Resumen

La Investigación en Ciencias de la Tierra se realiza siguiendo cuatro grandes direcciones: la observación de eventos naturales realizada en el campo, la experimentación en laboratorio para reproducir fenómenos-modelo, la modelación teórica por medio de ecuaciones matemáticas complejas, y finalmente la experimentación numérica. Cada una de las metodologías anteriormente señaladas tiene sus ventajas. Hoy en día, es fundamental combinar estos diferentes puntos de vista de la investigación para mejorar la comprensión de cada fenómeno. La manera más natural para combinar teoría y observaciones está en la utilización de datos recolectados en los modelos numéricos. Este libro presenta las etapas de construcción de una herramienta de optimización que permite introducir datos en la modelación. Estos métodos son generales y han sido utilizados desde hace unos 20 años en varios campos de la Investigación en Ciencias de la Tierra. Para darle un tono general hemos escrito cada capítulo introduciendo paulatinamente los elementos de Física, Informática y Matemática necesarios para la modelación, el análisis de sensibilidad y la optimización. En cada uno de estos pasos se ha expuesto la metodología general y su aplicación a un sistema de ecuaciones de mecánica de los fluidos. Es de esperarse que el aplicación a otros problemas de Ciencias de la Tierra resulte claro a través del ejemplo de las columnas eruptivas plinianas. Activamente estudiado en vulcanología física, este ejemplo se presta para ilustrar la aplicación de los métodos expuestos. Considerando la relación entre nuestra sociedad y el ambiente, no es solamente importante la observación o la previsión de los fenómenos naturales, sino una mayor comprensión de cada uno de estos. La reconstrucción de fenómenos pasados tales como la detección de contaminación del medio ambiente, el seguimiento de eventos actuales para la planeación de la evacuación de poblaciones, el diseño de dispositivos (*e.g.* diques) con impacto mínimo sobre el medio ambiente, o la predicción en meteorología y climatología son ejemplos de procesos que pueden ser modelados y optimizados siguiendo la metodología aquí expuesta.

Abstract

Earth Sciences Research is conducted in four great directions: field observation of natural events, laboratory experimentation to reproduce "model" phenomena theoretical modelling by means of complex mathematical equations, and finally numerical experimentation. Each of these methodologies has its own advantages. Nowadays, it is fundamental to combine these different points of view to improve the understanding of each phenomenon. The natural way to combine theory and observations is in the use of data in the numerical models. This book presents the stages of construction of an optimization tool that allows for the introduction of data in the modelling. These methods are general and have been used for about 20 years in several Earth Science fields. To give it a general tone we wrote each chapter introducing gradually the elements of Physics, Computer Science and Mathematics necessary to the modeling, the analysis of sensitivity and the optimization. Each of these steps exhibits the general methodology and its application to a system of equations in fluid mechanics. We expect that the application to other Earth Science problems results clear through the example of plinians eruptive columns. Actively studied in physical volcanology, this exemple is used to illustrate the application of the discussed methods. Considering the relationship between our society and the ambient environment, observing or forecasting the natural phenomena are important, but a greater understanding of each of these is desirable. The reconstruction of past phenomena such as the detection of anthropic contamination, the monitoring of present events for planning emergency population evacuation, the design of devices (sea walls) with minimal impact on the environment, or the prediction in meteorology and climatology are examples of processes that can be modeled and be optimized following the methodology presented herein.

1. Introducción

La Investigación en Ciencias de la Tierra se realiza principalmente siguiendo cuatro grandes direcciones. En primer lugar la observación de los eventos naturales realizada en el campo, aun desde la antigüedad, por medio de varios instrumentos dedicados a seguir sus cursos. Hoy en día, se deducen de estas observaciones informaciones cuantitativas tales como la velocidad, la temperatura, la presión (durante el evento) o el volumen y dimensiones de los depósitos geológicos. Se consignan también datos cualitativos (impacto sobre el ambiente: colores, texturas, ...). Por supuesto esas informaciones dependen del fenómeno observado, no se miden las mismas cosas para el océano que para el aire o la Tierra sólida. Se hace notar que las observaciones son en general parciales, por lo que no se puede acceder a todas las partes del dominio espacio-temporal donde ocurió el evento. Además, los eventos no son reproducibles por los investigadores (a excepción de avalanchas causadas intencionalmente como medida preventiva, por ejemplo). En segundo lugar se puede usar la experimentación en laboratorio para intentar reproducir fenómenos en una caja dedicada, es decir a una escala más pequeña. Se construye un fenómeno-modelo por medio de instrumentos mecánicos que permiten aproximarse al fenómeno real. Eso permite hacer muchos experimentos a partir de condiciones diferentes para medir el impacto de cada factor (velocidad inicial de un flujo, temperatura del ambiente en la caja, ...). Se agregan herramientas de medición y de observación (cámaras) para seguir el desarrollo de cada ocurrencia (reproducción) del mismo fenómeno. Las ventajas del estudio de los fenómenos desarrollados en laboratorio son que se conocen, de una manera precisa, las condiciones experimentales iniciales y que se puede observar el fenómeno de una manera más completa (aunque partes del fenómeno pueden estar todavía ocultas). Así se puede conseguir un conjunto muy completo, compuesto para cada experimento, de las condiciones experimentales utilizadas y de las observaciones resultantes.

Finalmente, se encuentran los métodos teóricos. La modelación teórica se realiza por medio de ecuaciones matemáticas complejas basadas en conceptos físicos. A menudo, la complejidad de un fenómeno se puede traducir en ecuaciones, pero sus soluciones son generalmente tan difíciles de calcular que se buscan sólo algunas de sus propiedades (existencia y unicidad de la solución, por ejemplo). Por eso los matemáticos utilizan suposiciones en el dominio del cálculo y las ecuaciones que pueden amenazar la complejidad del problema matemático. Es decir, lo alejan del problema real. Así se puede, por ejemplo, trabajar con océanos cuadrados (de largo unitario) que aparecen como una caricatura de un oceáno real. A pesar de esas desventajas, este trabajo teórico resulta muy importante porque pueden deducirse generalizaciones, justificadas al menos físicamente, y conducir a la modelación numérica. La modelación y la experimentación numérica consisten en discretizar las ecuaciones en los espacios matemáticos adecuados (este trabajo requiere también un estudio teórico) para construir un código informático que permita calcular soluciones numéricas del conjunto de ecuaciones matemáticas eventualmente más complejo que el conjunto utilizado para resaltar las propiedades matemáticas. Una vez construido, el modelo puede ser utilizado para la experimentación numérica, es decir para la reconstrucción numérica de fenómenos naturales. Como en el caso de la experimentación en laboratorio, los fenómenos pueden ser simulados muchas veces a partir de conjuntos de datos diferentes. Además, las soluciones calculadas son enteramente conocidas sobre la malla de discretización lo que permite medir exactamente (desde un punto de vista numérico) el valor de cada variable. Por otro lado, se pueden agregar a los códigos herramientas que permitan buscar datos de entrada que fuercen la

solución para verificar algunas propiedades.

Los problemas que se presentan a los investigadores, y más generalmente a la sociedad (gobierno, empresas, población), no son solamente la observación o la reproducción de fenómenos naturales, sino una mayor comprensión de cada fenómeno para, finalmente, poder más o menos "controlarlo". Por supuesto, no se pueden controlar todos los fenómenos porque son generalmente de una potencia superior a la potencia humana. Entonces, lo que se busca son acciones de reconstrucción de fenómenos pasados tales como la detección de contaminación del medio ambiente de origen antrópico, el seguimiento de eventos actuales para la planeación de evacuaciones de la población o el diseño de dispositivos (puerto, dique) con impacto mínimo sobre el medio ambiente, y la predicción de fenómenos futuros en meteorología y climatología por ejemplo. Obviamente se puede reflexionar en más posibilidades ya que la modelación nos permite una mayor libertad en los estudios. Hoy en día, es fundamental combinar los diferentes puntos de vista de la investigación para mejorar la comprensión de cada fenómeno. Cada metodología (observación, experimentación de laboratorio, modelación teórica, modelación y experimentación numérica) tiene sus ventajas, queda a los investigadores conjugarlas para aprovechar lo mejor de ellas. La manera más natural para combinar teoría y observaciones está en la utilización de datos recolectados en los modelos numéricos. En este trabajo se desarrollan teóricamente y numéricamente algunos métodos matemáticos que lo permiten. En muchos dominios de la investigación en Ciencias de la Tierra, se llaman métodos de asimilación de datos. Recíprocamente, los modelos numéricos, aprovechando los datos, pueden completar el conocimiento del fenómeno reconstruido, dando acceso a más información.

