;
        for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;

procedure armado(codigo,ubica:arraypxi;npuntos:intnpmx;var imax:intnpmtx;
                 var jmax:intjmx;var kmax:intntram;var asig:arrayx;
                 var comtramo,fintramo:arraytr1;var li1,li2:arrayntr;
                 var np1,np2:arraytr2;var j1,j2:arraycf);

     (* arma la estructura general de la matriz *)

var i:-1..npmtx;j:0..jmx;k:0..ntramx;

begin
 k:=0;
 i:=-1;
 j:=0;
 for m:=1 to npuntos do
   if((codigo[m]=11) or (codigo[m]=12) or (codigo[m]=25) or
        (codigo[m]=51) or (codigo[m]=55)) then
      begin
       k:=k+1;
       i:=i+2;
       comtramo[k]:=m;          (* punto de comienzo del tramo k *)
       np1[k]:=i;               (* primer fila del tramo k *)
       asig[m]:=k;              (* tramo al que pertenece el punto k *)
       if((codigo[m]=11)or (codigo[m]=12)) then

                                (* m es punto extremo abierto aguas arriba *)

                                          begin
                                            j1[k]:=0;
                                            li1[k]:=0
                                          end
                                        else
```

```
  if((codigo[m]=51) and (ubica[m]>m)) then

                          (* m es punto extremo cerrado aguas arriba
                             y pertenece al nodo de efluencia
                             {ubica[m]-1,m,ubica[m]} *)

                                      begin
                                          j:=j+1;
                                          j1[k]:=j;
                                          li1[k]:=1

     (* j indica la j-sima confluencia;
        j1[k] indica que el extremo aguas arriba del tramo k
           es la j-sima confluencia;
        li1[k] es la primer fila correspondiente al tramo k *)

                                       end
                                    else
  if((codigo[m]=51) and (ubica[m]<m)) then
                       begin
                        j1[k]:=j1[asig[ubica[m]]];
                        li1[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]]

                       end
                                    else
  if(codigo[m]=25)  then

           (* m es un punto extremo aguas arriba de tramo perteneciente
              al nodo de afluencia {m-1,ubica[m],m} *)

                       begin
                          j1[k]:=j2[k-1];
                          if (ubica[m]<m) then li1[k]:=3
                                          else li1[k]:=2
                       end
                                    else
  if(codigo[m]=55) then

             (* m es un punto extremo aguas arriba de tramo pertenecien


               al nodo de efluencia {m-1,ubica[m],m} *)

                       begin
                          j1[k]:=j2[k-1];
                          if(ubica[m]<m)then
                         li1[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]

                                      else li1[k]:=2
                       end
end
         else
 if((codigo[m]=0) and (codigo[m+1]<>20) and (codigo[m+1]<>25) and
          (codigo[m+1]<>55))    then

             (* m es un punto interior de tramo normal
                (no antecede a un punto extremo aguas abajo)  *)

      begin
        i:=i+2;
        asig[m]:=k
      end
```

```
if (codigo[m+1]=20) then asig[m]:=k
        else
if((codigo[m+1]=25) or (codigo[m+1]=55) or (codigo[m]>90)
    or (codigo[m]=21)) then
      begin
        np2[k]:=i+1;
        fintramo[k]:=m;
        asig[m]:=k;
        if(codigo[m]>90) then

                (* m es punto extremo abierto aguas abajo  *)

                          begin
                            j2[k]:=0;
                            li2[k]:=0
                          end
                        else
        if((codigo[m]=21) and (ubica[m]<m)) then

                 (* m es punto extremo cerrado aguas abajo
                    perteneciente al nodo de confluencia
                    {ubica[m]-1,m,ubica[m]} *)

                   begin
                     j2[k]:=j2[asig[ubica[m]-1]];
                     li2[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]]]
                   end
                          else
        if((codigo[m]=21) and (ubica[m]>m)) then
            begin
              j:=j+1;
              j2[k]:=j;
              li2[k]:=1
            end
                          else
        if((codigo[m+1]=55) and (ubica[m+1] <m)) then

             (* m es punto extremo cerrado aguas abajo
                perteneciente al nodo de efluencia {m,ubica[m],m+1} *)

                   begin
                    j2[k]:=j1[asig[ubica[m+1]]];
          li2[k]:=2+np2[asig[ubica[m+1]]]-np1[asig[ubica[m+1]]]
                   end
                          else
        if((codigo[m+1]=25) and (ubica[m+1]<m)) then

               (* m es punto extremo cerrado aguas abajo perteneciente
                  al nodo de afluencia {m,ubica[m],m+1} *)

                    begin
                      j2[k]:=j2[asig[ubica[m+1]]];
                      li2[k]:=2
                    end
                         else
```

```
                if(((codigo[m+1]=25) or (codigo[m+1]=55))
                  and(ubica[m+1]>m)) then
                   begin
                    j:=j+1;
                    j2[k]:=j;
                    li2[k]:=1
                   end
         end;
        kmax:=k;     (* numero de tramos *)
        jmax:=j;     (* numero de confluencias *)
        imax:=i+1    (* numero de ecuaciones de Saint-Venant discretizadas
                        entre puntos consecutivos *)
end;


procedure matriz(npuntos:intnpmx;imax:intnpmtx;jmax:intjmx;kmax:intntram;
                 teta,dts:real;
                 codigo,ubica:arraypxi;asig:arrayx;
                 comtramo,fintramo:arraytr1;li1,li2:arrayntr;
                 np1,np2:arraytr2;j1,j2:arraycf;
                 dat:arrayd;var c:arraypx3; var cz:arrayntrj;
                 var cr,Y:arrayrhs;var cd:arraycua;
                 var Q,Z:arraypx);

    (* resuelve el sistema Ax=b en cada intervalo de tiempo de calculo *)

  var k:intntram;a:array25;
      l,ll:intntr;m1,m2,m:intnpmx;j,jpost:0..jmx;i:-1..npmtx;cq,coef:real;
      r:0..1;

procedure rectder(cf,cz,c1,c2,c3:real;var cz1,cq,cd,cr:real);

      (* completa la triangulacion del tramo fila a fila si el extremo
         aguas arriba pertenece a un nodo de confluencia *)

var coef:real;

begin
  cz1:=-cz*cf;
  coef:=cq/c1;
  cd:=cd-cz*coef;
  cr:=cr-c3*coef;
  cq:=-coef*c2
end;
```

```
procedure coefis(m:intnpmx;dts,teta:real ;
                 xcoor,Q,Z:arraypx; var a:array25);

     (* calcula los elementos no nulos de A y b en dos filas consecutivas *

  var  r,dosrt:real;

  begin
    r:=dts/abs(xcoor[m+1]-xcoor[m]);
    dosrt:=2*r*teta;
    a[1,1]:=1;
    a[1,2]:=-dosrt;
    a[1,3]:=1;
    a[1,4]:=dosrt;
    a[1,5]:=2*r*(Z[m]-Z[m+1]);
    a[2,1]:=-dosrt;
    a[2,2]:=1;
    a[2,3]:=dosrt;
    a[2,4]:=1;
    a[2,5]:=2*r*(Q[m]-Q[m+1]);
  end;


procedure desciende(i:integer;a:array25;X1,X2,X3:real;var Y1,Y2,Y3,coef:rea




         (* triangula las ecuaciones del tramo *)

begin
  coef:=a[1+i,1+i]/X1;
  Y1:=a[1+i,2+i]-X2*coef;
  Y2:=a[1+i,3+i];
  Y3:=a[1+i,5]-X3*coef
end;

procedure comienzo(i:integer;dd:real;a:array25;var X1,X2,X3,Y1,Y2,Y3:real);

         (* triangula las dos primeras filas correspondientes a un tramo *)

var coef:real;

begin
  X1:=a[1,1+i];
  X2:=a[1,3];
  X3:=a[1,5]-dd*a[1,2-i];
  if(i=1) then desciende(1,a,X1,X2,X3,Y1,Y2,Y3,coef)
          else
            begin
              Y1:=a[2,3];
              Y2:=a[2,4];
              Y3:=a[2,5]-dd*a[2,2]
            end
end;
```

```
procedure  banda(var a:array25);

       (* tridiagonaliza las filas correspondientes a un tramo *)

var l:integer;coef:real;

begin
  coef:=a[2,1]/a[1,1];
  for l:= 2 to 5 do   a[2,l]:=a[2,l]-coef*a[1,l];
  coef:=a[1,4]/a[2,4];
  for l:=2 to 3 do a[1,l]:=a[1,l]-coef*a[2,l];
  a[1,5]:=a[1,5]-coef*a[2,5]
end;

procedure rhs(var a:array25;dd:real);

   (* modifica las ecuaciones para considerar inyeccion lateral de caudal

var i:1..2;

begin
  for i:=1 to 2 do a[i,5]:=a[i,5]+dd*a[i,3]
end;


begin                                (* matriz *)
  i:=-1;
  if (jmax>0) then                 (* inicializacion en cero *)
    begin
       for l:=1 to ntr do for j:=1 to jmax do cz[l,j]:=0;
       for j:=1 to jmax do
         begin
           cr[j]:=0;
           for jpost:=1 to jmax do cd[j,jpost]:=0
         end
    end;
  for k:=1 to kmax do
    begin
      m1:=comtramo[k];
      m2:=fintramo[k];
      j:=j1[k];
      i:=i+2;
      coefis(m1,dts,teta,xcoor,Q,Z,a);
      banda(a);
      if(j=0) then    (*   m1 es un extremo abierto   *)
          begin
       if(codigo[m1]=11) then comienzo(1,dat[ubica[m1]]-Q[m1],
                                a,c[i,1],c[i,2],c[i,3],
                                    c[i+1,1],c[i+1,2],c[i+1,3])

                     (* condicion de contorno aguas arriba caudal Q *)

                          else
       if(codigo[m1]=12) then comienzo(0,dat[ubica[m1]]-Z[m1],
                                a,c[i,1],c[i,2],c[i,3],
                                     c[i+1,1],c[i+1,2],c[i+1,3])

                      (* condicion de contorno aguas arriba nivel Z *)
          end
```

```
      else
              (* m1 pertenece a un nodo de confluencia *)
    begin
      comienzo(0,0,a,c[i,1],c[i,2],c[i,3],c[i+1,1],c[i+1,2],c[i+1,3])


            (* anula la fila con condicion de compatibilidad de
               Stoker en los caudales *)

      l:=li1[k];
      cz[l,j]:=a[1,2];
      cz[l+1,j]:=a[2,2];
      coef:=1/c[i,1];
      cq:=-c[i,2]*coef;
      cd[j,j]:=cd[j,j]-cz[l,j]*coef;
      cr[j]:=cr[j]-c[i,3]*coef
    end;
for m:=m1+1 to m2-1 do
 if(codigo[m+1]<>20) then
   begin
     i:=i+2;
     if(j>0) then l:=l+2;
     coefis(m,dts,teta,xcoor,Q,Z,a);
     if(codigo[m+2]=20) then rhs(a,dat[ubica[m+2]]-(Q[m+2]-Q[m+1]));
     banda(a);
     for r:=0 to 1 do
       begin
         desciende(r,a,c[i+r-1,1],c[i+r-1,2],c[i+r-1,3],
                       c[i+r,1],c[i+r,2],c[i+r,3],coef);
         if(j>0) then
                        rectder(coef,cz[l-1+r,j],c[i-1+r,1],c[i-1+r,2]

                           c[i-1+r,3],cz[l+r,j],cq,cd[j,j],cr[j])
       end
   end;
 if(codigo[m2]=91) then

      (* condicion de contorno aguas abajo caudal Q *)

   begin
      for r:=0 to 1 do
            c[i+r,3]:=c[i+r,3]-c[i+r,2-r]*(dat[ubica[m2]]-Q[m2]);
      c[i,2]:=0;
      c[i+1,1]:=c[i+1,2];
      c[i+1,2]:=0;
      if(j>0) then cr[j]:=cr[j]-cq*(dat[ubica[m2]]-Q[m2])
   end
                    else
 if(codigo[m2]=92) then
      (* condicion de contorno aguas abajo nivel Z *)

   begin
      c[i+1,3]:=c[i+1,3]-c[i+1,2]*(dat[ubica[m2]]-Z[m2]);
      c[i+1,2]:=0;
      if(j>0) then
        begin
         coef:=cq/c[i+1,1];
         cd[j,j]:=cd[j,j]-cz[l+1,j]*coef;
         cr[j]:=cr[j]-c[i+1,3]*coef
        end
    end
```

```
                                else

          (* el extremo aguas abajo m2 pertenece a un nodo de confluencia *)

            begin
              jpost:=j2[k];
              ll:=li2[k];
              cz[ll,jpost]:=c[i+1,2];
              c[i+1,2]:=0;
              coef:=1/c[i+1,1];
              cd[jpost,jpost]:=cd[jpost,jpost]+coef*cz[ll,jpost];
              cr[jpost]:=cr[jpost]+c[i+1,3]*coef;
              if(j>0)then
                begin
                  cq:=cq/c[i+1,1];
                  cd[jpost,j]:=cd[jpost,j]+coef*cz[l+1,j];
                  cd[j,jpost]:=cd[j,jpost]-cq*cz[ll,jpost];
                  cd[j,j]:=cd[j,j]-cq*cz[l+1,j];
                  cr[j]:=cr[j]-cq*c[i+1,3]
                end
            end
    end
end;


procedure asciende(kmax:intntram;jmax:intjmx;j1,j2:arraycf;
                   np1,np2:arraytr2;li1,li2:arrayntr;
                   c:arraypx3;cz:arrayntrj;cr:arrayrhs;
                   cd:arraycua;var X:arraysol;var Y:arrayrhs);