El libro estudia la modelación, el análisis de sensibilidad y la modelación inversa de las columnas plinianas. En el capítulo 2 se pone mucha atención en la descripción del modelo del fenómeno, con el fin de escribirlo en forma sencilla, en un sistema general de ecuaciones matemáticas. Utilizamos un modelo unidimensional para enseñar que los modelos de columnas bi o tridimensionales entran en el formalismo genérico también. Así, mostramos que las herramientas pueden aplicarse a muchos otros problemas de Ciencias de la Tierra y del Universo. El capítulo 3 está dedicado al análisis de sensibilidad mostrando cómo construir códigos lineales por medio de herramientas de diferenciación automática. Se completa con un análisis en el contexto de las erupciones plinianas. El capítulo 4 trata de la modelación inversa estilo "asimilación de datos". Se introduce el famoso modelo adjunto para solucionar este problema de optimización numérica por medio de un método de gradiente. Como ejemplos, consideramos dos problemas inversos diferentes. El primero introduce "experimentos gemelos" cuyas observaciones son producidas por el modelo estudiado. Eso permite comprobar la validez del proceso de optimización. El segundo trata datos reales. El librito se completa con una extensa bibliografía y un conjunto de códigos informáticos escritos en `Fortran`. Estos códigos pueden descargarse desde la página `www.lpmm.fr/charpentier/Libro/`

2. Modelación Pliniana

La formación de columnas plinianas es un tema fundamental en vulcanología y climatología. Estas columnas son mezclas de vapor, otros volátiles, agua líquida, y fragmentos sólidos a altas temperaturas y expulsados a grandes velocidades durante las erupciones volcánicas explosivas. A través de este mecanismo grandes cantidades de partículas y de aerosoles son inyectados en la alta atmósfera. En la región baja (Fig. 2.1), llamada zona de empuje, el movimiento de la columna está dominado por la diferencia de presión entre la mezcla que sale del conducto volcánico y la presión de la atmósfera. Forzada a través de la atmósfera, la mezcla pierde energía cinética a la vez que ingiere aire del ambiente. El aire atmosférico ingerido es incorporado a la columna y calentado. Dado que el aire al calentarse disminuye su densidad, la mezcla puede adquirir densidades inferiores a las del aire circundante y en consecuencia adquirir flotación, lo que le permite alcanzar altitudes que rebasan la tropopausa. La parte de la columna cuya dinámica está dictada por las fuerzas de flotación se conoce por lo tanto como zona de flotación.

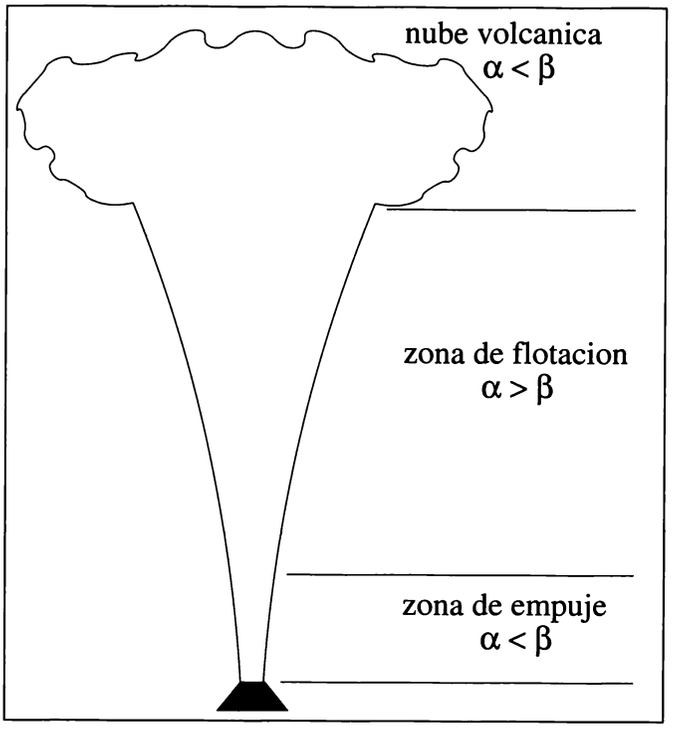

Fig. 2.1 Columna pliniana y sus tres zonas definidas por la diferencia de densidad entre la mezcla volcánica (β) y la densidad del aire ambiente (α).

Los procesos que gobiernan la dinámica de la columna son complejos, extendiéndose de las micro escalas a las macro escalas. Los primeros modelos fueron propuestos hace tres décadas suponiendo que los piroclastos son granos finos. Bajo esta hipótesis, la transferencia de calor y de momento entre las partículas y el gas es rápido. Así, es posible considerar a la mezcla como homogénea en temperatura y velocidad, y trabajar con una sola fase. A esta aproximación se le llama a veces "aproximación de pseudo-flujo". Wilson (1976), Wilson et al. (1978) y Settle (1978) basaron sus modelos en las ecuaciones de conservación de masa, de momento y de energía del movimiento de un fluido en otro siguiendo las teorías de la mecánica de fluidos aplicadas a las toberas de reacción (jets) (Prandtl, 1949) y de las plumas térmicas (Morton, Taylor y Turner, 1956). Sparks (1986) mejoró los modelos anteriores tomando en cuenta la influencia de la temperatura del magma y los perfiles atmosféricos verticales de presión y de temperatura. En un modelo más sofisticado, Woods (1998) aunó los modelos del chorro y de pluma para proponer una formulación llamada de sombrero de copa ("top hat") derivada de la conservación de la energía y del impulso, así como la hipótesis de la ingestión de aire por arrastre introducida por Morton, Taylor y Turner (1956).

Por otra parte, los modelos polifásicos (Glaze, Baloga y Wilson, 1997)han sido desarrollados para considerar procesos importantes tales

como el transporte de las partículas y los efectos de condensación/evaporación. Una revisión de los modelos dedicados a la formación y la dinámica de las columnas se presenta en Sparks et al. (1997). Existen varios modelos bidimensionales o tridimensionales con evolución temporal (Valentine y Wolhetz, 1989; Dobran, Neri y Macedonio, 1993; Oberhuber et al., 1998). Entre ellos, ATHAM (Active Tracer High resolution Atmospheric Model) propuesto en (Herzog et al., 1998; Oberhuber, 1998) calcula la solución de las ecuaciones de Navier-Stokes en el caso no-hidrostático. Recientemente, los esfuerzos científicos fueron dedicados al diseño de modelos complejos que manejan elementos de vulcanología que se extienden de los fenómenos de micro escala a los de macro escala. El trabajo común propuesto en (Textor, 2005) es probablemente el más impresionante ya que contiene modelos que van de los procesos del conducto a los efectos meteorológicos y climatológicos de la dispersión de cenizas y aerosoles.

2.1. Modelo unidimensional monofásico

En este libro, utilizaremos el modelo "top hat" desarrollado por Woods (1998)puesto que es el más general entre los modelos monofásicos. El modelo es unidimensional por lo que las variables como radio, velocidad, temperatura y densidad de la columna son funciones de la altura. La gráfica 2.2 ilustra las consideraciones básicas del modelo y explica su nombre de "sombrero de copa". Esta simplificación para un proceso que es netamente turbulento se justifica al considerar que el promedio de las variables que intervienen es aproximadamente constante punto a punto. Por otro lado el modelo es así mismo estacionario, lo cual haya su justificación en el hecho de que durante las erupciones plinianas la columna es sostenida por tiempos lo suficientemente largos con respecto a las etapas inicial y final para considerarse estacionario.

Como se ha establecido, las ecuaciones se escriben en términos del radio r, de la velocidad u, de la temperatura θ y de la densidad efectiva de la mezcla β, relacionada con la fracción de gas n, de la columna. Las condiciones a la frontera son r_0 (m), u_0 $(m.s^{-1})$, θ_0 (K), y n_0 al nivel del cráter $z = z_0$ (m). El sistema de ecuaciones depende de parámetros volcánicos tales como:

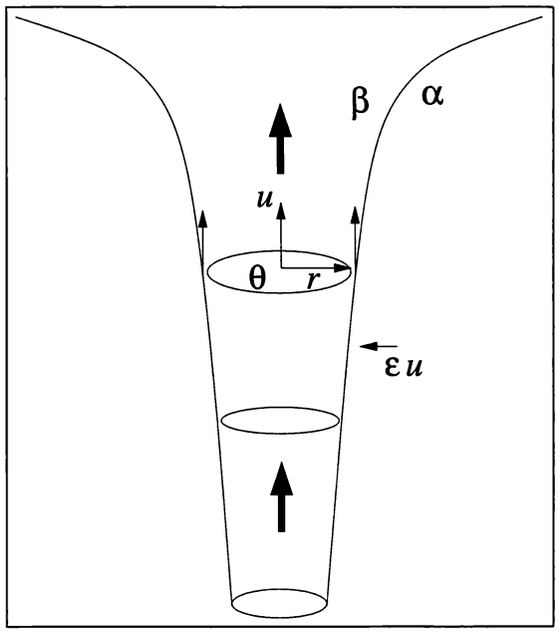

Fig. 2.2 Columna pliniana y sus variables.

- la constante de gas volcánico R_{g_0} $(J.mol^{-1}.K^{-1})$,
- el calor específico C_{p0} $(J.kg^{-1}.K)$ de la mezcla en la columna,
- la densidad de los piroclastos σ $(kg.m^{-3})$, y
- la función de ingestión ε que representa la forma media en que el aire es atrapado por la columna,

y de parámetros físicos como:

- la temperatura T (K), la presión P (Pa) y la densidad del aire ambiente α calculadas por otro medio (a partir del modelo de atmósfera estándar del párrafo 2.4 por ejemplo),
- la constante de gas del aire R_a $(J.mol^{-1}.K^{-1})$,
- el calor específico del aire C_a $(J.kg^{-1}.K)$, y
- la aceleración de la gravedad g $(kg.m.s^{-2})$.

Las variables R_g y C_p son estimadas a lo largo de la columna. Los valores de todos los datos (parámetros y otras constantes) están juntos en un archivo `datos.inc` (ver párrafo 2.5) agregado a todos los archivos que utilizaremos.