      (* barrido ascendente de la matriz *)

var k:intntram;j,jant:0..jmx;i:intnpmtx;l:0..ntr;

procedure gausspivmax(n:intjmx;a:arraycua;b:arrayrhs;var x:arrayrhs);

    (* solucion de un sistema lineal por el metodo de Gauss con
       pivote maximal por columna *)

var i,j,k,fili,filk:intjmx;piv:real;fil:array[intjmx] of intjmx;

begin
  for i:=1 to n do fil[i]:=i;
  for i:=1 to n-1 do
    begin
      piv:=a[fil[i],i];
      j:=i;
      for k:=i+1 to n do
        if(abs(piv)<abs(a[fil[k],i])) then
          begin
            piv:=a[fil[k],i];
            j:=k
          end;
        if(j<>i) then
          begin
            k:=fil[i];
            fil[i]:=fil[j];
            fil[j]:=k
          end;
```

```
        fili:=fil[i];
        for k:=i+1 to n do
          begin
            filk:=fil[k];
            if(abs(a[filk,i])>0.0000000001) then
              begin
                piv:=a[filk,i]/a[fili,i];
                for j:=i+1 to n do a[filk,j]:=a[filk,j]-piv*a[fili,j];
                b[filk]:=b[filk]-piv*b[fili]
              end
          end;
        x[n]:=b[fil[n]]/a[fil[n],n];
        for i:=n-1 downto 1 do
          begin
            fili:=fil[i];
            x[i]:=b[fili];
            for j:=i+1 to n do x[i]:=x[i]-a[fili,j]*x[j];
            x[i]:=x[i]/a[fili,i]
          end
      end;


begin
  if(jmax>0) then gausspivmax(jmax,cd,cr,Y);
  for k:=kmax downto 1 do
    begin
      j:=j2[k];
      i:=np2[k];
      jant:=j1[k];
      l:=li1[k];
      if((j=0) and (jant=0)) then X[i]:=c[i,3]/c[i,1]
                             else
      if((j>0) and (jant=0)) then
              X[i]:=(c[i,3]-Y[j]*cz[li2[k],j])/c[i,1]
                             else
      if((j=0)and(jant>0)) then
              X[i]:=(c[i,3]-Y[jant]*cz[l+np2[k]-np1[k],jant])/c[i,1]
                             else
      if((j>0) and (jant>0)) then
              X[i]:=(c[i,3]-Y[j]*cz[li2[k],j]
                          -Y[jant]*cz[l+np2[k]-np1[k],jant])/c[i,1];
      for i:=np2[k]-1 downto np1[k] do
       if(jant>0) then X[i]:=(c[i,3]-Y[jant]*cz[l+i-np1[k],jant]
                                     -X[i+1]*c[i,2])/c[i,1]
                  else X[i]:=(c[i,3]-c[i,2]*X[i+1])/c[i,1]
    end
end;
```

```
procedure asignar(kmax:intntram;comtramo,fintramo:arraytr1;
                  codigo,ubica:arraypxi;dat:arrayd;X:arraysol;
                  Y:arrayrhs;var Q,Z:arraypx);

         (* asigna valores de la solucion a los caudales Q y niveles Z
            en el nuevo intervalo temporal calculado *)

var k:intntram;j:0..jmx;i:1..npmtx;m,m1,m2:intnpmx;
begin
  i:=1;
  for k:=1 to kmax do
    begin
      j:=j1[k];
      m1:=comtramo[k];
      m2:=fintramo[k];
      if(j>0) then
        begin
          Z[m1]:=Y[j]+Z[m1];
          Q[m1]:=X[i]+Q[m1]
        end
             else
      if(codigo[m1]=11) then
        begin
          Z[m1]:=X[i]+Z[m1];
          Q[m1]:=dat[ubica[m1]]
        end
             else
      if(codigo[m1]=12) then
        begin
          Z[m1]:=dat[ubica[m1]];
          Q[m1]:=X[i]+Q[m1]
        end;
      i:=i+1;
      for m:=m1+1 to m2-1 do
        if(codigo[m+1]<>20) then
           begin
             Q[m]:=X[i]+Q[m];
             Z[m]:=X[i+1]+Z[m];
             i:=i+2
           end
                              else
           begin
             Q[m]:=X[i]+Q[m+1]-dat[ubica[m+1]]+Q[m];
             Z[m]:=X[i+1]+Z[m]
           end;
       j:=j2[k];
       if(j=0) then
         begin
           if(codigo[m2]=91) then
             begin
               Q[m2]:=dat[ubica[m2]];
               Z[m2]:=X[i]+Z[m2]
             end
           else
             begin
               Z[m2]:=dat[ubica[m2]];
               Q[m2]:=X[i]+Q[m2]
             end
         end
```

```
                else
          begin
            Z[m2]:=Y[j]+Z[m2];
            Q[m2]:=X[i]+Q[m2]
          end;
      i:=i+1
   end
 end;


 begin                                                    (* deltsimp *)
   write('Nombre del archivo de resultados ');
   readln(contor);
   assign(salida,contor);
   rewrite(salida);
   leemodelo(npuntos,nombre,xcoor,codigo,ubica);
   leeinic(npuntos,Q,Z);
   leeborde(npuntos,nombre,xcoor,ubica,codigo,ndat,impri,tinic,dt,
            tfin);
   imp:=0;
   write('Archivo de condiciones de contorno ');
   readln(contor);
   assign(contorno,contor);
   reset(contorno);
   precoefis(npuntos,Z);
   imprime(tinic,npuntos,nombre,Z,Q);
   leedat(ndat,dat1,t1);
   leedat(ndat,dat2,t2);
    t:= tinic+dt;
    imp:=1;
    armado(codigo,ubica,npuntos,imax,jmax,kmax,asig,comtramo,fintramo,
           li1,li2,np1,np2,j1,j2);
    while ( t <= tfin+0.0001 ) do
                  begin
                    interdat(t,ndat,dat,dat1,dat2,t1,t2);
                    matriz(npuntos,imax,jmax,kmax,teta,dt,
                    codigo,ubica,asig,comtramo,fintramo,li1,li2,
                    np1,np2,j1,j2,dat,c,cz,cr,Y,cd,Q,Z);

asciende(kmax,jmax,j1,j2,np1,np2,li1,li2,c,cz,cr,cd,X,Y);
                    asignar(kmax,comtramo,fintramo,codigo,ubica,
                                 dat,X,Y,Q,Z);
                    precoefis(npuntos,Z);
                    if(imp=impri) then
                                  begin
                                    imprime(t,npuntos,nombre,Z,Q);
                                    imp:=0
                                  end;
                    t:=t+dt;
                    imp:=imp+1
                  end;
   close(salida);
   close(contorno)
 end.
```

```
program delta;

(* este programa resuelve el modelo de red fluvial con estructura
   deltaica mediante un metodo directo de solucion del sistema Ax=b *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.85;        (* parametro de "implicitud" *)
        ntabx=2;          (* maximo numero de elementos de las tablas *)
        imprix=10;        (* maximo numero de intervalos de impresion *)
        ntramx=20;        (* maximo numero de tramos *)
        npmtx=200;        (* maximo orden de la submatriz superior izquierda *)

        jmx=20;           (* maximo numero de confluencias *)
        ntr=20;           (* maximo numero de elementos no nulos de cada columna
                             del rectangulo superior derecho *)

  type intnpmx=1..npmx;
       intjmx=1..jmx;
       intnpmtx=1..npmtx;
       intntr=1..ntr;
       intntabx=1..ntabx;
       intndmx=1..ndmx;
       intimpr=0..imprix;
       intntram=1..ntramx;
       arraycf=array[intntram] of 0..jmx;
       arraytr2=array[intntram] of intnpmtx;
       arrayx=array[intnpmx] of intntram;
       arraytr1=array[intntram]  of intnpmx;
       array25=array[1..2,1..5] of real;
       arrayd=array[intndmx] of real;
       arraynt=array[intnpmx,intntabx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       arrayt=array[intntabx] of real;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;
       arraypx3=array[intnpmtx,1..3] of real;
       arraysol=array[intnpmtx] of real;
       arrayntr=array[intntram] of 0..ntr;
       arrayntrj=array[intntram,intjmx] of real;
       arraycua=array[intjmx,intjmx] of real;
       arrayrhs=array[intjmx] of real;

   var contorno,salida:text;unidad:string7;
       m,npuntos: intnpmx;  imp,impri:intimpr; ntab:intntabx;
       ndat:intndmx; t,tfin,dt,dts,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,cota,Q,Z,S,D,B,derD: arraypx;
       nombre: arrayp;
       codigo,ubica: arraypxi;
       contor: string[12] ;tablaH:arrayt;
       tablaB,tablaD:arraynt;
       np1,np2:arraytr2;comtramo,fintramo:arraytr1;
       li1,li2:arrayntr;j1,j2:arraycf;imax:intnpmtx;jmax:intjmx;
       asig:arrayx;kmax:intntram;cr,Y:arrayrhs;c:arraypx3;
       cd:arraycua;X:arraysol;cz:arrayntrj;
```

```
function min(i1,i2:integer):integer;
begin
 if(i1<i2) then min:=i1 else min:=i2
end;

procedure leemodelo(var npuntos:intnpmx;var ntab:intntabx;var nombre:arrayp;
                    var tablaH:arrayt;var xcoor,cota:arraypx;
                    var codigo,ubica:arraypxi;var tablaB,tablaD:arraynt);

(* lee los datos fisicos,geometricos y topograficos de la red fluvial *)


  var red:text;modelo:string[12];titulo:string[80];
      k:intntabx;m:intnpmx;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    read(red,titulo);
    write(salida,titulo);
    read(red,ntab,npuntos);
    for k:=1 to ntab do read(red,tablaH[k]);readln(red);
    writeln(salida,'    Tabla de alturas (en m)');
    for k:= 1 to ntab do writeln(salida,k:2,tablaH[k]:12:2);
    for m:=1 to npuntos do
      begin
        read(red,nombre[m],xcoor[m],cota[m],codigo[m],ubica[m]);
        writeln(salida,'Seccion ',m,' ',nombre[m],
                  '  Coordenada (km)  ',xcoor[m]:8:2);
        writeln(salida,'      Cota de referencia (m) ',cota[m]:8:2,
                  '  Codigo ',codigo[m]:3,' relacionado con ',ubica[m]:3);
        writeln(salida,' Tabla de anchos (en m)  ',
                  '      Tabla de coeficientes de conduccion (en 1000m3/s)');
        xcoor[m]:=xcoor[m]*1000;
        for k:=1 to ntab do read(red,tablaB[m,k]);
        for k:=1 to ntab do
            begin
              read(red,tablaD[m,k]);
              tablaD[m,k]:=tablaD[m,k]*1000.;
              writeln(salida,'       ',tablaB[m,k]:10:2,'
                  tablaD[m,k]:12:2)
            end;
        readln(red)
     end;
   close(red)
  end;
```

```
procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales en los puntos de discretizacion *)

  var iniciales: text;m:intnpmx;
      inicial:string[12];

  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:= 1 to npuntos do read(iniciales,Q[m]);
    for m:= 1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;

procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor,cota:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real;var unidad:string7);

(* este procedimiento indica cuales seran las condiciones de contorno *)

  var borde:integer; tiem: char; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= caudal dado');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/caudal');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Unidad de tiempo (d,h,m) ');
       readln(tiem);
       write('Numero de condiciones de contorno ');
       readln(ndat);
       if(tiem='d') then unidad:=' dias'
            else if(tiem='h') then unidad:=' horas'
                 else unidad:=' minutos';
       writeln(salida,'       Resumen del modelo');
       writeln(salida,
         ' Seccion  Coordenada (km)    Cota de referencia (m)   Codigo
           ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
         writeln(salida, nombre[m]:8,xcoor[m]/1000.0:12:3,
                         cota[m]:27:2,codigo[m]:12,ubica[m]:11);
```

```
      write('Tiempo inicial de calculo ');
      readln(tinic);
      write('Intervalo dt de calculo ');
      readln(dt);
      write('Tiempo final de calculo ');
      readln(tfin);
      write('Cada cuantos intervalos se imprime? ');
      readln(impri);
      if(impri=0) then impri:=1;
      writeln(salida,'El tiempo se mide en',unidad,'.');
      writeln(salida,'Tiempo inicial ',tinic:9:2,unidad);
      writeln(salida,'Intervalo dt de calculo ',dt:7:2,unidad);
      writeln(salida,'Tiempo final ',tfin:9:2 ,unidad);
      writeln(salida,'Numero de condiciones de contorno ',ndat)
  end;

procedure precoefis(npuntos:intnpmx;ntab:intntabx;Z,cota:arraypx;
                    tablaH:arrayt;tablaB,tablaD:arraynt;
                    var B,derD,S,D:arraypx);

   (* calcula las areas, anchos, coeficientes de conduccion
      y derivadas de coeficientes de conduccion
      en cada instante t *)

  var h,dz:real;m:intnpmx;k,k1:intntabx;

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
      begin
        S[m]:=0;
        h:= Z[m]-cota[m];
        if(h<0) then fracaso(nombre[m]);
        k:= 1;
        repeat k:=k+1 until (h<tablaH[k]) or (k=ntab);
        dz:= h-tablaH[k-1];
        B[m]:= tablaB[m,k-1]+dz*(tablaB[m,k]-tablaB[m,k-1])/(tablaH[k]-
                                tablaH[k-1]);

                (* B[m] es el ancho mojado en el punto m *)

        derD[m]:= (tablaD[m,k]-tablaD[m,k-1])/(tablaH[k]-tablaH[k-1]);

                (* derD[m] es la derivada del coeficiente de conduccion
                       respecto de la altura *)

        if(k>2) then for k1:=2 to k-1 do
           S[m]:=S[m]+(tablaB[m,k1]+tablaB[m,k1-1])*(tablaH[k1]-tablaH[k

        S[m]:=0.5*((B[m]+tablaB[m,k-1])*dz+S[m]);

                 (* S[m] es el area mojada *)
```

```
        D[m]:=tablaD[m,k-1]+dz*derD[m]

                (* D[m] es el coeficiente de conduccion *)

      end
   end;

procedure imprime(unidad:string7;t:real;npuntos:intnpmx;nombre:arrayp;
                  Z,Q,S:arraypx);

           (* imprime resultados en determinados instantes *)

  var m,m1,m2,nn: intnpmx;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:2, unidad);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Cota  (m)');
        for m:= m1 to m2 do write(salida,Z[m]:7:2);
        writeln(salida);
        write(salida,'Q  (m3/s)');
        for m:= m1 to m2 do write(salida,Q[m]:7:0);
        writeln(salida);
        write(salida,'Vel.(m/s)');
        for m:= m1 to m2 do write(salida,Q[m]/S[m]:7:2)
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

           (* lee datos de contorno en instantes t *)

  var nd:intndmx;
  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;
```

```
procedure interdat(t:real;ndat:intndmx;
                   var dat,dat1,dat2:arrayd;var t1,t2:real);

      (* interpola linealmente datos de contorno en instantes t
         en funcion de los datos leidos en instantes t1 y t2
         con t1 < t <= t2 *)

  var nd:intndmx;tet1,tet2:real;

  begin
      while(t > t2+0.0001)do
           begin
             t1:=t2;
             dat1:=dat2;
             leedat(ndat,dat2,t2)
           end;
        tet1:= (t2-t)/(t2-t1);
        tet2:=1-tet1;
        for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;

procedure armado(codigo,ubica:arraypxi;npuntos:intnpmx;var imax:intnpmtx;
                 var jmax:intjmx;var kmax:intntram;var asig:arrayx;
                 var comtramo,fintramo:arraytr1;var li1,li2:arrayntr;
                 var np1,np2:arraytr2;var j1,j2:arraycf);

          (* arma la estructura general de la matriz *)

var i:-1..npmtx;j:0..jmx;k:0..ntramx;

begin
 k:=0;
 i:=-1;
 j:=0;
 for m:=1 to npuntos do
   if((codigo[m]=11) or (codigo[m]=12) or (codigo[m]=25) or
        (codigo[m]=51) or (codigo[m]=55)) then
      begin
       k:=k+1;
       i:=i+2;
       comtramo[k]:=m;          (* m es punto de comienzo del tramo k *)
       np1[k]:=i;               (* primera fila del tramo k *)
       asig[m]:=k;              (* tramo al que pertenece el punto m *)
       if((codigo[m]=11)or (codigo[m]=12)) then
                              (* m es punto extremo abierto aguas arriba *)
                                    begin
                                      j1[k]:=0;
                                      li1[k]:=0
                                    end
                                  else
```

```
     if((codigo[m]=51) and (ubica[m]>m)) then
                              (* m es punto extremo cerrado aguas arriba
                                 tal que pertenece al nodo de efluencia
                                 {ubica[m]-1,m,ubica[m]}  *)
                                          begin
                                              j:=j+1;
                                              j1[k]:=j;
                                              li1[k]:=1

(* j indica la j-sima columna de confluencias;
   j1[k] es la columna de confluencias correspondiente al
         extremo cerrado aguas arriba del tramo k;
   li1[k] indica que sobre la columna j hay li1[k]-1 elementos no nulos *)

                                           end
                                         else
     if((codigo[m]=51) and (ubica[m]<m)) then
                           begin
                            j1[k]:=j1[asig[ubica[m]]];
                            li1[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]]]

                           end
                                        else
     if(codigo[m]=25)  then

          (* m es punto extremo cerrado aguas arriba tal que el nodo de
            afluencia al que pertenece es {m-1,ubica[m],m} *)

                           begin
                              j1[k]:=j2[k-1];
                              if (ubica[m]<m) then li1[k]:=3
                                              else li1[k]:=2
                           end
                                       else
     if(codigo[m]=55) then

          (*  m es punto extremo cerrado aguas arriba tal que el nodo de
              efluencia al que pertenece es {m-1,ubica[m],m} *)

                           begin
                              j1[k]:=j2[k-1];
                              if(ubica[m]<m)then
                             li1[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]]]

                                       else li1[k]:=2
                           end
   end
            else
    if((codigo[m]=0) and (codigo[m+1]<>20) and (codigo[m+1]<>25) and
             (codigo[m+1]<>55)) then
          begin
            i:=i+2;
            asig[m]:=k
          end
             else
    if (codigo[m+1]=20) then asig[m]:=k
             else
```

```
      if((codigo[m+1]=25) or (codigo[m+1]=55) or (codigo[m]>90)
          or (codigo[m]=21)) then
            begin
              np2[k]:=i+1;           (* ultima fila del tramo k *)
              fintramo[k]:=m;        (* ultimo punto del tramo k *)
              asig[m]:=k;
              if(codigo[m]>90) then
                                begin
                                  j2[k]:=0;
                                  li2[k]:=0
                                end
                               else
              if((codigo[m]=21) and (ubica[m]<m)) then
                          begin
                            j2[k]:=j2[asig[ubica[m]-1]];
                            li2[k]:=3+np2[asig[ubica[m]]]-np1[asig[ubica[m]]]
                          end
                                else
              if((codigo[m]=21) and (ubica[m]>m)) then
                  begin
                    j:=j+1;
                    j2[k]:=j;
                    li2[k]:=1
                  end
                                else
              if((codigo[m+1]=55) and (ubica[m+1] <m)) then
                          begin
                           j2[k]:=j1[asig[ubica[m+1]]];
                li2[k]:=2+np2[asig[ubica[m+1]]]-np1[asig[ubica[m+1]]]
                          end
                                else
              if((codigo[m+1]=25) and (ubica[m+1]<m)) then
                           begin
                             j2[k]:=j2[asig[ubica[m+1]]];
                             li2[k]:=2
                           end
                                else
               if(((codigo[m+1]=25) or (codigo[m+1]=55))
                 and(ubica[m+1]>m)) then
                  begin
                   j:=j+1;
                   j2[k]:=j;
                   li2[k]:=1
                  end
        end;
       kmax:=k;                (* numero de tramos *)
       jmax:=j;                (* numero de confluencias *)
       imax:=i+1    (* numero de ecuaciones de Saint-Venant discretizadas
                          entre puntos contiguos *)
end;
```

```
procedure matriz(npuntos:intnpmx;imax:intnpmtx;jmax:intjmx;kmax:intntram;
                 teta,dts:real;S,B,D,derD:arraypx;
                 codigo,ubica:arraypxi;asig:arrayx;
                 comtramo,fintramo:arraytr1;li1,li2:arrayntr;
                 np1,np2:arraytr2;j1,j2:arraycf;
                 dat:arrayd;var c:arraypx3; var cz:arrayntrj;
                 var cr,Y:arrayrhs;var cd:arraycua;
                 var Q,Z:arraypx);

    (* resuelve el sistema Ax=b en cada intervalo de tiempo *)

  var k:intntram;a:array25;
      l,ll:intntr;m1,m2,m:intnpmx;j,jpost:0..jmx;i:-1..npmtx;cq,coef:real;
      r:0..1;

procedure rectder(cf,cz,c1,c2,c3:real;var cz1,cq,cd,cr:real);

          (* completa la triangulacion del tramo fila a fila si el
             extremo aguas arriba es un extremo cerrado *)

var coef:real;

begin
  cz1:=-cz*cf;
  coef:=cq/c1;
  cd:=cd-cz*coef;
  cr:=cr-c3*coef;
  cq:=-coef*c2
end;

procedure coefis(m:intnpmx;dts,teta:real ;
                 xcoor,Q,Z,S,D,B,derD:arraypx; var a:array25);

      (* calcula los coeficientes de las ecuaciones
         de Saint-Venant discretizadas para dos filas consecutivas *)

  var  dx,xc,x1,x2,x3,x4,efe:real;

  begin
    dx:=abs(xcoor[m+1]-xcoor[m]);
    xc:=dx/(19.62*dts);
    x1:=dx*abs(Q[m])/sqr(D[m]);
    x2:=dx*abs(Q[m+1])/sqr(D[m+1]);
    x3:=Q[m]/(9.81*sqr(S[m]));
    x4:=Q[m+1]/(9.81*sqr(S[m+1]));
    efe:=dx/(2*teta*dts);
    a[1,1]:=-1;
    a[1,2]:=efe*B[m];
    a[1,3]:=1;
    a[1,4]:=efe*B[m+1];
    a[1,5]:=(Q[m]-Q[m+1])/teta;
    a[2,1]:=teta*(x1-x3)+xc/S[m];
    a[2,2]:=-teta-teta*x1*Q[m]*derD[m]/D[m]
            +(teta*x3-xc/S[m])*Q[m]*B[m]/S[m];
    a[2,3]:=teta*(x2+x4)+xc/S[m+1];
    a[2,4]:=teta-teta*x2*Q[m+1]*derD[m+1]/D[m+1]-
             (teta*x4+xc/S[m+1])*Q[m+1]*B[m+1]/S[m+1];
    a[2,5]:=Z[m]-Z[m+1]-0.5*(x1-x3)*Q[m]-(x2+x4)*Q[m+1]
  end;
```

```
procedure desciende(i:integer;a:array25;X1,X2,X3:real;var Y1,Y2,Y3,coef:real);



    (* triangula las ecuaciones del tramo *)

begin
  coef:=a[1+i,1+i]/X1;
  Y1:=a[1+i,2+i]-X2*coef;
  Y2:=a[1+i,3+i];
  Y3:=a[1+i,5]-X3*coef
end;

procedure comienzo(i:integer;dd:real;a:array25;var X1,X2,X3,Y1,Y2,Y3:real);

     (* triangula las dos primeras filas correspondientes a un tramo *)

var coef:real;

begin
  X1:=a[1,1+i];
  X2:=a[1,3];
  X3:=a[1,5]-dd*a[1,2-i];
  if(i=1) then desciende(1,a,X1,X2,X3,Y1,Y2,Y3,coef)
          else
            begin
              Y1:=a[2,3];
              Y2:=a[2,4];
              Y3:=a[2,5]-dd*a[2,2]
            end
end;


procedure  banda(var a:array25);

        (* tridiagonaliza las filas de cada tramo *)

var l:integer;coef:real;

begin
  coef:=a[2,1]/a[1,1];
  for l:= 2 to 5 do   a[2,l]:=a[2,l]-coef*a[1,l];
  coef:=a[1,4]/a[2,4];
  for l:=2 to 3 do a[1,l]:=a[1,l]-coef*a[2,l];
  a[1,5]:=a[1,5]-coef*a[2,5]
end;

procedure rhs(var a:array25;dd:real);

      (* modifica las ecuaciones si hay inyeccion lateral de caudal *)

var i:1..2;

begin
  for i:=1 to 2 do a[i,5]:=a[i,5]+dd*a[i,3]
end;
```

```
begin                                   (* matriz *)
  i:=-1;
  if (jmax>0) then
    begin
       for l:=1 to ntr do for j:=1 to jmax do cz[l,j]:=0;
       for j:=1 to jmax do
         begin
           cr[j]:=0;
           for jpost:=1 to jmax do cd[j,jpost]:=0
         end
    end;
  for k:=1 to kmax do
    begin
      m1:=comtramo[k];
      m2:=fintramo[k];
      j:=j1[k];
      i:=i+2;
      coefis(m1,dts,teta,xcoor,Q,Z,S,D,B,derD,a);
      banda(a);
      if(j=0) then        (* m1 es extremo abierto *)
          begin
       if(codigo[m1]=11) then comienzo(1,dat[ubica[m1]]-Q[m1],
                                a,c[i,1],c[i,2],c[i,3],
                                    c[i+1,1],c[i+1,2],c[i+1,3])

                         (* condicion de contorno aguas arriba caudal Q *)

                          else
       if(codigo[m1]=12) then comienzo(0,dat[ubica[m1]]-Z[m1],
                                a,c[i,1],c[i,2],c[i,3],
                                     c[i+1,1],c[i+1,2],c[i+1,3])