Usando estas notaciones y omitiendo la dependencia de las variables con la altura z, las ecuaciones son:

- conservación de la masa a lo largo de la columna (z_0, z_{top}):

$$\frac{d\beta u r^2}{dz} = 2\varepsilon u r \alpha, \tag{2.1}$$

- conservación del momento:

$$\frac{du(\beta u r^2)}{dz} = g(\alpha - \beta) r^2,$$

es decir:

$$\frac{du}{dz}(\beta u r^2) + u\frac{d\beta u r^2}{dz} = g(\alpha - \beta) r^2.$$

Utilizando (2.1) se deduce una ecuación diferencial para la velocidad:

$$\frac{du}{dz} = \frac{g(\alpha - \beta) r^2}{\beta u r^2} - u\frac{2\varepsilon u r \alpha}{\beta u r^2}, \tag{2.2}$$

- conservación de la energía (entalpía):

$$\frac{d}{dz}\left(\beta u r^2 \left(C_p\theta + \frac{u^2}{2} + gz\right)\right) = \\ = 2\varepsilon u r \alpha (C_a T + gz),$$

es decir:

$$\frac{d(\beta u r^2)}{dz}\left(C_p\theta + \frac{u^2}{2} + gz\right) + \\ + (\beta u r^2)\left(\frac{d(C_p\theta)}{dz} + u\frac{du}{dz} + g\right) = \\ = 2\varepsilon u r \alpha (C_a T + gz),$$

o sea, utilizando (2.1) y (2.2):

$$\frac{d(C_p\theta)}{dz} = \\ = \frac{1}{\beta u r^2}\left(2\varepsilon u r \alpha (C_a T + gz)\right) - \\ - \frac{1}{\beta u r^2}\frac{d(\beta u r^2)}{dz}\left(C_p\theta + \frac{u^2}{2} + gz\right) - \\ - u\frac{du}{dz} - g.$$

Se deduce:

$$\frac{dC_p\theta}{dz} = \\ = \frac{2u\varepsilon r \alpha}{\beta u r^2}\left(C_a T - C_p\theta - \frac{u^2}{2}\right) - \\ - u\left(\frac{g(\alpha - \beta) r^2}{\beta u r^2} - u\frac{2\varepsilon u r \alpha}{\beta u r^2}\right) - g. \tag{2.3}$$

Al final, el modelo de Woods (1988) se presenta así:

- ecuaciones diferenciales:

$$\begin{cases} \dfrac{d\beta u r^2}{dz} = 2\varepsilon u r \alpha, \\ \dfrac{du}{dz} = \dfrac{g(\alpha - \beta) r^2}{\beta u r^2} - u\dfrac{2\varepsilon u r \alpha}{\beta u r^2}, \\ \dfrac{dC_p\theta}{dz} = \dfrac{2u\varepsilon r \alpha}{\beta u r^2}\left(C_a T - C_p\theta - \dfrac{u^2}{2}\right) - \\ - u\left(\dfrac{g(\alpha - \beta) r^2}{\beta u r^2} - u\dfrac{2u\varepsilon r \alpha}{\beta u r^2}\right) - g, \end{cases} \tag{2.4}$$

- relaciones funcionales entre variables y parámetros:

$$\begin{cases} \dfrac{1}{\beta} = (1 - n)\dfrac{1}{\sigma} + \dfrac{n R_g \theta}{P}, \\ R_g = R_a + \\ + (R_{g0} - R_a)\left(\dfrac{1 - n}{n}\right)\left(\dfrac{n_0}{1 - n_0}\right), \\ n = 1 + (n_0 - 1)\dfrac{u_0 \beta_0 r_0^2}{u \beta r^2}, \\ C_p = C_a + (C_{p_0} - C_a)\dfrac{(1 - n)}{(1 - n_0)}. \end{cases} \tag{2.5}$$

Utilizamos el código del párrafo 2.5 para conseguir las figuras 2.3–2.6. Se pueden reproducir usando los archivos puestos en la página Internet `www.lpmm.fr/charpentier/Libro/Pliniano`.

La figura 2.3 muestra la velocidad a lo largo de la columna.

Nótese que la velocidad disminuye hasta un mínimo, éste define la zona de empuje. A partir de este punto el movimiento ascendente de la pluma es dominado por las fuerzas de flotación. La velocidad vuelve a crecer, luego disminuye hasta hacerse nula en la cima de la columna donde la densidad de la columna se iguala con la densidad del aire circundante y cesan las fuerzas de flotación. Esta es la zona de flotación.

La figura 2.4 muestra la variación del radio promedio de la columna con la altura, ésta es casi una representación de la forma de la columna; nótese en la parte superior la formación del hongo característico de la pluma. En esta zona la

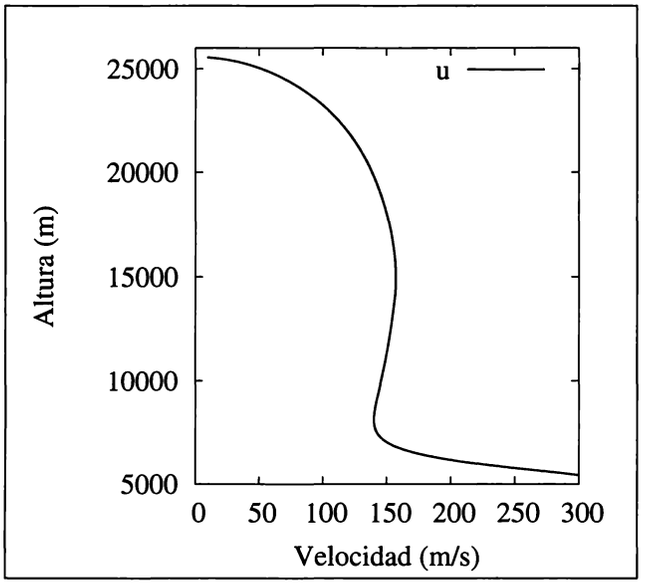

Fig. 2.3 Velocidad de la mezcla en la columna.

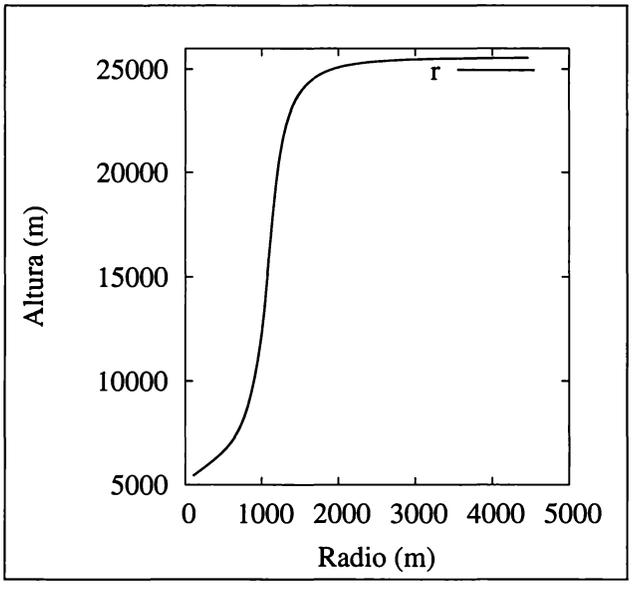

Fig. 2.4 Radio de la mezcla en la columna.

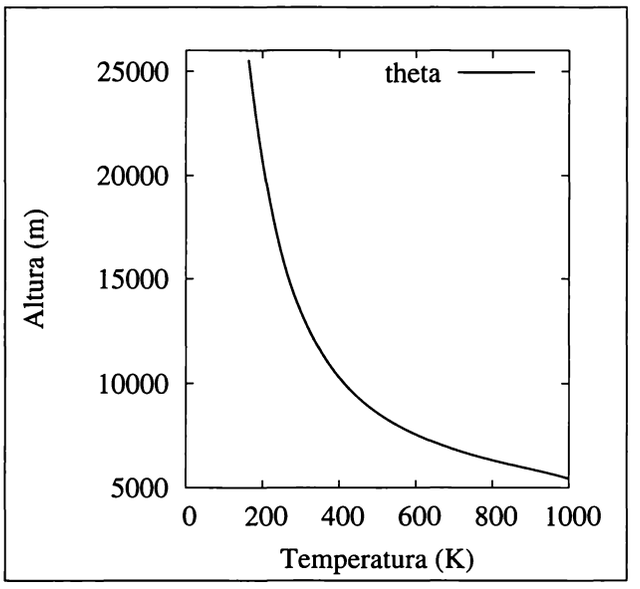

Fig. 2.5 Temperatura de la mezcla en la columna.

componente vertical de la velocidad es casi nula y el movimiento es prácticamente horizontal..

La temperatura de la columna (Fig. 2.5) decrece continuamente por efecto de la expansión de la columna y la ingestión de aire).

Finalmente la Fig. 2.6 muestra la variación de la densidad y fracción de gas a lo largo de la columna. La densidad disminuye rápidamente en la zona de empuje y disminuye en la zona de flotación por efecto del enfriamiento de la columna. La fracción de gas aumenta rápidamente en la base de la columna y posteriormente permanece casi constante debido a que la incorporación de aire es función de la velocidad en la columna, misma que disminuye a partir de cierta altura en zona de flotación.

A continuación presentamos cómo se puede implementar el modelo (2.4)–(2.5).