                         (* condicion de contorno aguas arriba nivel Z *)

          end
             else
          begin           (* m1 pertenece a una confluencia *)
             comienzo(0,0,a,c[i,1],c[i,2],c[i,3],c[i+1,1],c[i+1,2],c[i+1,3]

             l:=li1[k];
             cz[l,j]:=a[1,2];
             cz[l+1,j]:=a[2,2];
             coef:=1/c[i,1];
             cq:=-c[i,2]*coef;
             cd[j,j]:=cd[j,j]-cz[l,j]*coef;
             cr[j]:=cr[j]-c[i,3]*coef

(* anula la fila con condicion de compatibilidad de Stoker en los caudales *



          end;
       for m:=m1+1 to m2-1 do
        if(codigo[m+1]<>20) then
          begin
            i:=i+2;
            if(j>0) then l:=l+2;
            coefis(m,dts,teta,xcoor,Q,Z,S,D,B,derD,a);
            if(codigo[m+2]=20) then rhs(a,dat[ubica[m+2]]-(Q[m+2]-Q[m+1]));

                  (* inyeccion lateral de caudal *)
```

```
          banda(a);
          for r:=0 to 1 do
            begin
              desciende(r,a,c[i+r-1,1],c[i+r-1,2],c[i+r-1,3],
                        c[i+r,1],c[i+r,2],c[i+r,3],coef);
              if(j>0) then
                        rectder(coef,cz[l-1+r,j],c[i-1+r,1],c[i-1+r,2]

                           c[i-1+r,3],cz[l+r,j],cq,cd[j,j],cr[j])
            end
        end;
      if(codigo[m2]=91) then

          (* condicion de contorno aguas abajo caudal Q *)

        begin
          for r:=0 to 1 do
              c[i+r,3]:=c[i+r,3]-c[i+r,2-r]*(dat[ubica[m2]]-Q[m2]);
          c[i,2]:=0;
          c[i+1,1]:=c[i+1,2];
          c[i+1,2]:=0;
          if(j>0) then cr[j]:=cr[j]-cq*(dat[ubica[m2]]-Q[m2])
        end
                    else
      if(codigo[m2]=92) then

          (* condicion de contorno aguas abajo nivel Z *)

        begin
          c[i+1,3]:=c[i+1,3]-c[i+1,2]*(dat[ubica[m2]]-Z[m2]);
          c[i+1,2]:=0;
          if(j>0) then
            begin
             coef:=cq/c[i+1,1];
             cd[j,j]:=cd[j,j]-cz[l+1,j]*coef;
             cr[j]:=cr[j]-c[i+1,3]*coef
            end
         end
                      else
         begin     (* extremo aguas abajo pertenece a nodo de confluencia*)

           jpost:=j2[k];
           ll:=li2[k];
           cz[ll,jpost]:=c[i+1,2];
           c[i+1,2]:=0;
           coef:=1/c[i+1,1];
           cd[jpost,jpost]:=cd[jpost,jpost]+coef*cz[ll,jpost];
           cr[jpost]:=cr[jpost]+c[i+1,3]*coef;
           if(j>0)then
             begin
               cq:=cq/c[i+1,1];
               cd[jpost,j]:=cd[jpost,j]+coef*cz[l+1,j];
               cd[j,jpost]:=cd[j,jpost]-cq*cz[ll,jpost];
               cd[j,j]:=cd[j,j]-cq*cz[l+1,j];
               cr[j]:=cr[j]-cq*c[i+1,3]
             end
         end
    end
end;
```

```
procedure asciende(kmax:intntram;jmax:intjmx;j1,j2:arraycf;
                   np1,np2:arraytr2;li1,li2:arrayntr;
                   c:arraypx3;cz:arrayntrj;cr:arrayrhs;
                   cd:arraycua;var X:arraysol;var Y:arrayrhs);

          (* barrido ascendente *)

var k:intntram;j,jant:0..jmx;i:intnpmtx;l:0..ntr;

procedure gausspivmax(n:intjmx;a:arraycua;b:arrayrhs;var x:arrayrhs);

        (* solucion de un sistema lineal por el metodo de Gauss con
           pivote maximal por columnas *)

var i,j,k,fili,filk:intjmx;piv:real;fil:array[intjmx] of intjmx;

begin
  for i:=1 to n do fil[i]:=i;
  for i:=1 to n-1 do
    begin
      piv:=a[fil[i],i];
      j:=i;
      for k:=i+1 to n do
        if(abs(piv)<abs(a[fil[k],i])) then
          begin
            piv:=a[fil[k],i];
            j:=k
          end;
        if(j<>i) then
          begin
            k:=fil[i];
            fil[i]:=fil[j];
            fil[j]:=k
          end;
        fili:=fil[i];
        for k:=i+1 to n do
          begin
            filk:=fil[k];
            if(abs(a[filk,i])>0.0000000001) then
              begin
                piv:=a[filk,i]/a[fili,i];
                for j:=i+1 to n do a[filk,j]:=a[filk,j]-piv*a[fili,j];
                b[filk]:=b[filk]-piv*b[fili]
              end
            end
          end;
        x[n]:=b[fil[n]]/a[fil[n],n];
        for i:=n-1 downto 1 do
          begin
            fili:=fil[i];
            x[i]:=b[fili];
            for j:=i+1 to n do x[i]:=x[i]-a[fili,j]*x[j];
            x[i]:=x[i]/a[fili,i]
          end
      end;
```

```
begin
  if(jmax>0) then gausspivmax(jmax,cd,cr,Y);
  for k:=kmax downto 1 do
    begin
      j:=j2[k];
      i:=np2[k];
      jant:=j1[k];
      l:=li1[k];
      if((j=0) and (jant=0)) then X[i]:=c[i,3]/c[i,1]
                             else
      if((j>0) and (jant=0)) then
              X[i]:=(c[i,3]-Y[j]*cz[li2[k],j])/c[i,1]
                             else
      if((j=0)and(jant>0)) then
              X[i]:=(c[i,3]-Y[jant]*cz[l+np2[k]-np1[k],jant])/c[i,1]
                             else
      if((j>0) and (jant>0)) then
              X[i]:=(c[i,3]-Y[j]*cz[li2[k],j]
                          -Y[jant]*cz[l+np2[k]-np1[k],jant])/c[i,1];
      for i:=np2[k]-1 downto np1[k] do
       if(jant>0) then X[i]:=(c[i,3]-Y[jant]*cz[l+i-np1[k],jant]
                                    -X[i+1]*c[i,2])/c[i,1]
                  else X[i]:=(c[i,3]-c[i,2]*X[i+1])/c[i,1]
    end
end;

procedure asignar(kmax:intntram;comtramo,fintramo:arraytr1;
                  codigo,ubica:arraypxi;dat:arrayd;X:arraysol;
                  Y:arrayrhs;var Q,Z:arraypx);

    (* asignacion de valores de la solucion a los caudales Q y a los nivele
        Z *)

var k:intntram;j:0..jmx;i:1..npmtx;m,m1,m2:intnpmx;

begin
  i:=1;
  for k:=1 to kmax do
    begin
      j:=j1[k];
      m1:=comtramo[k];
      m2:=fintramo[k];
      if(j>0) then
        begin
          Z[m1]:=Y[j]+Z[m1];
          Q[m1]:=X[i]+Q[m1]
        end
            else
      if(codigo[m1]=11) then
        begin
          Z[m1]:=X[i]+Z[m1];
          Q[m1]:=dat[ubica[m1]]
        end
            else
```

```
      if(codigo[m1]=12) then
        begin
          Z[m1]:=dat[ubica[m1]];
          Q[m1]:=X[i]+Q[m1]
        end;
      i:=i+1;
      for m:=m1+1 to m2-1 do
        if(codigo[m+1]<>20) then
           begin
             Q[m]:=X[i]+Q[m];
             Z[m]:=X[i+1]+Z[m];
             i:=i+2
           end
                              else
           begin
             Q[m]:=X[i]+Q[m+1]-dat[ubica[m+1]]+Q[m];
             Z[m]:=X[i+1]+Z[m]
           end;
       j:=j2[k];
       if(j=0) then
         begin
           if(codigo[m2]=91) then
             begin
               Q[m2]:=dat[ubica[m2]];
               Z[m2]:=X[i]+Z[m2]
             end
           else
             begin
               Z[m2]:=dat[ubica[m2]];
               Q[m2]:=X[i]+Q[m2]
             end
         end
                 else
          begin
            Z[m2]:=Y[j]+Z[m2];
            Q[m2]:=X[i]+Q[m2]
          end;
      i:=i+1
  end
end;


 begin                                                          (* delta *
   write('Nombre del archivo de resultados ');
   readln(contor);
   assign(salida,contor);
   rewrite(salida);
   leemodelo(npuntos,ntab,nombre,tablaH,xcoor,cota,codigo,ubica,tablaB,
             tablaD) ;
   leeinic(npuntos,Q,Z);
   leeborde(npuntos,nombre,xcoor,cota,ubica,codigo,ndat,impri,tinic,dt,
              tfin,unidad);
   imp:=0;
   write('Archivo de condiciones de contorno ');
   readln(contor);
   assign(contorno,contor);
   reset(contorno);
   precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,tablaD,B,derD,S,D);
   imprime(unidad,tinic,npuntos,nombre,Z,Q,S);
```

```
    leedat(ndat,dat1,t1);
    leedat(ndat,dat2,t2);
    if (unidad=' minutos') then dts:=dt*60.0
                           else if(unidad=' horas') then dts:= dt*3600.0
                           else dts:= dt*86400.0;
     t:= tinic+dt;
     imp:=1;
     armado(codigo,ubica,npuntos,imax,jmax,kmax,asig,comtramo,fintramo,
            li1,li2,np1,np2,j1,j2);
     while ( t <= tfin+0.0001 ) do
                    begin
                      interdat(t,ndat,dat,dat1,dat2,t1,t2);
                      matriz(npuntos,imax,jmax,kmax,teta,dts,S,B,D,derD,
                      codigo,ubica,asig,comtramo,fintramo,li1,li2,
                      np1,np2,j1,j2,dat,c,cz,cr,Y,cd,Q,Z);

asciende(kmax,jmax,j1,j2,np1,np2,li1,li2,c,cz,cr,cd,X,Y);
                      asignar(kmax,comtramo,fintramo,codigo,ubica,
                                   dat,X,Y,Q,Z);
                      precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,
                              tablaD,B,derD,S,D) ;
                      if(imp=impri) then
                                     begin
                                       imprime(unidad,t,npuntos,nombre,
                                               Z,Q,S);
                                       imp:=0
                                     end;
                      t:=t+dt;
                      imp:=imp+1
                    end;
    close(salida);
    close(contorno)
  end.
```

```
program delta1;

(* este programa resuelve el modelo de red fluvial con estructura deltaica
   por el metodo iterativo de Kacmarcz o de proyecciones para la resolucion
   del sistema lineal Ax=b   *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.90;       (* parametro de "implicitud" *)
        npmx2=200;        (* maximo orden posible de la matriz *)
        npmx5=1000;       (* maximo numero de elementos no nulos de la matriz*)

        ntabx=2;          (* maximo numero de elementos de las tablas *)
        imprix=1000;        (* maximo numero de intervalos entre impresiones *)


  type intnpmx=1..npmx;
       intntabx=1..ntabx;
       intndmx=1..ndmx;
       int0ndmx=0..ndmx;
       intimpr=0..imprix;
       intnpmx2=1..npmx2;
       intnpmx5=1..npmx5;
       array15=array[1..5] of real;
       array15i=array[1..npmx5] of integer;
       array15r=array[1..npmx5] of real;
       array12i=array[1..npmx2] of integer;
       array12r=array[1..npmx2] of real;
       arrayd=array[intndmx] of real;
       arraynt=array[intnpmx,intntabx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       arrayt=array[intntabx] of real;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;

   var contorno,salida:text;unidad:string7;
       m,npuntos: intnpmx;  imp,impri:intimpr; ntab:intntabx;
       ndat:intndmx; converg,t,tfin,dt,dts,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,cota,Q,Z,S,D,B,derD: arraypx;
       nombre: arrayp;
       codigo,ubica,np1,np2,np3: arraypxi;
       contor: string[12] ;tablaH:arrayt;
       tablaB,tablaD:arraynt;
       v: array12i; w:array15i;
       jmax:intnpmx5;imax:intnpmx2;

function min(a1,a2: integer): integer;
  begin if (a1<a2) then min:=a1 else min:=a2 end;
```

```
procedure leemodelo(var npuntos:intnpmx;var ntab:intntabx;var nombre:arra
                    var tablaH:arrayt;var xcoor,cota:arraypx;
                    var codigo,ubica:arraypxi;var tablaB,tablaD:arraynt);