2.2. Aspectos generales

Considerando la solución numérica del problema planteado en el apartado anterior, se nota que:

- las 4 variables de estado que buscamos a lo largo de la columna son el radio r, la velocidad u, la temperatura θ y la fracción de gas

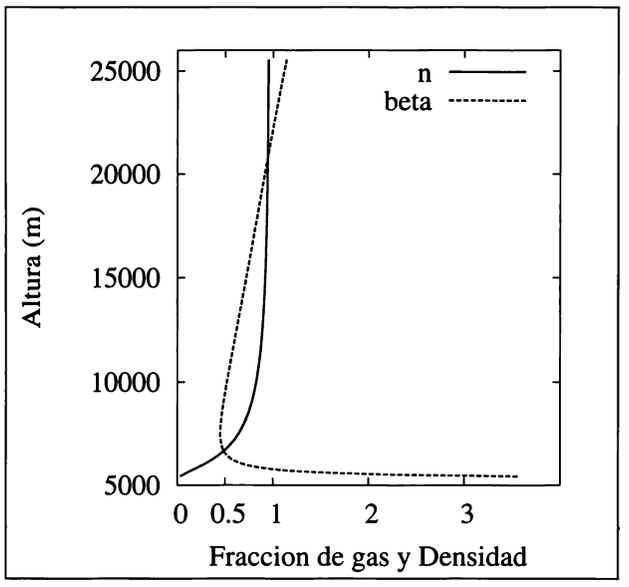

Fig. 2.6 Fracción de gas y densidad de la mezcla en la columna.

n. No aparecen solas en los términos izquierdos de las ecuaciones diferenciales ordinarias (2.4),

- el sistema (2.4) tiene tres ecuaciones diferenciales ordinarias y 4 incógnitas,
- las variables β y n parecen interdependientes en (2.5).

Para remediar estas deficiencias, se puede trabajar un poco más construyendo un sistema de 4 ecuaciones diferenciales ordinarias con 4 incógnitas (Charpentier, to appear).El trabajo se hace en 4 etapas.

1. Cambio de notación. Para simplificar la escritura del sistema se introducen 4 nuevas variables:

$$\begin{cases} F = \beta u r^2, & \text{flujo de masa,} \\ E = C_p \theta, & \text{energía,} \\ D = (\alpha - \beta)/\beta, & \text{densidad relativa,} \\ C = 2\varepsilon\alpha u r/(\beta u r^2). \end{cases} \tag{2.6}$$

El sistema (2.4) se vuelve:

$$\begin{cases} \dfrac{dF}{dz} = FC, \\ \dfrac{du}{dz} = gD/u - Cu, \\ \dfrac{dE}{dz} = C(C_a T - E - u^2/2) - \\ \qquad -u(gD/u - Cu) - g. \end{cases} \tag{2.7}$$

2. Derivación de una ecuación diferencial ordinaria para la fracción de gas. De hecho la diferenciación de la tercera ecuación de (2.5) conduce a:

$$\begin{aligned} \frac{dn}{dz} &= \tfrac{d}{dz}(1 + (n_0 - 1)F_0/F) = \\ &= -(n_0 - 1)F_0 \frac{dF/dz}{F^2} = \\ &= (1 - n_0)\frac{F_0 C}{F}. \end{aligned} \tag{2.8}$$

3. Deducción de las condiciones de salida nuevas a través del cambio biyectivo $\mathcal{T}$ que mapea $(r_0, u_0, \theta_0, n_0)$ en (F_0, u_0, E_0, n_0) usando:

$$\begin{cases} F_0 = \dfrac{P_0 \sigma u_0 r_0^2}{P_0(1 - n_0) + n_0 R_{g0} \theta_0 \sigma}, \\ E_0 = C_{p0}\theta_0. \end{cases} \tag{2.9}$$

4. Selección de las relaciones funcionales para que no sean interdependientes durante el cálculo numérico:

$$\begin{cases} C_p = C_a + (C_{p_0} - C_a)\dfrac{(1-n)}{(1-n_0)}, \\ R_g = R_a + \\ +(R_{g0} - R_a)\left(\dfrac{1-n}{n}\right)\left(\dfrac{n_0}{1-n_0}\right), \\ \theta = E/C_p, \\ \beta = \dfrac{P\sigma}{P(1-n) + nR_g\theta\sigma}, \\ D = \dfrac{\alpha - \beta}{\beta}, \\ r = \sqrt{\dfrac{F}{\beta u}}, \\ C = 2u\varepsilon r\alpha/F. \end{cases} \tag{2.10}$$

Al modelo pliniano compuesto de las ecuaciones (2.7)–(2.10) se le denota como $\mathcal{P}$.

Se puede resumir el modelo pliniano $\mathcal{P} \circ \mathcal{T}$ en el modelo genérico $\mathcal{M}$:

$$(\mathcal{M}) \quad \begin{cases} \dfrac{d\mathrm{u}}{dz} = M(\mathrm{u}, p) \quad en\ (z_0, z_{top}), \\ \mathrm{u}(z_0) = \mathrm{u}_0, \end{cases} \tag{2.11}$$

donde $\mathrm{u} = (r, u, \theta, n)$ es la variable de estado y M es un operador no lineal que describe la dinámica del modelo. Puede depender de parámetros p. El vector $\mathrm{u}_0 = (r_0, u_0, \theta_0, n_0)$ de condiciones de salida pertenece al espacio $\mathcal{U}_0$ de las condiciones

de salida admisibles para formar una columna pliniana.

Este sistema tan general permite modelar también las ecuaciones a las derivadas parciales espacio-temporales (PDE) tales como las ecuaciones de Navier-Stokes usadas en los modelos plinianos de 2 o 3 dimensiones. Así, los desarrollos teóricos (análisis de sensibilidad y modelación inversa) que presentamos en los capítulos 2 y 3 podrían ser aplicados a las ecuaciones de flujo (diferenciables) de cualquier modelo pliniano.

Nota: Se pueden describir de la misma manera los problemas de meteorología, de oceanología, de hidrología (ríos o lagunas), de flujos de lodo o de convección en el manto de la Tierra.

2.3. Método de Runge Kutta de 4o orden

Existen varios métodos numéricos para la solución de problemas de ecuaciones diferenciales ordinarias. Entre ellos, el método de Runge Kutta de 4o órden (RK4) es conocido por su gran estabilidad. Aunque es menos eficiente que otros, nuestro sistema es muy rápido para integrar por lo que la desventaja no es importante. RK4 permite la aproximación de las soluciones de ecuaciones diferenciales ordinarias tal es que:

$$\begin{cases} \dfrac{dy}{dz} = f(z,y), \\ y(z_0) = y_0, \end{cases}$$

donde $z \in IR$. La función matemática f y la variable y pueden ser escalares o vectoriales. Se reconoce bien el sistema general (2.11).

Tomando un paso o incremento δ_z pequeño, RK4 aproxima la solución exacta $y(z_0+(n+1)\delta_z)$ por la solución discreta y^{n+1} calculada a partir del paso anterior y^n por la formula:

$$y^{n+1} = y^n + \frac{\delta z}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

donde:

$$\begin{cases} k_1 = f(z^n, y^n), \\ k_2 = f(z^n + \delta z/2, y^n + k_1 \delta z/2), \\ k_3 = f(z^n + \delta z/2, y^n + k_2 \delta z/2), \\ k_4 = f(z^n + \delta z, y^n + k_3 \delta z). \end{cases}$$

RK4 es de orden 4 por lo que su error de integración es del orden de $(\delta z)^4$.

En nuestro caso, se utilizará con el vector de variables y tal que:

$$y = (y_1, y_2, y_3, y_4) = (F, u, E, n) = \mathcal{T}(\mathrm{u})$$

y la función matemática M tal que:

$$M(z,y) = (M_1, M_2, M_3, M_4)$$

donde:

$$\begin{cases} M_1 = FC, \\ M_2 = gD/u - Cu, \\ M_3 = C(C_\alpha T - E - \dfrac{u^2}{2}) - uM_2 - g, \\ M_4 = (1-n_0)\dfrac{F_0 C}{F}. \end{cases}$$

Como lo hemos visto en (2.6), las variables C y D dependen de y por lo que también tienen que ser evaluadas de manera correcta. La variable z no aparece en las ecuaciones, pero está presente implícitamente en el cálculo de la atmósfera estándar y en el vector de variables y.

2.4. Modelo de atmósfera estándar

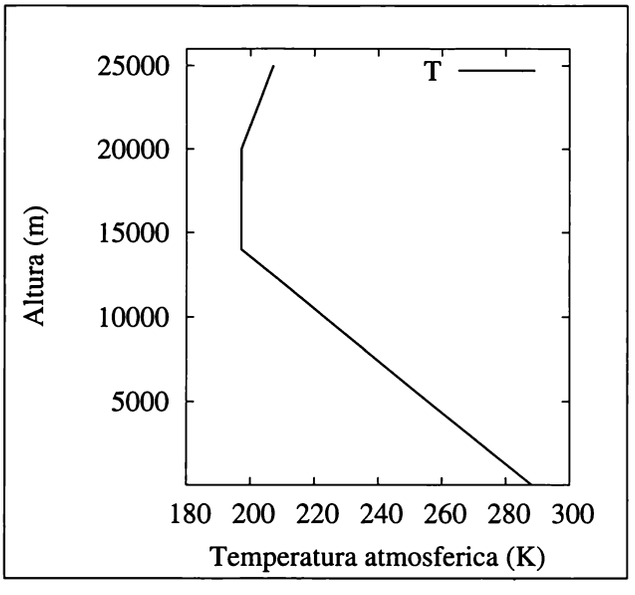

Fig. 2.7 Temperatura atmosférica estándar.

Las fuerzas aerodinámicas dependen directamente de las propiedades de la atmósfera. Así es muy útil definir un modelo de atmósfera estándar

para calcular sus variaciones con respecto a la altura. La atmósfera estándar de los EUA adoptada en 1976 presenta 3 zonas diferentes: troposfera, estratosfera baja y estratosfera alta, suponiendo que las variaciones de temperatura y presión dependen únicamente de la altura. En particular, se considera que:

- la gravedad es constante: $g = 9,81 m.s^{-2}$, y
- la composición del aire no varía. Su masa molecular M_a y su constante R_a son:

$$\begin{aligned} M_a &= 28,9 \cdot 10^{-3} kg.mol^{-1}, \\ R_a &= \frac{R_{gases\_perfectos}}{M_a} = \\ &= 287,54 m^2.s^{-2}.K^{-1}. \end{aligned}$$

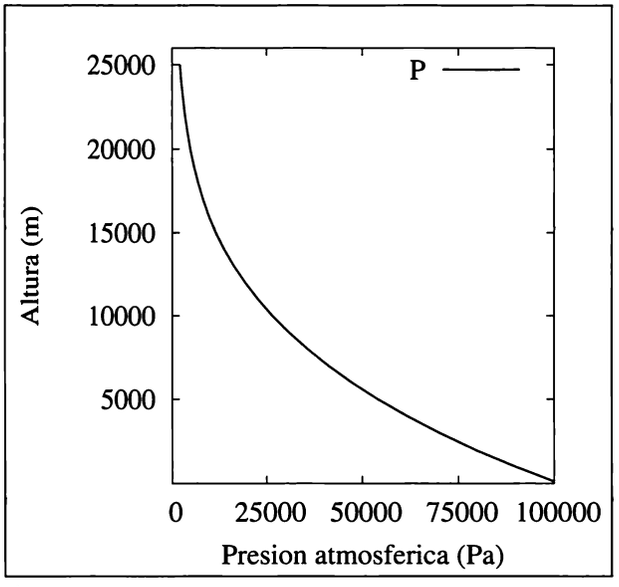

Fig. 2.8 Presión estándar.

Existen varios modelos: día estándar, día caliente, día frío, y día tropical. Sus parámetros están actualizados usando datos atmosféricos recientes. En estos, la presión P (Pa), la temperatura T (K) y la densidad del aire α son calculadas a partir del sistema compuesto por:

- la ecuación hidrostática de una columna de aire:

$$\frac{dP}{dz} = -\alpha g, \tag{2.12}$$

donde z (m) es la altura geopotencial,

- la ley de los gases perfectos aplicada al aire (visto como un gas perfecto):

$$P = \alpha R_a T, \tag{2.13}$$

- una ecuación que describe el comportamiento de la temperatura.

En la troposfera, desde el nivel del mar hasta la tropopausa (de altura $h_1 = 14km$ por ejemplo), la temperatura disminuye linealmente siguiendo:

$$T = T_0 - l_0 z, \tag{2.14}$$

donde $T_0 = 288,15K$ (=15 grados C) es la temperatura al nivel del mar. A la razón del decremento l_0 se le conoce como gradiente térmico o taza de disminución térmica (lapse rate). Aquí, l_0 vale $0,0065 K.m^{-1}$. La solución de (2.12)–(2.14) permite deducir una ecuación de cálculo para la presión:

$$P = P_0 \left(\frac{T}{T_0} \right)^{g/R_a l_0},$$

donde $P_0 = 101300 Pa$ es la presión al nivel del mar. Se nota que la altura h_1 de la tropopausa varia de $8km$ a los polos hasta $18km$ al ecuador.

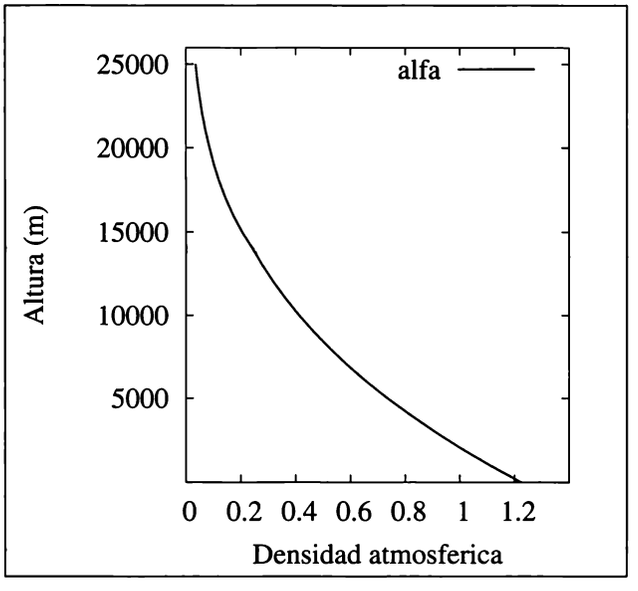

Fig. 2.9 Densidad atmosférica estándar.

En la estratosfera baja, es decir de la tropopausa hasta $h_2 = 20km$, la temperatura permanece constante mientras la presión sigue bajando exponencialmente. Tenemos:

$$\begin{cases} T_1 = T_0 - l_0 h_1 = 216,65, \\ P = P_1 exp\left(\dfrac{g(h_1 - z)}{R T_1} \right), \end{cases}$$

donde P_1 y T_1 son los valores de presión y de temperatura al nivel de la tropopausa.

La estratosfera alta se extiende hasta $h_2 = 25km$. En esta zona, la temperatura aumenta ligeramente de forma lineal, la presión sigue bajando exponencialmente. Se calculan por medio de las fórmulas:

$$\begin{cases} T_2 = 216{,}65 + l_2 z = 216{,}65 + 0{,}002z, \\ P = P_2 \left(\dfrac{T_2}{T_1} \right)^{g/R_a l_2}, \end{cases}$$

donde P_2 y T_2 son los valores de presión y de temperatura a la altura h_2.

En cada zona, la densidad del aire se deriva de la ley de los gases perfectos, es decir:

$$\alpha = p/(R_a T).$$

Las variables atmosféricas estándares están trazadas en las figuras 2.7–2.9 para una tropopausa a 14km.

Al seguir presentamos los códigos del modelo pliniano $\mathcal{P}$. Implementan las ecuaciones (2.7)–(2.10).