(* lee los datos fisicos,geometricos y topograficos de la red fluvial *)

  var red:text;modelo:string[12];titulo:string[80];
      k,m:integer;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    read(red,titulo);
    write(salida,titulo);
    read(red,ntab,npuntos);
    for k:=1 to ntab do read(red,tablaH[k]);readln(red);
    writeln(salida,'    Tabla de alturas (en m)');
    for k:= 1 to ntab do writeln(salida,k:2,tablaH[k]:12:2);
    for m:=1 to npuntos do
      begin
        read(red,nombre[m],xcoor[m],cota[m],codigo[m],ubica[m]);
        writeln(salida,'Seccion ',m,' ',nombre[m],
                '  Coordenada (km)  ',xcoor[m]:8:2);
        writeln(salida,'      Cota de referencia (m) ',cota[m]:8:2,
                '  Codigo ',codigo[m]:3,' relacionado con ',ubica[m]:3);
        writeln(salida,' Tabla de anchos (en m)  ',
                '     Tabla de coeficientes de conduccion (en 1000m3/s)'
        xcoor[m]:=xcoor[m]*1000;
        for k:=1 to ntab do read(red,tablaB[m,k]);
        for k:=1 to ntab do
            begin
              read(red,tablaD[m,k]);
              tablaD[m,k]:=tablaD[m,k]*1000.;
              writeln(salida,'      ',tablaB[m,k]:10:2,'
                  tablaD[m,k]:12:2)
            end;
        readln(red)
     end;
   close(red)
  end;

procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales de cada punto de discretizacion *)

  var iniciales: text;m:intnpmx;
      inicial:string[12];

  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:= 1 to npuntos do read(iniciales,Q[m]);
    for m:= 1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;
```

```
procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor,cota:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real;var unidad:string7);

(* este procedimiento indica cuales seran las condiciones de contorno *)

  var borde:integer; tiem: char; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= caudal dado');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/caudal');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Unidad de tiempo (d,h,m) ');
       readln(tiem);
       write('Numero de condiciones de contorno ');
       readln(ndat);
       if(tiem='d') then unidad:=' dias'
            else if(tiem='h') then unidad:=' horas'
                 else unidad:=' minutos';
       if(impri=0) then impri:=1;
       writeln(salida,'       Resumen del modelo');
       writeln(salida,
         ' Seccion  Coordenada (km)    Cota de referencia (m)   Codigo
           ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
         writeln(salida, nombre[m]:8,xcoor[m]/1000.0:12:3,
                         cota[m]:27:2,codigo[m]:12,ubica[m]:11);
       write('Tiempo inicial de calculo ');
       readln(tinic);
       write('Intervalo dt de calculo ');
       readln(dt);
       write('Tiempo final de calculo ');
       readln(tfin);
       write('Cada cuantos intervalos se imprime? ');
       readln(impri);
       writeln(salida,'El tiempo se mide en',unidad,'.');
       writeln(salida,'Tiempo inicial ',tinic:9:2,unidad);
       writeln(salida,'Intervalo dt de calculo ',dt:7:2,unidad);
       writeln(salida,'Tiempo final ',tfin:9:2 ,unidad);
       writeln(salida,'Numero de condiciones de contorno ',ndat);
       writeln(salida,'Tolerancia admitida ',converg:10:5);
       writeln(salida,'teta = ',teta:6:3)
  end;
```

```
procedure precoefis(npuntos:intnpmx;ntab:intntabx;Z,cota:arraypx;
                    tablaH:arrayt;tablaB,tablaD:arraynt;
                    var B,derD,S,D:arraypx);

   (* calcula areas, anchos, coeficientes de conduccion y
      derivadas de coeficientes de conduccion *)

  var h:real;m,k,k1:integer;dz:real;

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
      begin
        h:= Z[m]-cota[m];
        if(h<0) then fracaso(nombre[m]);
        k:= 1;
        S[m]:=0;
        repeat k:=k+1 until (h<tablaH[k]) or (k=ntab);
        dz:= h-tablaH[k-1];
        B[m]:= tablaB[m,k-1]+dz*(tablaB[m,k]-tablaB[m,k-1])/(tablaH[k]-
                                 tablaH[k-1]);

                    (* B es el ancho mojado *)

        derD[m]:= (tablaD[m,k]-tablaD[m,k-1])/(tablaH[k]-tablaH[k-1]);

           (* derD es la derivada del coeficiente de conduccion respecto
              del nivel Z *)

        if(k>2) then for k1:=2 to k-1 do
          S[m]:=S[m]+(tablaB[m,k1]+tablaB[m,k1-1])*
                        (tablaH[k1]-tablaH[k1-1]);
        S[m]:=0.5*((B[m]+tablaB[m,k-1])*dz+S[m]);

                (* S es el area mojada *)

        D[m]:=tablaD[m,k-1]+dz*derD[m]

                 (* D es el coeficiente de conduccion *)
      end
   end;
```

```
procedure imprime(unidad:string7;t:real;npuntos:intnpmx;nombre:arrayp;
                  Z,Q,S:arraypx);

     (* imprime resultados en determinados instantes t *)

  var m,m1,m2,nn: integer;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:2, unidad);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Cota  (m)');
        for m:= m1 to m2 do write(salida,Z[m]:7:2);
        writeln(salida);
        write(salida,'Q  (m3/s)');
        for m:= m1 to m2 do write(salida,Q[m]:7:0);
        writeln(salida);
        write(salida,'Vel.(m/s)');
        for m:= m1 to m2 do write(salida,Q[m]/S[m]:7:2)
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

   (* lee datos de contorno en determinados instantes t *)

  var nd:intndmx;
  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;

procedure interdat(t:real;ndat:intndmx;var dat,dat1,dat2:arrayd;var
t1,t2:real);

    (* interpola linealmente datos de contorno en el instante t
       cuando dispone de datos en los instantes t1 y t2 con t1<t<=t2 *)

  var nd:intndmx;tet1,tet2:real;

  begin
       while(t > t2+0.0001) do
          begin
            t1:=t2;
            dat1:=dat2;
            leedat(ndat,dat2,t2)
          end;
       tet1:= (t2-t)/(t2-t1);
       tet2:=1-tet1;
       for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;
```

```
procedure armado(npuntos:intnpmx;codigo,ubica:arraypxi;var np1,np2,np3:
                 arraypxi;var v:array12i;var w:array15i;var jmax:intnpmx5;
                 var imax:intnpmx2);

 (* arma la estructura de la matriz por medio de vectores apuntadores *)

  var i,m,k,j:integer;

     (* v[i] indica el numero de elementos no nulos de la fila i;
        w[j] indica la posicion del j-simo elemento no nulo de
             la matriz, recorriendo fila a fila, en la fila
             correspondiente, que se sabe cual es gracias a v[i];
        z[j] indica el valor numerico de w[j]    *)

procedure arme(i1,i2,m,i: integer;var j:integer;var np2:arraypxi;
               var v:array12i;var w:array15i);

(* arma los vectores v y w para los puntos que no anteceden a uno
   perteneciente a un nodo de afluencia o efluencia *)

  var k:integer;
  begin
    np2[m+1]:= np2[m]+i1;
    v[i]:=i2;
    v[i+1]:=i2;
    for k:= 1 to i2 do
      begin
        w[j+k]:=np2[m]+k-1;
        w[j+k+i2]:= w[j+k]
      end;
    j:=j+2*i2
  end;

procedure arme1(i1,i2,i3,i4,i5,i6,i7,m,i:integer;ubica,np3:arraypxi;
                var j:integer;
                var np2:arraypxi;var v:array12i;var w:array15i);

    (* arma los vectores v y w para puntos pertenecientes a
              nodos de confluencia *)

var i8,k:integer;

begin
  i8:=min(1,i4);
  np2[m+1]:=np2[m]+i1;
  v[i]:=i2;
  v[i+1]:=i2;
  if(ubica[m+i3]<m) then
       begin
         for k:=2 to i2 do
           begin
             w[j+k]:=np2[m]+k-2;
             w[j+k+i2]:=w[j+k]
           end;
         w[j+1]:=np2[ubica[m+i3]-i8]+i4;
         w[j+1+i2]:=w[j+1];
         w[np3[ubica[m+i3]-i8]+i5]:=w[j+i7];
         w[np3[ubica[m+i3]-i8]+i6]:=w[j+i7]
       end
```

```
                    else
       for k:=1 to i2-1 do
           begin
             w[j+k]:=np2[m]+k-1;
             w[j+k+i2]:=w[k+j]
           end;
  j:=j+i2*2
end;

    begin                                                    (* armado *)
      j:=0;
      np1[1]:=1;     (* np1 indica la primera fila del par de filas
                     correspondiente al punto m *)
      np2[1]:=1;     (* np2 indica la columna en que comienzan los valores
                      no nulos del par de filas correspondientes al
                      punto m, exceptuando los valores aislados *)

      for m:= 1 to npuntos - 1 do
       if((codigo[m]=20) or (codigo[m]=21) or (codigo[m]=55) or (codigo[m]=25)

           or (codigo[m]>90)) then
              begin
                np1[m+1]:=np1[m];  (* esos puntos no tienen ecuaciones *)
                np2[m+1]:=np2[m];  (* la columna de comienzo no se corre *)
                np3[m]:=0
              end
                          else
              begin
                np1[m+1]:= np1[m]+2;
                i:=np1[m];
                np3[m]:=j+1;      (* np3 indica el lugar del vector w en que
                                     comienzan los valores correspondientes
                                     a la fila i *)
                if ((codigo[m]=11) or (codigo[m]=12))then
                          arme (1,3,m,i,j,np2,v,w)
                                     else
                if((codigo[m]=0)  and (codigo[m+1]=0) or (codigo[m+1]=20))
then
                          arme (2,4,m,i,j,np2,v,w)
                                     else
                if(codigo[m+1]>=90) then
                          arme (3,3,m,i,j,np2,v,w)
                                     else
                if(codigo[m+1]=21)then
                          arme1(3,4,1,1,4,9,4,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m+1]=25)then
                          arme1(2,5,1,2,3,7,3,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m]=51) then
                          arme1(1,4,0,1,4,9,2,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m+1]=55)then
                          arme1(2,5,1,0,3,7,3,m,i,ubica,np3,j,np2,v,w)
              end;
      jmax:=j;
      imax:=np1[npuntos]-1;
      np3[npuntos]:=0
    end;
```

```
procedure
 matriz(npuntos:intnpmx;imax:intnpmx2;teta,dts:real;S,B,D,derD:arraypx;
                codigo,ubica:arraypxi;dat:arrayd;v:array12i;w:array15i;
                var Q,Z:arraypx);

  var a1,a2:array15;
      be,x:array12r;
      ze:array15r;i:intnpmx2;j:intnpmx5;ik,i1:integer;

procedure coefis(m:intnpmx;dts,teta:real ;
                xcoor,Q,Z,S,B,D,derD:arraypx; var a1,a2:array15);

        (* calcula los elementos no nulos de la matriz A *)

  var      dx,xc,x1,x2,x3,x4,efe:real;      n:integer;

  begin
    if((codigo[m+1]=55) or (codigo[m+1]=25) or (codigo[m+1]=20))
          then n:=m+1 else n:=m;
    dx:=abs(xcoor[n+1]-xcoor[n]);
    xc:=dx/(19.62*dts);
    x1:=dx*abs(Q[n])/sqr(D[n]);
    x2:=dx*abs(Q[n+1])/sqr(D[n+1]);
    x3:=Q[n]/(9.81*sqr(S[n]));
    x4:=Q[n+1]/(9.81*sqr(S[n+1]));
    a1[1]:=teta*(x1-x3)+xc/S[n];
    a1[2]:=-teta-teta*x1*Q[n]*derD[n]/D[n]
          +(teta*x3-xc/S[n])*Q[n]*B[n]/S[n];
    a1[3]:=teta*(x2+x4)+xc/S[n+1];
    a1[4]:=teta-teta*x2*Q[n+1]*derD[n+1]/D[n+1]-
           (teta*x4+xc/S[n+1])*Q[n+1]*B[n+1]/S[n+1];
    a1[5]:=Z[n]-Z[n+1]-0.5*(x1-x3)*Q[n]-(x2+x4)*Q[n+1];
    efe:=dx/(2*teta*dts);
    a2[1]:=-1;
    a2[2]:=efe*B[n];
    a2[3]:=1;
    a2[4]:=efe*B[n+1];
    a2[5]:=(Q[n]-Q[n+1])/teta
  end;


procedure compact1(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r;
                   var be:array12r);

    (* asigna valores numericos a los elementos de la matriz para un
       punto m extremo abierto aguas arriba con caudal Q dado, o extremo
       abierto aguas abajo con nivel Z dado, o extremo cerrado aguas
       abajo perteneciente a un nodo de afluencia de la forma
            {ubica[m]-1,m,ubica[m]}  *)

  var k:1..3;

  begin
    for k:=1 to 3 do
      begin
        ze[j+k+i1]:=a1[k+i3];
        ze[j+k+3+i1+i2]:=a2[k+i3]
      end;
```

```
    if(i2=0) then
      begin
        be[i]:=a1[5]-dd*a1[4-3*i3];
        be[i+1]:=a2[5]-dd*a2[4-3*i3]
      end
            else
      begin
        be[i]:=a1[5];
        be[i+1]:=a2[5];
        ze[j+4-3*i1]:=a1[4];
        ze[j+8-3*i1]:=a2[4]
      end;
    j:=j+6+2*i2
  end;

procedure compact2(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r; var be:array12r);

        (* asigna valores numericos a los elementos no nulos de la matriz
           para m punto extremo aguas arriba con nivel Z dado, o
           extremo abierto aguas abajo con caudal Q dado, o
           extremo cerrado aguas arriba perteneciente a nodo de efluencia
           de la forma {ubica[m]-1,m,ubica[m]} *)

  var k:integer;

  begin
    ze[j+1+i1+2*i3]:=a1[1+3*i3];
    ze[j+4+i1+i2+2*i3]:=a2[1+3*i3];
    for k:=1 to 2 do
      begin
        ze[j+k+i1+1-i3]:=a1[k+2-2*i3];
        ze[j+k+i1+4+i2-i3]:=a2[k+2-2*i3]
      end;
    if(i2=1) then
      begin
        be[i]:=a1[5];
        ze[j+4-3*i1]:=a1[2];
        be[i+1]:=a2[5];
        ze[j+8-3*i1]:= a2[2]
      end
            else
      begin
        be[i]:=a1[5]-dd*a1[2+i3];
        be[i+1]:=a2[5]-dd*a2[2+i3]
      end;
    j:=j+6+2*i2
  end;
```

```
procedure compact3(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r; var be:array12r);

     (* asigna valores numericos a los elementos no nulos de la matriz
        para punto m interior a tramo simple, o punto extremo aguas
        arriba de afluente, o punto extremo aguas arriba de efluente
        de la forma {m-1,ubica[m], m } *)

  var k:integer;

  begin
    for k:=1 to 4 do
      begin
        ze[j+k+i1]:=a1[k];
        ze[j+k+4+i1+i2]:=a2[k]
      end;
    if((i2=0) and (i3=0)) then
      begin
        be[i]:=a1[5]-dd*a1[1];
        be[i+1]:=a2[5]-dd*a2[1]
      end
                      else
      begin
        be[i]:=a1[5];
        be[i+1]:=a2[5];
        if(i2=1) then
          begin
            ze[j+5-4*i1]:=a1[1]*i3;
            ze[j+10-4*i1]:=a2[1]*i3
          end
      end;
    j:=j+8+2*i2
  end;

procedure kacz(imax:intnpmx2;v:array12i; w:array15i;be:array12r;ze:array15
               var x:array12r);

     (* metodo de las proyecciones *)

  const iter=200000.0;
  var i,ik,i1,j,l: integer;k,epsi,prod,as,eps,coef:real;xant,asq:array12r;
  procedure noconverge;

  begin
    writeln(salida);
    writeln(salida,'El procedimiento de Kacmarz no converge ');
    close(salida);
    close(contorno);
    halt
  end;
```

```
begin                                                          (* kacz *)
  k:=0;
  j:=0;
  ik:=0;
  for i:=1 to imax do
    begin
      x[i]:=0;
      as:=0;
      for i1:=1 to v[i] do as:=as+sqr(ze[j+i1]);
      asq[i]:=as;
      xant[i]:=0;
      j:=j+v[i]
    end;
  eps:=converg+1;
  j:=0;
  repeat
    ik:=ik+1;
    if(ik>imax) then begin j:=0 ;ik:=1 end;
    k:=k+1;
    prod:=0;
    for i1:=1 to v[ik] do
       begin
            l:=w[j+i1];
            prod:=prod+ze[j+i1]*x[l]
       end;
     coef:=(be[ik]-prod)/asq[ik];
     for i1 :=1 to v[ik] do
       begin
            l:=w[j+i1];
            x[l]:=x[l]+0.5*coef*ze[j+i1];
       end;
     if(ik=imax) then
       begin
         eps:=0;
         for i:=1 to imax do
           begin
             epsi:=abs(xant[i]-x[i]);
             xant[i]:=x[i];
             if(epsi>eps) then eps:=epsi
           end (*;writeln(eps,k:6)   *)
        end;
      j:=j+v[ik]
  until ((k>=iter) or (eps<converg));writeln(k:15:0,eps:15:5);
  if((k>=iter) and (eps >=converg)) then noconverge
end;
```

```
procedure recupera(npuntos:intnpmx;np2,codigo,ubica:arraypxi;x:array12r;
                   dat:arrayd; var Q,Z:arraypx);

       (* asigna resultados a las variables Z y Q actualizadas *)

  var m:intnpmx;j:intnpmx2;
  begin
    for m:=1 to npuntos do
      begin
        j:=np2[m];
        if(codigo[m]=11) then
          begin
            Z[m]:=Z[m]+x[j];
            Q[m]:=dat[ubica[m]]
          end
                         else
        if(codigo[m]=12) then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=
            dat[ubica[m]]
          end
                         else
        if(codigo[m]=0)  then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=Z[m]+x[j+1]
          end
                          else
        if(codigo[m]=20) then
          begin
            Q[m]:=Q[m-1]+dat[ubica[m]];
            Z[m]:=Z[m-1]
          end
                          else
        if(codigo[m]=21) then
          begin
            Q[m]:=Q[m]+x[j-1];
            Z[m]:=Z[m]+x[np2[ubica[m]-1]+1]
          end
                          else
        if(codigo[m]=51) then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=Z[m]+x[np2[ubica[m]-1]+1]
          end
                          else
        if(codigo[m]=91) then
          begin
            Q[m]:=dat[ubica[m]];
            Z[m]:=Z[m]+x[j-1]
          end
                          else
        if(codigo[m]=92) then
          begin
            Z[m]:=dat[ubica[m]];
            Q[m]:=Q[m]+x[j-1]
          end
      end;
```

```
    for m:=1 to npuntos do
        if(codigo[m]=25) then
          begin
            Z[m]:=Z[m-1];
            Q[m]:=Q[m-1]+Q[ubica[m]]
          end
                        else
        if(codigo[m]=55) then
          begin
            Z[m]:=Z[m-1];
            Q[m]:=Q[m-1]-Q[ubica[m]]
          end
  end;

begin                                                (* matriz *)
  j:=0;
  i:=1;
  for m:=1 to npuntos-1 do
    if((codigo[m]<>20) and (codigo[m]<>21) and (codigo[m]<>25)
       and (codigo[m]<>55) and (codigo[m]<90)) then
         begin
           if(codigo[m]=51) then ik:=0 else ik:=1;
           if(ubica[m+ik]<m) then i1:=1 else i1:=0;
           coefis(m,dts,teta,xcoor,Q,Z,S,B,D,derD,a1,a2);
           if(codigo[m]=11) then compact1(0,0,1,i,a1,a2,
              dat[ubica[m]]-Q[m],j,ze,be)
                            else
           if(codigo[m+1]=92) then compact1(0,0,0,i,a1,a2,
              dat[ubica[m+1]]-Z[m+1],j,ze,be)
                            else
           if(codigo[m+1]=21) then compact1(i1,1,0,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m]=12) then compact2(0,0,0,i,a1,a2,
               dat[ubica[m]]-Z[m] ,j,ze,be)
                            else
           if(codigo[m+1]=91) then compact2(0,0,1,i,a1,a2,
               dat[ubica[m+1]]-Q[m+1],j,ze,be)
                            else
           if(codigo[m]=51) then compact2(i1,1,0,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m+1]=20) then compact3(0,0,0,i,a1,a2,
                dat[ubica[m+1]]-(Q[m+1]-Q[m]),j,ze,be)
                            else
           if(codigo[m+1]=25) then compact3(i1,1,1,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m+1]=55) then compact3(i1,1,-1,i,a1,a2,0.,j,ze,be
                            else
                                 compact3(0,0,1,i,a1,a2,0.,j,ze,be);
          i:=i+2
         end;
  kacz(imax,v,w,be,ze,x);
  recupera(npuntos,np2,codigo,ubica,x,dat,Q,Z)
end;
```

```
begin                                                          (* delta1 *)
  write('Nombre del archivo de resultados ');
  readln(contor);
  assign(salida,contor);
  rewrite(salida);
  write('Tolerancia admitida ');
  readln(converg);
  leemodelo(npuntos,ntab,nombre,tablaH,xcoor,cota,codigo,ubica,tablaB,
            tablaD) ;
  leeinic(npuntos,Q,Z);
  leeborde(npuntos,nombre,xcoor,cota,ubica,codigo,ndat,impri,tinic,dt,
             tfin,unidad);
  imp:=0;
  write('Archivo de condiciones de contorno ');
  readln(contor);
  assign(contorno,contor);
  reset(contorno);
  precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,tablaD,B,derD,S,D);
  imprime(unidad,tinic,npuntos,nombre,Z,Q,S);
  armado(npuntos,codigo,ubica,np1,np2,np3,v,w,jmax,imax);
  leedat(ndat,dat1,t1);
  leedat(ndat,dat2,t2);
  if (unidad=' minutos') then dts:=dt*60.0
                         else if(unidad=' horas') then dts:= dt*3600.0
                         else dts:= dt*86400.0;
   t:= tinic+dt;
   imp:=1;
   while ( t <= tfin ) do
                  begin
                    interdat(t,ndat,dat,dat1,dat2,t1,t2);
                    matriz(npuntos,imax,teta,dts,S,B,D,derD,codigo,
                           ubica,dat,v,w,Q,Z);
                    precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,
                           tablaD,B,derD,S,D) ;
                    if(imp=impri) then
                                    begin
                                      imprime(unidad,t,npuntos,nombre,
                                              Z,Q,S);
                                      imp:=0
                                    end;
                    t:=t+dt;
                    imp:=imp+1
                  end;
  close(salida);
  close(contorno)
end.
```

```
program delta2;

(* este programa resuelve el modelo de red fluvial con estructura deltaica
   por el metodo de Kacmarz con proyecciones sobre intersecciones de
   hiperplanos para la resolucion del sistema lineal
   Ax=b *)

  const npmx=48;          (* maximo numero de puntos de discretizacion *)
        ndmx=6;           (* maximo numero de condiciones de contorno *)
        teta=0.90;       (* parametro de "implicitud" *)
        npmx2=200;        (* maximo orden posible de la matriz *)
        npmx5=1000;       (* maximo numero de elementos no nulos de la matriz*

        ntabx=2;          (* maximo numero de elementos de las tablas *)
        imprix=500;        (* maximo numero de intervalos entre impresiones *)
        nmedio=100;

  type intnpmx=1..npmx;
       intmedio=1..nmedio;
       intntabx=1..ntabx;
       intndmx=1..ndmx;
       int0ndmx=0..ndmx;
       intimpr=0..imprix;
       intnpmx2=1..npmx2;
       intnpmx5=1..npmx5;
       arraymedio=array[intmedio] of real;
       arraylog=array[intmedio] of boolean;
       array15=array[1..5] of real;
       array15i=array[1..npmx5] of integer;
       array15r=array[1..npmx5] of real;
       array12i=array[1..npmx2] of integer;
       array12r=array[1..npmx2] of real;
       arrayd=array[intndmx] of real;
       arraynt=array[intnpmx,intntabx] of real;
       arraypx=array[intnpmx] of real;
       arraypxi=array[intnpmx] of integer;
       arrayt=array[intntabx] of real;
       string6=string[6];string7=string[7];
       arrayp=array[intnpmx] of string6;

    var contorno,salida:text;unidad:string7;
       m,npuntos: intnpmx;  imp,impri:intimpr; ntab:intntabx;
       ndat:intndmx;converg,t,tfin,dt,dts,tinic,t1,t2:real;
       dat1,dat2,dat:arrayd;
       xcoor,cota,Q,Z,S,D,B,derD: arraypx;
       nombre: arrayp;
       codigo,ubica,np1,np2,np3: arraypxi;
       contor: string[12] ;tablaH:arrayt;
       tablaB,tablaD:arraynt;
       v: array12i; w:array15i;
       jmax:intnpmx5;imax:intnpmx2;

function min(a1,a2: integer): integer;
  begin if (a1<a2) then min:=a1 else min:=a2 end;
```

```
procedure leemodelo(var npuntos:intnpmx;var ntab:intntabx;var nombre:arrayp
                    var tablaH:arrayt;var xcoor,cota:arraypx;
                    var codigo,ubica:arraypxi;var tablaB,tablaD:arraynt);

(* lee los datos fisicos,geometricos y topograficos de la red fluvial *)


  var red:text;modelo:string[12];titulo:string[80];
      k,m:integer;

  begin
    write('Nombre del archivo de la red fluvial ');
    readln(modelo);
    assign(red,modelo);
    reset(red) ;
    read(red,titulo);
    write(salida,titulo);
    read(red,ntab,npuntos);
    for k:=1 to ntab do read(red,tablaH[k]);readln(red);
    writeln(salida,'    Tabla de alturas (en m)');
    for k:= 1 to ntab do writeln(salida,k:2,tablaH[k]:12:2);
    for m:=1 to npuntos do
      begin
        read(red,nombre[m],xcoor[m],cota[m],codigo[m],ubica[m]);
        writeln(salida,'Seccion ',m,' ',nombre[m],
                  ' Coordenada (km)  ',xcoor[m]:8:2);
        writeln(salida,'      Cota de referencia (m) ',cota[m]:8:2,
                  ' Codigo ',codigo[m]:3,' relacionado con ',ubica[m]:3);
        writeln(salida,' Tabla de anchos (en m)  ',
                  '     Tabla de coeficientes de conduccion (en 1000m3/s)');
        xcoor[m]:=xcoor[m]*1000;
        for k:=1 to ntab do  read(red,tablaB[m,k]);
        for k:=1 to ntab do
            begin
              read(red,tablaD[m,k]);
              tablaD[m,k]:=tablaD[m,k]*1000.;
              writeln(salida,'       ',tablaB[m,k]:10:2,'          ',
                             tablaD[m,k]/1000:12:3)
            end;
       readln(red)
     end;
   close(red)
  end;

procedure leeinic(npuntos:intnpmx;var Q,Z:arraypx);