2.5. Código original

Se presentan sucesivamente el archivo `README` que describe el directorio `Pliniano`, el archivo `datos.inc` que contiene los parámetros del código, la rutina principal `programa`, y las rutinas `modelo`, `funcion_M` y `atmósfera`. El uso de la rutina `error` está descrito en el capítulo siguiente. Cada rutina figura en su propio archivo. Por ejemplo, la rutina `modelo` está en el archivo `modelo.f`. Las rutinas se pueden descargar desde la pagina Internet `www.lpmm.fr/charpentier/Libro/Pliniano`

```
******   Pliniano/README   ********************************
El directorio Pliniano contiene:
* las rutinas del codigo pliniano original:
  - datos.inc
  - programa.f
  - modelo.f
  - funcion_M.f
  - atmosfera.f
  - error.f
* un Makefile para el compilador gfortran
  - Makefile
  produce el archivo ejecutable pliniano.exe
* un archivo gnuplot para trazar las figuras 2.3 y 2.4
  - curvas_c2.gnu
  produze las figuras postscript del capitulo 2
*************************************************************
```

```
!-------------------------------------------------------
!---   PLineal/datos.inc   -----------------------------
!-------------------------------------------------------
      IMPLICIT NONE
! Temperatura (T0) y presion (P0) al nivel del mar
! Constantes del aire (Ca,Ra)
! Alturas de la tropausa (h1), de la estratosfera baja (h2)
! Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 T0,P0,Ca,Ra,h1,h2,l0,l1,l2,g
      DATA T0,P0,Ca,Ra /288.15,101320.,998.,287.54/
      DATA h1,h2,g /14000.,20000.,9.81/
      DATA l0,l1,l2 /.0065,.0,-.002/
! Altura del crater (z0)
! Datos de la erupcion (Cp0,Rg0,sigma)
! Parametro de arrastre
      REAL*8 Cp0,Rg0,sigma,eps
      DATA Cp0,Rg0,sigma,eps /1617.,462.,1617.,0.09/
      INTEGER z0,dz
      DATA z0,dz /5425, 5/
! Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
      DATA velmin / 10./
! Observaciones
      REAL*8 uobs(6000),thetaobs(6000),robs(6000)
      COMMON/obs/uobs,thetaobs,robs
!-------------------------------------------------------
```

```
!-------------------------------------------------------
!---   Pliniano/programa.f   ---------------------------
!-------------------------------------------------------
      PROGRAM pliniano
! Este programa calcula el radio (r), la velocidad (u),
! la temperatura (theta), la fraccion de gas (n) y la
! densidad (beta) a lo largo de una columna pliniana por
! el modelo de Woods(1988) en la formulacion (2.9)-(2.10).
! Se utiliza 'columna.dat' para trazar la Fig. 2.3
! Se utiliza 'atmosfera.dat' para trazar la Fig. 2.4
      INCLUDE 'datos.inc'
! Condiciones de salida del crater (r0,u0,theta0,n0)
      REAL*8 r0,u0,theta0,n0
      REAL*8 costo
      REAL*8 t,p,alfa
      INTEGER i,zn

! Valores de salida del crater
```

```
      r0=100; u0=300; theta0=1000; n0=0.03

! Apertura del archivo 'columna.dat'
      OPEN(1,file='columna.dat')
      OPEN(2,file='atmosfera.dat')
      REWIND(1)
! Llamada al modelo
      CALL modelo(r0,u0,theta0,n0,costo)

! Apertura del archivo 'atmosfera.dat'
      OPEN(2,file='atmosfera.dat')
      REWIND(2)
      zn = 0.
      DO WHILE (zn.le.25000)
        CALL atmosfera(zn,t,p,alfa)
        WRITE(2,*) zn,t,p,alfa
        zn = zn+1000
      ENDDO
      CLOSE(2)
      END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   Pliniano/modelo.f   -----------------------------------
!-------------------------------------------------------------
      SUBROUTINE modelo (r0,u0,theta0,n0,costo)
! Modelo de Woods modificado:
      INCLUDE 'datos.inc'
! Variables de salida del crater
      REAL*8 r0,u0,theta0,n0 !entradas del modelo
      REAL*8 beta0, E0, F0
! Costo
      REAL*8 costo
! Variables de esta DO
      REAL*8 u,theta,r,n,beta,Cp,Rg,E,F
! Variables intermedias por RK4
      REAL*8 yn(4)  !contiene la solucion del modelo yn
      REAL*8 M1(4),M2(4),M3(4),M4(4) !ecuacion (2.20)
      REAL*8 yy(4)                   !var. intermedias
      INTEGER zz
      REAL*8 dzz
! Variables atmosfericas
      REAL*8 t,p,alfa
      INTEGER i,zn

! Initializaciones
      dzz = dz*1.d0
      costo = 0.d0
      Rg = Rg0
      Cp = Cp0
      CALL atmosfera(z0,t,p,alfa)
      beta0 = 1.d0/(((1.d0-n0)/sigma)+(n0*Rg*theta0/P))
      zn=z0+dz
! Cambio de variables
      F0 = u0*r0*r0*beta0
      E0 = theta0*Cp0
      yn(1) = F0; yn(2) = u0; yn(3) = E0; yn(4) = n0
! Desarollo de la columna
! Solucion por Runge Kutta de orden 4
       DO WHILE (yn(2).GE.velmin)
! Calculos (2.19)
        CALL funcion_M(zn,yn,n0,F0,M1)
        zz = zn + dzz*0.5
         DO i = 1,4
          yy(i) = yn(i) + dzz*M1(i)*0.5d0
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M2)
         DO i = 1,4
          yy(i) = yn(i) + dzz*M2(i)*0.5d0
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M3)
        zz = zn + dz
         DO i = 1,4
          yy(i) = yn(i) + dzz*M3(i)
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M4)
! Ecuacion (2.18)
        DO i = 1,4                          !yn
          yn(i)=yn(i)
     *    +(dzz*(M1(i)+2.d0*M2(i)+2.d0*M3(i)+M4(i))/6.d0)
        ENDDO
```

```
! Cambio de variables inverso
       F = yn(1); u = yn(2); E = yn(3); n = yn(4)
! Relacions funcionales
       Cp=ca+((Cp0-ca)*((1.d0-n)/(1.d0-n0)))
       Rg=Ra+ (Rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
       theta=E/Cp
       beta=1.d0/(((1.d0-n)/sigma)+(n*Rg*theta/p))
       r=SQRT(F/(u*beta))
! evaluacion del error
       CALL error(zn,r,u,theta,costo)
! Escritura del archivo de resultados
       write(1,111) zn,r,u,theta,n,beta
111   FORMAT(I5,10F12.6)
! Incremento de la altura
       zn=zn+dz
     END DO
     CLOSE(1)
     END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   Pliniano/funcion_M.f   ------------------------------
!-------------------------------------------------------------
     SUBROUTINE funcion_M(z,y,n0,F0,M)
! funcion matematica (2.20)
     INCLUDE 'datos.inc'
! Variables atmosfericas,
     REAL*8 T,P,alfa
     INTEGER z
! Variables de RK4
     REAL*8 M(4),y(4)
! Variables intermedias
     REAL*8 C,D,Cp,Rg,theta,r,beta
     REAL*8 F,u,E,n
     REAL*8 n0,F0

! Atmosfera estandar a la altura z
     CALL atmosfera(z,T,P,alfa)
! Cambio de variables
     F = y(1); u = y(2); E = y(3); n = y(4)
! Relaciones funcionales (2.15)
     Cp=ca+((Cp0-ca)*((1.d0-n)/(1.d0-n0)))
     Rg=Ra+ (Rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
     theta=E/Cp
     beta=1.d0/(((1.d0-n)/sigma)+(n*Rg*theta/p))
     r=SQRT(F/(u*beta))
! Variables intermedias
     D = (alfa-beta)/beta
     C = 2.d0*u*eps*r*alfa/F
! Ecuaciones (2.20)
     M(1) = F*C
     M(2) = g*D/u - C*u
     M(3) = c*(Ca*T-E-u*u*.5d0)-u*M(2) - g
     M(4) = (1.d0-n0)*F0*C/F
     END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   Pliniano/atmosfera.f   ------------------------------
!-------------------------------------------------------------
     SUBROUTINE atmosfera(z,T,P,alfa)
! modelo de atmosfera estandar
! los parametros estan definidos en 'datos.inc'
     INCLUDE 'datos.inc'
! Variables de entrada
     INTEGER z         !altura
! Variables de salida
     REAL*8 T,P,alfa   !temperatura, presion y
                       !densidad del aere en z
! Variables intermedias
     REAL*8 T1         !temperatura a la tropospausa (h1)
     REAL*8 P1,P2      !pressures at height h1 y h2
! Temperatura
     IF (z.LE.h1) T = T0  - 10*z       !Ec. (2.24)
     T1  = T0 - 10*h1
     IF (z.GT.h1) T = T1               !Ec. (2.26)
     IF (z.GE.h2) T = T1 - 12*(z-h2)   !Ec. (2.27)
! Presion
     IF (z.LE.h1) P   = P0*(T/T0)**(g/(Ra*10))  !Ec. (2.25)
     P1 = P0*(T1/T0)**(g/(Ra*10))
```

```
      IF (ABS(z-h1).LE.1.E-3) P = P1
                                          !Ec. (2.26)
      IF((z.GT.h1).AND.(z.LT.h2)) P=P1*EXP(g*(h1-z)/(Ra*T1))
      P2 = P1*EXP(g*(h1-h2)/(Ra*T1))
      IF (ABS(z-h2).LE.1.E-3) P = P2
      IF (z.GE.h2) P=P2*(T/T1)**(g/(Ra*l2))      !Ec. (2.27)
! Densidad del aere
      alfa = P/(Ra*T)                     !Ec. (2.28)
      END
!-------------------------------------------------------------


!-------------------------------------------------------------
!---   Pliniano/error.f   ------------------------------
!-------------------------------------------------------------
      SUBROUTINE error(z,r,u,theta,costo)
! Evalua una funcion objetivo
!  * error cuadratica entre observaciones y variables
!  * bajo la tropopausa
!  * cada 500m
      INCLUDE 'datos.inc'
      REAL*8 r,u,theta,costo
      INTEGER z,i

      IF ((MOD(z,500).EQ.0).AND.(z.LE.H1)) THEN
        i = (z-z0)/dz
        costo = costo + (r-robs(i))*(r-robs(i))
     *        + (u-uobs(i))*(u-uobs(i))
     *        + (theta-thetaobs(i))*(theta-thetaobs(i))
      ENDIF
      END
!-------------------------------------------------------------


#-------------------------------------------------------------
#---   Pliniano/Makefile   ------------------------------
#-------------------------------------------------------------
# Makefile para linux: compilador gfortran
#! /bin/sh -f
#
FC= gfortran
LINKER= gfortran
#
FOBJS= programa.o modelo.o funcion_M.o atmosfera.o error.o

.SUFFIXES:.o .f90 .f
.f90.o:
        $(FC) -c $(FFLAGS) $<
.f.o:
        $(FC) -c $(FFLAGS) $<
#
pliniano.exe: $(FOBJS)
        $(LINKER) $(LINKFLAGS) -o pliniano.exe $(FOBJS)
#
clean:
        @rm -f *~ *.o *.mod
#-------------------------------------------------------------
```

3. Análisis de sensibilidad

En las ciencias naturales, y particularmente en la geofísica, para comprender los procesos de la naturaleza se utiliza un sistema equivalente al proceso estudiado pero que permita ser manipulado ya sea en el laboratorio o matemáticamente. En este último caso el modelo intenta capturar en una formulación matemática los principales procesos del fenómeno, quizás más complejo, que se estudia. Sin embargo su validez y su pertenencia desde el punto de vista físico se evalúan por el grado de concordancia de sus resultados con las observaciones. En este sentido, la cuestión que emerge inmediatamente es la siguiente: ¿cuál es la sensibilidad de las variables de ese modelo a un cambio dado en los datos de entrada? Este punto es crucial ya que revela la estabilidad del sistema de ecuaciones y la importancia relativa de cada una de las entradas del modelo. A esta cuestión, se le puede contestar cualitativamente o cuantitativamente. Por un lado, se puede hacer una comparación entre ensayos numéricos sucesivos y las observaciones eligiendo, con cierta arbitrariedad, aquellas entradas "que parecen mejores". Por otro lado, se pueden evaluar los errores de modelación calculando las derivadas de primer ó segundo orden del modelo, como se verá a continuación. En el vocabulario científico, esos cálculos se llaman análisis de sensibilidad.