(* lee condiciones iniciales para todos los puntos de discretizacion *)

  var iniciales:text;m:intnpmx;
      inicial:string[12];
  begin
    write('Nombre del archivo de condiciones iniciales ');
    readln(inicial);
    assign(iniciales,inicial);
    reset(iniciales);
    for m:=1 to npuntos do read(iniciales,Q[m]);
    for m:=1 to npuntos do read(iniciales,Z[m]);
    close(iniciales)
  end;
```

```
procedure leeborde(npuntos:intnpmx;nombre:arrayp;xcoor,cota:arraypx;
                   var  ubica, codigo:arraypxi;var ndat:intndmx;
                   var impri:intimpr;
                   var tinic,dt,tfin:real;var unidad:string7);

(* este procedimiento indica cuales seran las condiciones de contorno *)

  var borde:integer; tiem: char; m:intnpmx;

  begin
      for m:=1 to npuntos do
        if (codigo[m]=10) or (codigo[m]=90) then
             begin
               writeln('Codigo de la seccion ',m);
               writeln('Codigo 1= caudal dado');
               writeln('Codigo 2= cota dada');
               writeln('Codigo 3= ley cota/caudal');
               readln(borde);
               while (borde<1) or (borde >3) do
                 begin
                    write('Dato mal ingresado. Ingresarlo de nuevo');
                    readln(borde)
                 end;
               codigo[m]:=codigo[m]+borde;
               write('Ubicacion del dato en el vector de datos ');
               readln(ubica[m])
             end;
       write('Unidad de tiempo (d,h,m) ');
       readln(tiem);
       write('Numero de condiciones de contorno ');
       readln(ndat);
       if(tiem='d') then unidad:=' dias'
            else if(tiem='h') then unidad:=' horas'
                 else unidad:=' minutos';
       if(impri=0) then impri:=1;
       writeln(salida,'       Resumen del modelo');
       writeln(salida,
         ' Seccion  Coordenada (km)    Cota de referencia (m)   Codigo
           ' Apuntador');
       writeln(salida);
       for m:=1 to npuntos do
         writeln(salida, nombre[m]:8,xcoor[m]/1000.0:12:3,
                         cota[m]:27:2,codigo[m]:12,ubica[m]:11);
       writeln(salida,'Tolerancia = ',converg:10:5);
       write('Tiempo inicial de calculo ');
       readln(tinic);
       write('Intervalo dt de calculo ');
       readln(dt);
       write('Tiempo final de calculo ');
       readln(tfin);
       write('Cada cuantos intervalos se imprime? ');
       readln(impri);
       writeln(salida,'El tiempo se mide en',unidad,'.');
       writeln(salida,'Tiempo inicial ',tinic:9:2,unidad);
       writeln(salida,'Intervalo dt de calculo ',dt:7:2,unidad);
       writeln(salida,'Tiempo final ',tfin:9:2 ,unidad);
       writeln(salida,'Numero de condiciones de contorno ',ndat)
  end;
```

```
procedure precoefis(npuntos:intnpmx;ntab:intntabx;Z,cota:arraypx;
                    tablaH:arrayt;tablaB,tablaD:arraynt;
                    var B,derD,S,D:arraypx);

    (* calcula areas, anchos, coeficientes de conduccion
       y derivadas de coeficientes de conduccion *)

  var h:real;m,k,k1:integer;dz:real;

  procedure fracaso(mm:string6);
  begin
    writeln(salida);
    writeln(salida,' altura negativa en seccion ',mm);
    close(salida);
    close(contorno);
    halt
  end;

  begin
    for m:= 1 to npuntos do
      begin
        h:= Z[m]-cota[m];
        if(h<0) then fracaso(nombre[m]);
        k:= 1;
        S[m]:=0;
        repeat k:=k+1 until (h<tablaH[k]) or (k=ntab);
        dz:= h-tablaH[k-1];
        B[m]:= tablaB[m,k-1]+dz*(tablaB[m,k]-tablaB[m,k-1])/(tablaH[k]-
                                tablaH[k-1]);

                                (* B es el ancho mojado *)

        derD[m]:= (tablaD[m,k]-tablaD[m,k-1])/(tablaH[k]-tablaH[k-1]);

        (* derD es la derivada del coeficiente de conduccion
           respecto de Z *)

        if(k>2) then for k1:=2 to k-1 do
         S[m]:=S[m]+(tablaB[m,k1]+tablaB[m,k1-1])*(tablaH[k1]-tablaH[k1-1]
        S[m]:=0.5*((B[m]+tablaB[m,k-1])*dz+S[m]);

                (* S es el area mojada *)

        D[m]:=tablaD[m,k-1]+dz*derD[m]

                (* D es el coeficiente de conduccion *)

      end
   end;
```

```
procedure imprime(unidad:string7;t:real;npuntos:intnpmx;nombre:arrayp;
                  Z,Q,S:arraypx);

     (*  imprime resultados en determinados instantes *)

  var m,m1,m2,nn: integer;
  begin
    writeln(salida);writeln(salida);
    writeln(salida,'Tiempo ',t:12:2, unidad);
    for nn:=1 to (npuntos-1) div 10 +1 do
      begin
        m1:=(nn-1)*10+1;
        m2:=min(nn*10,npuntos);
        writeln(salida);
        write(salida,'          ');
        for m:= m1 to m2 do write(salida,nombre[m]:7);
        writeln(salida);
        write(salida,'Cota  (m)');
        for m:= m1 to m2 do write(salida,Z[m]:7:2);
        writeln(salida);
        write(salida,'Q  (m3/s)');
        for m:= m1 to m2 do write(salida,Q[m]:7:0);
        writeln(salida);
        write(salida,'Vel.(m/s)');
        for m:= m1 to m2 do write(salida,Q[m]/S[m]:7:2)
      end
  end;

procedure leedat(ndat:intndmx;var dat:arrayd ;var t:real);

    (* lee condiciones de contorno en determinados instantes t *)

var nd:intndmx;

  begin
    read(contorno,t);
    for nd:= 1 to ndat do read(contorno,dat[nd])
  end;

procedure interdat(t:real;ndat:intndmx;var dat,dat1,dat2:arrayd;var
t1,t2:real);

    (* interpola linealmente condiciones de contorno en un instante t
       en funcion  de los datos de que dispone en instantes t1 y t2,
       con t1<t<=t2  *)

  var nd:intndmx;tet1,tet2:real;

  begin
      while(t > t2+0.0001) do
           begin
             t1:=t2;
             dat1:=dat2;
             leedat(ndat,dat2,t2)
           end;
        tet1:= (t2-t)/(t2-t1);
        tet2:=1-tet1;
        for nd:=1 to ndat do dat[nd]:=tet1*dat1[nd]+tet2*dat2[nd]
  end;
```

```
procedure armado(npuntos:intnpmx;codigo,ubica:arraypxi;var np1,np2,np3:
                 arraypxi;var v:array12i;var w:array15i;var jmax:intnpmx5;
                 var imax:intnpmx2);

(* arma la estructura de la matriz por medio de vectores apuntadores *)


  var i,m,k,j:integer;

  (* v[i] indica el numero de elementos no nulos de la fila i;
     w[j] indica la ubicacion del j-simo valor no nulo de la matriz,
          recorriendo fila a fila, en la fila a la cual pertenece,
          que sabe cual es gracias a v[i];
     z[j] es el valor numerico de w[j] *)

procedure arme(i1,i2,m,i: integer;var j:integer;var np2:arraypxi;
               var v:array12i;var w:array15i);

(* arma los vectores v y w para los puntos que no anteceden a uno
   perteneciente a un nodo de afluencia o efluencia *)

  var k:integer;

  begin
    np2[m+1]:= np2[m]+i1;
    v[i]:=i2;
    v[i+1]:=i2;
    for k:= 1 to i2 do
      begin
        w[j+k]:=np2[m]+k-1;
        w[j+k+i2]:= w[j+k]
      end;
    j:=j+2*i2
  end;

procedure arme1(i1,i2,i3,i4,i5,i6,i7,m,i:integer;ubica,np3:arraypxi;
                var j:integer;
                var np2:arraypxi;var v:array12i;var w:array15i);

(* arma los vectores v y w para puntos pertenecientes a
    nnodos de confluencia *)

var i8,k:integer;

begin
  i8:=min(1,i4);
  np2[m+1]:=np2[m]+i1;
  v[i]:=i2;
  v[i+1]:=i2;
  if(ubica[m+i3]<m) then
       begin
         for k:=2 to i2 do
           begin
             w[j+k]:=np2[m]+k-2;
             w[j+k+i2]:=w[j+k]
           end;
```

```
          w[j+1]:=np2[ubica[m+i3]-i8]+i4;
          w[j+1+i2]:=w[j+1];
          w[np3[ubica[m+i3]-i8]+i5]:=w[j+i7];
          w[np3[ubica[m+i3]-i8]+i6]:=w[j+i7]
        end
                        else
        for k:=1 to i2-1 do
            begin
              w[j+k]:=np2[m]+k-1;
              w[j+k+i2]:=w[k+j]
            end;
  j:=j+i2*2
end;

    begin                                                    (* armado *)
      j:=0;
      np1[1]:=1;     (* np1 indica la primera fila del par de filas
                     correspondiente al punto m *)
      np2[1]:=1;     (* np2 indica la columna en que comienzan los valores
                      no nulos del par de filas correspondientes al
                      punto m, exceptuando los valores aislados *)

      for m:= 1 to npuntos - 1 do
       if((codigo[m]=20) or (codigo[m]=21) or (codigo[m]=55) or (codigo[m]=2

           or (codigo[m]>90)) then
              begin
                np1[m+1]:=np1[m];  (* esos puntos no tienen ecuaciones *)
                np2[m+1]:=np2[m];  (* la columna de comienzo no se corre *)
                np3[m]:=0
              end
                           else
              begin
                np1[m+1]:= np1[m]+2;
                i:=np1[m];
                np3[m]:=j+1;      (* np3 indica el lugar del vector w en que
                                     comienzan los valores correspondientes
                                     a la fila i *)
                if ((codigo[m]=11) or (codigo[m]=12))then
                          arme (1,3,m,i,j,np2,v,w)
                                     else
                if((codigo[m]=0)  and (codigo[m+1]=0) or (codigo[m+1]=20))
then
                          arme (2,4,m,i,j,np2,v,w)
                                     else
                if(codigo[m+1]>=90) then
                          arme (3,3,m,i,j,np2,v,w)
                                     else
                if(codigo[m+1]=21)then
                          arme1(3,4,1,1,4,9,4,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m+1]=25)then
                          arme1(2,5,1,2,3,7,3,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m]=51) then
                          arme1(1,4,0,1,4,9,2,m,i,ubica,np3,j,np2,v,w)
                                     else
                if(codigo[m+1]=55)then
                          arme1(2,5,1,0,3,7,3,m,i,ubica,np3,j,np2,v,w)
              end;
```

```
      jmax:=j;
      imax:=np1[npuntos]-1;
      np3[npuntos]:=0
    end;

procedure
 matriz(npuntos:intnpmx;imax:intnpmx2;teta,dts:real;S,B,D,derD:arraypx;
                codigo,ubica:arraypxi;dat:arrayd;v:array12i;w:array15i;
                var Q,Z:arraypx);

  var a1,a2:array15;
      be,x:array12r;
      ze:array15r;i:intnpmx2;j:intnpmx5;ik,i1:integer;

procedure coefis(m:intnpmx;dts,teta:real ;
                xcoor,Q,Z,S,B,D,derD:arraypx; var a1,a2:array15);

    (* calcula los valores de los elementos no nulos de la matriz *)

  var      dx,xc,x1,x2,x3,x4,efe:real;      n:integer;

  begin
    if((codigo[m+1]=55) or (codigo[m+1]=25) or (codigo[m+1]=20))
           then n:=m+1 else n:=m;
    dx:=abs(xcoor[n+1]-xcoor[n]);
    xc:=dx/(19.62*dts);
    x1:=dx*abs(Q[n])/sqr(D[n]);
    x2:=dx*abs(Q[n+1])/sqr(D[n+1]);
    x3:=Q[n]/(9.81*sqr(S[n]));
    x4:=Q[n+1]/(9.81*sqr(S[n+1]));
    a1[1]:=teta*(x1-x3)+xc/S[n];
    a1[2]:=-teta-teta*x1*Q[n]*derD[n]/D[n]
           +(teta*x3-xc/S[n])*Q[n]*B[n]/S[n];
    a1[3]:=teta*(x2+x4)+xc/S[n+1];
    a1[4]:=teta-teta*x2*Q[n+1]*derD[n+1]/D[n+1]-
            (teta*x4+xc/S[n+1])*Q[n+1]*B[n+1]/S[n+1];
    a1[5]:=Z[n]-Z[n+1]-0.5*(x1-x3)*Q[n]-(x2+x4)*Q[n+1];
    efe:=dx/(2*teta*dts);
    a2[1]:=-1;
    a2[2]:=efe*B[n];
    a2[3]:=1;
    a2[4]:=efe*B[n+1];
    a2[5]:=(Q[n]-Q[n+1])/teta
  end;
```

```
procedure compact1(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r;
                   var be:array12r);

(* asigna valores numericos a las variables si el punto de discretizacion
   m es extremo abierto aguas arriba con condicion de contorno Q,
   o extremo abierto aguas abajo con condicion de contorno Z, o
   extremo cerrado aguas abajo perteneciente a un nodo de afluencia
   de la forma {ubica[m]-1,m,ubica[m] }  *)

  var k:1..3;

  begin
    for k:=1 to 3 do
      begin
        ze[j+k+i1]:=a1[k+i3];
        ze[j+k+3+i1+i2]:=a2[k+i3]
      end;
    if(i2=0) then
      begin
        be[i]:=a1[5]-dd*a1[4-3*i3];
        be[i+1]:=a2[5]-dd*a2[4-3*i3]
      end
            else
      begin
        be[i]:=a1[5];
        be[i+1]:=a2[5];
        ze[j+4-3*i1]:=a1[4];
        ze[j+8-3*i1]:=a2[4]
      end;
    j:=j+6+2*i2
  end;

procedure compact2(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r; var be:array12r);

  (* asigna valores numericos a las variables para un punto de discretizacic


     m extremo abierto aguas arriba con condicion de contorno Z,
     o extremo abierto aguas abajo con condicion de contorno  Q,
     o extremo cerrado aguas arriba perteneciente a un nodo de efluencia
     de la forma {ubica[m]-1,m,ubica[m]} *)

  var k:integer;

  begin
    ze[j+1+i1+2*i3]:=a1[1+3*i3];
    ze[j+4+i1+i2+2*i3]:=a2[1+3*i3];
    for k:=1 to 2 do
      begin
        ze[j+k+i1+1-i3]:=a1[k+2-2*i3];
        ze[j+k+i1+4+i2-i3]:=a2[k+2-2*i3]
      end;
    if(i2=1) then
      begin
        be[i]:=a1[5];
        ze[j+4-3*i1]:=a1[2];
        be[i+1]:=a2[5];
        ze[j+8-3*i1]:= a2[2]
      end
```

```
            else
      begin
        be[i]:=a1[5]-dd*a1[2+i3];
        be[i+1]:=a2[5]-dd*a2[2+i3]
      end;
    j:=j+6+2*i2
  end;

procedure compact3(i1,i2,i3:integer;i:intnpmx2;a1,a2:array15;dd:real;
                   var j:intnpmx5;var ze:array15r; var be:array12r);

   (* asigna valores numericos en la matriz para punto de discretizacion
      m interior a tramo simple, o punto extremo cerrado aguas arriba
      perteneciente a nodo de afluencia, o punto extremo cerrado
      aguas arriba perteneciente a nodo de efluencia de la forma
      {m-1,ubica[m],m} *)

  var k:integer;

  begin
    for k:=1 to 4 do
      begin
        ze[j+k+i1]:=a1[k];
        ze[j+k+4+i1+i2]:=a2[k]
      end;
    if((i2=0) and (i3=0)) then
      begin
        be[i]:=a1[5]-dd*a1[1];
        be[i+1]:=a2[5]-dd*a2[1]
      end
                     else
      begin
        be[i]:=a1[5];
        be[i+1]:=a2[5];
        if(i2=1) then
          begin
            ze[j+5-4*i1]:=a1[1]*i3;
            ze[j+10-4*i1]:=a2[1]*i3
          end
      end;
    j:=j+8+2*i2
  end;




procedure kacz(imax:intnpmx2;v:array12i; w:array15i;be:array12r;ze:array15r
               var x:array12r);

   (* metodo de las proyecciones modificado *)

type array2r=array[1..2] of real;
  const iter=100000.0;
  var i,ik,i1,j,l: integer;prod,lamd:array2r;  pivote:arraymedio;
                k,as1,as2,as3,epsi,eps,coef:real;asq,xant:array12r;
              vv:integer;              imx2:intmedio;
              ast:array[1..3,intmedio] of real;
              switch:arraylog;
```

```
procedure noconverge;

 begin
   writeln('El procedimiento de Kacmarz con semiciclos no converge ');
   close(salida);
   close(contorno);
   halt
 end;

procedure prodesc(j,vv:integer;ze:array15r;w:array15i;x:array12r;
                  b1,b2,pivote:real;
                  switch:boolean;var prod:array2r);
var i1,l:integer;aux:real;
 begin
   prod[1]:=0;
   prod[2]:=0;
   for i1:=1 to vv do
     begin
       l:=w[j+i1];
       prod[1]:=prod[1]+ze[j+i1]*x[l];
       prod[2]:=prod[2]+ze[j+i1+vv]*x[l]
     end;
   if(switch) then
     begin
       aux:=prod[1];
       prod[1]:=prod[2];
       prod[2]:=aux
     end;
   aux:=prod[1];
   prod[1]:=b1-aux;
   prod[2]:=b2-prod[2]+pivote*aux
 end;


procedure invers(var a1,a2,a3,b1,b2,pivote:real;var switch:boolean);
var aux:real;

begin
  if(a1<abs(a3)) then
    begin
      switch:=true;
      pivote:=a1/a3;
      a1:=a3;
      a3:=a2;
      a2:=a1-pivote*a3;
      aux:=b1;
      b1:=b2;
      b2:=aux-pivote*b1
    end
                    else
    begin
      switch:=false;
      pivote:=a3/a1;
      a2:=a2-pivote*a3;
      b2:=b2-pivote*b1
    end
end;
```

```
begin                                                    (* kacz *)
  imx2:=imax div 2;
  k:=0;
  j:=0;
  for i:=1 to imx2 do
    begin
      x[2*i-1]:=0;
      x[2*i]:=0;
      as1:=0;
      as2:=0;
      as3:=0;
      vv:=v[2*i];
      for i1:=1 to vv do
        begin
          as1:=as1+sqr(ze[j+i1]);
          as2:=as2+sqr(ze[j+i1+vv]);
          as3:=as3+ze[j+i1]*ze[j+i1+vv]
        end;
      j:=j+2*vv;
      xant[2*i-1]:=0;
      xant[2*i]:=0;
      ast[1,i]:=as1;
      ast[2,i]:=as2;
      ast[3,i]:=as3;
      invers(ast[1,i],ast[2,i],ast[3,i],be[2*i-1],be[2*i],pivote[i],
                switch[i])
    end;
  eps:=converg+1;
  ik:=0;
  j:=0;
  repeat
    ik:=ik+1;
    if(ik>imx2) then begin ik:=1;j:=0 end;
    k:=k+1;
    vv:=v[2*ik];
    prodesc(j,vv,ze,w,x,be[2*ik-1],be[2*ik],pivote[ik],switch[ik],prod);
    lamd[2]:=prod[2]/ast[2,ik];
    lamd[1]:=(prod[1]-ast[3,ik]*lamd[2])/ast[1,ik];
    for i1:=1 to vv do
      begin
       l:=w[j+i1];
       x[l]:=x[l]+lamd[1]*ze[j+i1]+lamd[2]*ze[j+i1+vv]
      end;
       if (ik=imx2) then
          begin
            eps:=0;
            for i:=1 to imax do
              begin
                epsi:=abs(xant[i]-x[i]);
                xant[i]:=x[i];
                if(epsi>eps) then eps:=epsi
              end
          end;
     j:=j+2*vv
  until ((k>=iter) or (eps<converg));writeln(k:15:0,eps:15:5);
  if((k>=iter) and (eps >=converg)) then noconverge
end;
```

```
procedure recupera(npuntos:intnpmx;np2,codigo,ubica:arraypxi;x:array12r;
                   dat:arrayd; var Q,Z:arraypx);

   (* asigna valores de los resultados a Q y Z actualizados *)

  var m:intnpmx;j:intnpmx2;
  begin
    for m:=1 to npuntos do
      begin
        j:=np2[m];
        if(codigo[m]=11) then
          begin
            Z[m]:=Z[m]+x[j];
            Q[m]:=dat[ubica[m]]
          end
                         else
        if(codigo[m]=12) then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=
            dat[ubica[m]]
          end
                         else
        if(codigo[m]=0)  then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=Z[m]+x[j+1]
          end
                          else
        if(codigo[m]=20) then
          begin
            Q[m]:=Q[m-1]+dat[ubica[m]];
            Z[m]:=Z[m-1]
          end
                          else
        if(codigo[m]=21) then
          begin
            Q[m]:=Q[m]+x[j-1];
            Z[m]:=Z[m]+x[np2[ubica[m]-1]+1]
          end
                          else
        if(codigo[m]=51) then
          begin
            Q[m]:=Q[m]+x[j];
            Z[m]:=Z[m]+x[np2[ubica[m]-1]+1]
          end
                          else
        if(codigo[m]=91) then
          begin
            Q[m]:=dat[ubica[m]];
            Z[m]:=Z[m]+x[j-1]
          end
                          else
        if(codigo[m]=92) then
          begin
            Z[m]:=dat[ubica[m]];
            Q[m]:=Q[m]+x[j-1]
          end
      end;
```

```
    for m:=1 to npuntos do
        if(codigo[m]=25) then
          begin
            Z[m]:=Z[m-1];
            Q[m]:=Q[m-1]+Q[ubica[m]]
          end
                         else
        if(codigo[m]=55) then
          begin
            Z[m]:=Z[m-1];
            Q[m]:=Q[m-1]-Q[ubica[m]]
          end
  end;

begin                                              (* matriz *)
  j:=0;
  i:=1;
  for m:=1 to npuntos-1 do
    if((codigo[m]<>20) and (codigo[m]<>21) and (codigo[m]<>25)
       and (codigo[m]<>55) and (codigo[m]<90)) then
         begin
           if(codigo[m]=51) then ik:=0 else ik:=1;
           if(ubica[m+ik]<m) then i1:=1 else i1:=0;
           coefis(m,dts,teta,xcoor,Q,Z,S,B,D,derD,a1,a2);
           if(codigo[m]=11) then compact1(0,0,1,i,a1,a2,
              dat[ubica[m]]-Q[m],j,ze,be)
                            else
           if(codigo[m+1]=92) then compact1(0,0,0,i,a1,a2,
              dat[ubica[m+1]]-Z[m+1],j,ze,be)
                            else
           if(codigo[m+1]=21) then compact1(i1,1,0,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m]=12) then compact2(0,0,0,i,a1,a2,
               dat[ubica[m]]-Z[m] ,j,ze,be)
                            else
           if(codigo[m+1]=91) then compact2(0,0,1,i,a1,a2,
               dat[ubica[m+1]]-Q[m+1],j,ze,be)
                            else
           if(codigo[m]=51) then compact2(i1,1,0,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m+1]=20) then compact3(0,0,0,i,a1,a2,
                dat[ubica[m+1]]-(Q[m+1]-Q[m]),j,ze,be)
                            else
           if(codigo[m+1]=25) then compact3(i1,1,1,i,a1,a2,0.,j,ze,be)
                            else
           if(codigo[m+1]=55) then compact3(i1,1,-1,i,a1,a2,0.,j,ze,be)
                            else
                                 compact3(0,0,1,i,a1,a2,0.,j,ze,be);
         i:=i+2
       end;
  kacz(imax,v,w,be,ze,x);
  recupera(npuntos,np2,codigo,ubica,x,dat,Q,Z)
end;
```

```
begin                                                    (* delta2 *)
  write('Nombre del archivo de resultados ');
  readln(contor);
  assign(salida,contor);
  rewrite(salida);
  write('Tolerancia ');
  readln(converg);
  leemodelo(npuntos,ntab,nombre,tablaH,xcoor,cota,codigo,ubica,tablaB,
            tablaD) ;
  leeinic(npuntos,Q,Z);
  leeborde(npuntos,nombre,xcoor,cota,ubica,codigo,ndat,impri,tinic,dt,
             tfin,unidad);
  imp:=0;
  write('Archivo de condiciones de contorno ');
  readln(contor);
  assign(contorno,contor);
  reset(contorno);
  precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,tablaD,B,derD,S,D);
  imprime(unidad,tinic,npuntos,nombre,Z,Q,S);
  armado(npuntos,codigo,ubica,np1,np2,np3,v,w,jmax,imax);
  leedat(ndat,dat1,t1);
  leedat(ndat,dat2,t2);
  if (unidad=' minutos') then dts:=dt*60.0
                         else if(unidad=' horas') then dts:= dt*3600.0
                         else dts:= dt*86400.0;
   t:= tinic+dt;
   imp:=1;
   while ( t <= tfin+0.0001 ) do
                  begin
                    interdat(t,ndat,dat,dat1,dat2,t1,t2);
                    matriz(npuntos,imax,teta,dts,S,B,D,derD,codigo,
                           ubica,dat,v,w,Q,Z);
                    precoefis(npuntos,ntab,Z,cota,tablaH,tablaB,
                           tablaD,B,derD,S,D) ;
                    if(imp=impri) then
                                  begin
                                    imprime(unidad,t,npuntos,nombre,
                                            Z,Q,S);
                                    imp:=0
                                  end;
                    t:=t+dt;
                    imp:=imp+1
                  end;
  close(salida);
  close(contorno)
end.
```