El análisis de sensibilidad proporciona información cualitativa y cuantitativa sobre el comportamiento del modelo estudiado. En el lenguaje conciso de las matemáticas, la sensibilidad de una función $f(x)$ con respecto a una perturbación δ de su entrada x_0 es la derivada $\frac{\partial f}{\partial x}(x_0).\delta$. Usualmente se aproxima al primer orden como:

$$\frac{\partial f}{\partial x}(x_0) \simeq \frac{f(x_0+\delta)-f(x_0)}{|\delta|}.$$

Su interés es triple ya que el análisis de sensibilidad:

- da una información sobre el comportamiento general del proceso físico mostrando cómo es afectado por un cambio en la variable de entrada x_0. Esta última puede figurar condiciones iniciales, condiciones a la frontera del dominio de cómputo o parámetros,
- permite determinar las incertidumbres en los parámetros del modelo,
- da acceso a las componentes del gradiente involucrado en la solución de problemas inversos (ver el capítulo 3).

Desde el punto de vista numérico, el análisis de sensibilidad se puede llevar a cabo con varios métodos. Entre ellos, se distingue el método de perturbación o desarrollo de Taylor que calcula derivadas aproximadas, de la linealización de las ecuaciones (continuas o discretas) y la diferenciación del código informático que permiten calcular derivadas exactas (a la precisión de la computadora). A este último modo de diferenciación se le llama "modo lineal tangente".

A continuación, presentamos resultados del análisis de sensibilidad para el modelo pliniano conseguidos por diferenciación en modo lineal tangente del código pliniano. Posteriormente mostraremos como conseguir códigos lineales tangentes en general.

3.1. Ejemplo de análisis de sensibilidad

Diferenciamos el código de columnas plinianas del párrafo 2.5 con respecto a las condiciones de salida del cráter, es decir el radio (r_0), la velocidad (u_0), la temperatura (θ_0) y la fracción de gas (n_0). Los resultados se presentan en la figura 3.1 donde se observan en particular la velocidad y sus sensibilidades con perturbaciones de 10 % de las condiciones de salida. Los códigos que permiten conseguir estos resultados y el archivo `curvas_c3.gnu` que les permite trazar usando `gnuplot` (en `linux`)

se pueden descargar desde la pagina Internet `www.lpmm.fr/charpentier/Libro/PLineal`

Cada sensibilidad se interpreta como la diferencia (linealizada) entre la simulación original y una simulación conseguida agregando una perturbación de 10 % sobre una de las condiciones de salida (Charpentier y Espíndola, 2005a; Charpentier y Espíndola, 2005b). En el caso del estudio de la velocidad, las sensibilidades muestran que:

1. la sensibilidad de la velocidad con respecto a n_0 es negativa. Esto significa que la velocidad disminuye cuando n_0 se incrementa. Un decremento en la velocidad significa una altura menor de la columna entonces una fracción de gas más alta significa una altitud más baja de la columna.

   Esta relación también podría haber sido deducida haciendo comparaciones de curvas de velocidad calculadas con diferentes valores de n_0 (ver Woods (1998),fig.3b). Una de las ventajas del análisis de sensibilidad por modelación lineal tangente es que muestra los resultados directamente.

2. la temperatura θ_0 actúa también de manera negativa sobre la velocidad en la columna en la zona de empuje. De hecho, mientras más alta es la temperatura de salida, más baja es la densidad en la columna lo que disminuye la altura de la zona de empuje. Sin embargo, se nota que el efecto de la temperatura de salida sobre la velocidad es positivo en la zona de flotación porque se calienta más el aire atrapado por la columna.

3. la velocidad reacciona positivamente a cambios positivos del radio y de la velocidad de salida. Esta relación es desde luego la esperada, ya que es claro que un aumento del radio incrementa la inercia de la mezcla en la columna y por lo tanto su velocidad. Aún más claro es el efecto de la velocidad inicial, ya que se esperaría que las velocidades a lo largo de la columna sean mayores si la velocidad de salida es mayor. Nótese sin embargo que el análisis de sensibilidad nos revela que la velocidad es más sensible a una perturbación de u_0 en la zona de empuje, y es casi constante en la zona de flotación.

Como puede desprenderse de lo anterior, básicamente derivado de las teorías de la propulsión a chorro y de las plumas térmicas, las sensibilidades son grandes tanto bajo la tropopausa, lo que indica la importancia de los dispositivos de detección terrestres, como sobre la tropopausa, lo que muestra también la importancia de los dispositivos de observación espaciales. En particular, se observan sensibilidades fuertes del radio, de la temperatura y de la velocidad con respecto a cambios en las condiciones de salida. Esto significa que la observación de estas 3 variables a lo largo de la columna puede proporcionar datos relevantes (por su contenido de información) para el proceso de solución del problema inverso. Por supuesto la relevancia de estos datos depende fuertemente del desempeño de los dispositivos de medida (disponibilidad, precisión, resoluciones espaciales y temporales). Por el contrario, los efectos de los cambios en la fracción de gas y la densidad son mucho más pequeños (nótese el factor 100 que usamos para poder hacer comparaciones con otras curvas). En particular, son casi insensibles a un cambio de la velocidad de salida. Esto significa que la velocidad al nivel del cráter no puede ser determinada con datos de fracción de gas o de densidad.

Nota: Los procesos volcánicos son fuertemente no lineales y que el comportamiento del modelo no se puede predecir por una linealización simple. Cerca del punto de transición, un cambio pequeño en alguno parámetro podría causar un cambio importante en el comportamiento de la columna: la columna convectiva puede volverse inestable y colapsar formando flujos piroclásticos.

Trazar influencias de una condición de salida sobre las variables del modelo (Fig. 3.2) es otra manera de explotar los mismos resultados.

3.2. Método de perturbación

Desde hace 200 años, el desarrollo en serie de Taylor representa la función $f : IR \to IR$, $x \mapsto f(x)$, supuestamente diferenciable al infinito, en el punto x_0 por:

$$f(x_0 + \delta_{x_0}) = f(x_0) + \sum_{k=1,\infty} \frac{\delta_{x_0}^k}{k!} \frac{\partial^k f}{\partial x^k}(x_0),$$

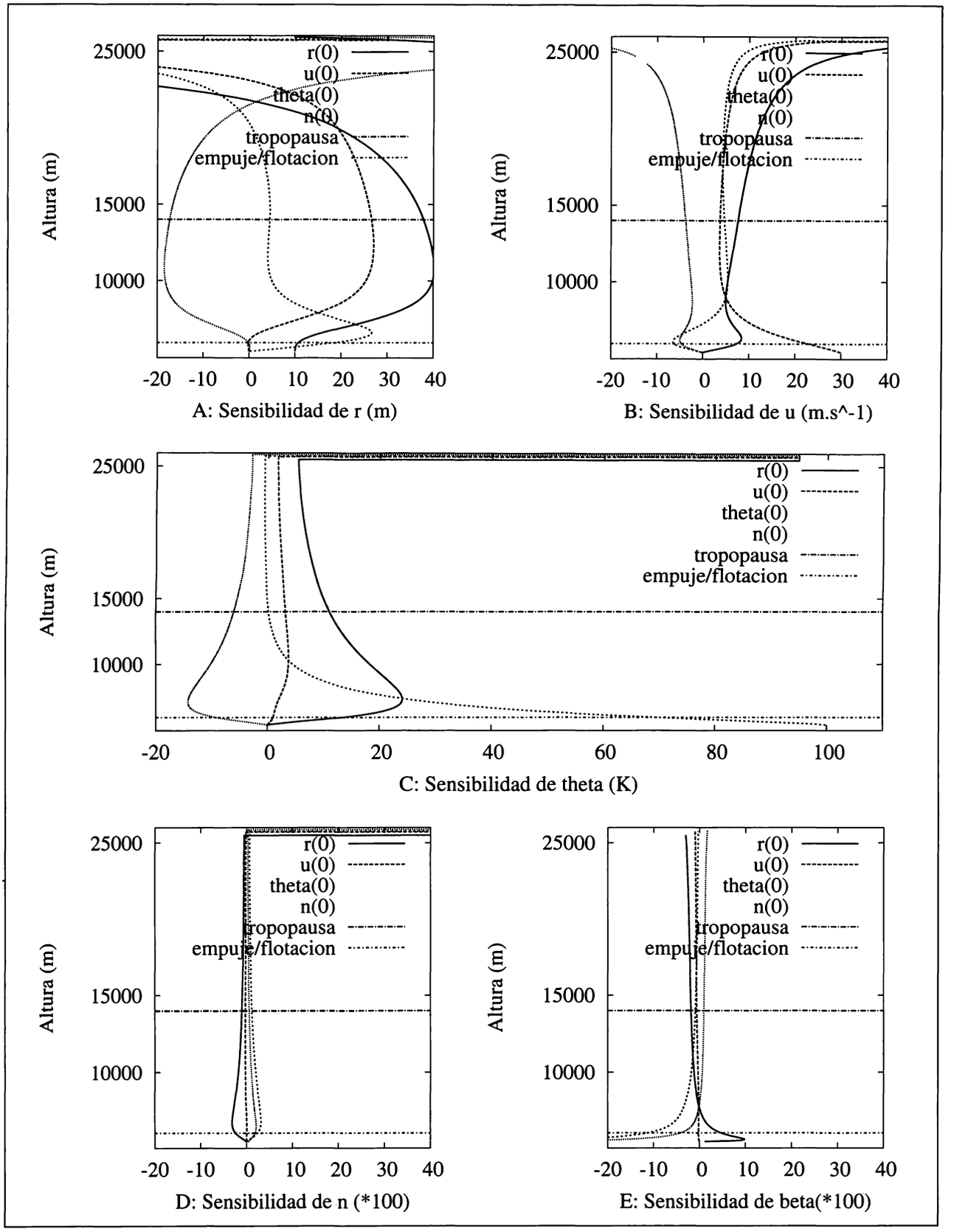

Fig. 3.1 A: Sensibilidades del radio (en m) con respecto a perturbaciones de 10 % de las condiciones de salida. Sensibilidades de la velocidad (B, en $m.s^{-1}$), de la temperatura (C, en grados K), de la fracción de gas (D), de la densidad (E).

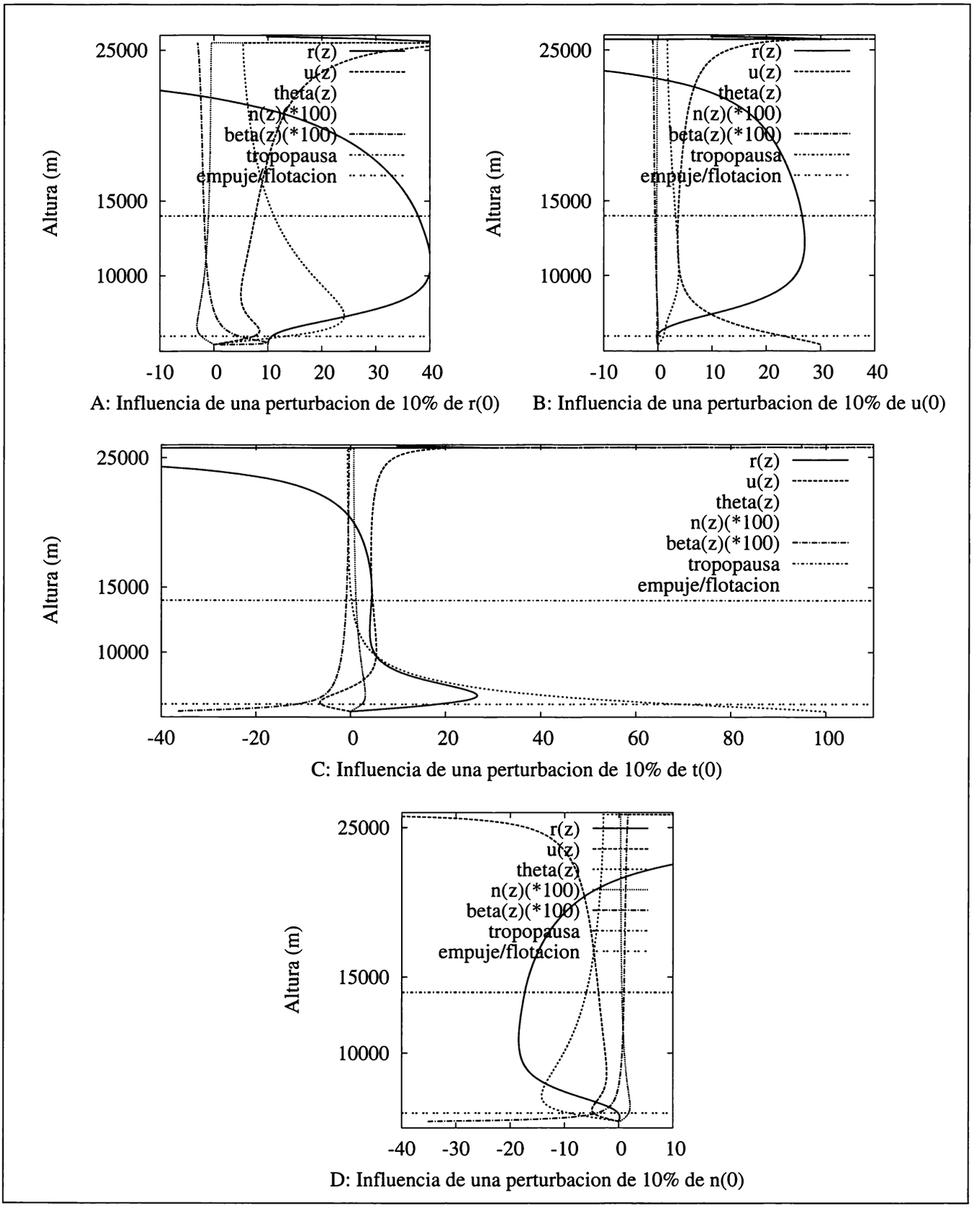

Fig. 3.2 A: Influencia sobre las variables de un cambio de 10 % del radio de salida r_0 (en unidades correspondientes). Influencia de un cambio de 10 % de la velocidad u_0 (B), de la temperatura θ_0 (C), de la fracción de gas n_0 (D).

donde δ_{x_0} es un real pequeño. Se puede truncar así:

$$f(x_0+\delta_{x_0}) = f(x_0)+\delta_{x_0}\frac{\partial f}{\partial x}(x_0)+O|\delta_{x_0}|^2, \quad (3.1)$$

donde el término $O|\delta_{x_0}|^2$ significa que la parte despreciada es del mismo orden que $|\delta_{x_0}|^2$. La fórmula (3.1) permite calcular la aproximación:

$$\frac{\partial f}{\partial x}(x_0) = \frac{f(x_0+\delta_{x_0})-f(x_0)}{\delta_{x_0}} + O|\delta_{x_0}|$$

de la derivada de f en x_0 con la precisión δ_{x_0}. Este método de aproximación es del tipo de diferencias finitas. Puede ser usado para estudiar la sensibilidad de un modelo con respecto a sus entradas.

Aplicando la aproximación por serie de Taylor a nuestro modelo general (2.11), se aproximan las sensibilidades a lo largo de la columna calculando sus derivadas parciales con respecto a las condiciones de salida:

$$\frac{\partial \mathcal{M}}{\partial \mathrm{u}_0}(\mathrm{u}_0,p_0) \simeq \frac{\mathcal{M}(\mathrm{u}_0+\delta_{\mathrm{u}_0},p_0)-\mathcal{M}(\mathrm{u}_0,p_0)}{\delta_{\mathrm{u}_0}},$$

y los parámetros:

$$\frac{\partial \mathcal{M}}{\partial p_0}(\mathrm{u}_0,p_0) \simeq \frac{\mathcal{M}(\mathrm{u}_0,p_0+\delta_{p_0})-\mathcal{M}(\mathrm{u}_0,p_0)}{\delta_{p_0}},$$

donde δ_{u_0} y δ_{p_0} son ahora direcciones de perturbación de norma pequeña.

Este método de "diferenciación" resulta barato en términos de desarrollo porque sólo se evalúa el modelo $\mathcal{M}$ en puntos diferentes. Pero, la aproximación trae un error dependiente:

- del tamaño de la norma $|\delta|$ del vector de perturbación $(\delta_{\mathrm{u}_0}, \delta_{p_0})$ (es el error de truncamiento predicho por la teoría), y
- de la precisión de la computadora (es el error de redondeo que sobreviene por la substracción de dos números muy cercanos).

Cuando $|\delta|$ disminuye, el error de truncamiento disminuye también; pero, pasos $|\delta|$ demasiado pequeños generan un error de redondeo que domina el error de truncamiento. Así, existe un paso óptimo que minimiza la suma de estos 2 errores. El problema es que este paso óptimo depende del punto de evaluación de las derivadas. Obviamente, la aproximación de Taylor no es bastante precisa para la solución de problemas inversos por un método de gradiente.

Su uso es todavía muy frecuente para cálculos de sensibilidades. Como veremos posteriormente en este capítulo (ver el párrafo 3.4.2), se usa también para comprobar las derivadas producidas por la diferenciación de los códigos.

3.3. Linealización de las ecuaciones

Los dos métodos para conseguir códigos lineales son:

- derivar las ecuaciones matemáticas continuas (o discretas) como se presenta en este párrafo y luego implementarlas, o
- escribir el código relativo a las ecuaciones discretas, y diferenciarlo como se verá en el párrafo 3.4.

Las bases del cálculo diferencial son sencillas. De hecho, el conocimiento de las reglas de diferenciación de los operadores, de las funciones elementales, y de la ley de composición de las funciones son suficientes para calcular derivadas de funciones compuestas complejas. A continuación presentamos sucesivamente tres opciones que permiten conseguir derivadas numéricas. Todas están basadas en las reglas clásicas de diferenciación, es decir reglas que el usuario hubiera usado para diferenciar a mano. A este modo de diferenciación se le dice "modo lineal tangente" o "modo directo". No hablaremos mucho de herramientas estilo `Maple` ya que no tratan códigos completos.

3.3.1. Diferenciación del modelo genérico $\mathcal{M}$

La diferenciación lineal tangente de las ecuaciones continuas del modelo $\mathcal{M}$ con respecto a su entrada u_0 se hace por medio de la ley de derivación de las funciones compuestas. En el vocabulario de la diferenciación automática, la variable u_0 se designa como:

- "activa" (para la diferenciación) porque diferenciamos ecuaciones con respecto a ella, e

- ecuaciones diferenciales ordinarias (2.7)–(2.8):

$$\begin{cases} \dfrac{d\delta F}{dz} = \delta F C + F \delta C, \\ \dfrac{d\delta u}{dz} = g(\delta D u - D \delta u)/u^2 - \delta C u - C \delta u, \\ \dfrac{d\delta E}{dz} = \delta C(C_\alpha T - E + u^2/2) - C\delta E + \\ \qquad + C u \delta u / 2 \delta D - \delta D, \\ \dfrac{d\delta n}{dz} = (1 - n_0)\dfrac{\delta F_0 C F + F_0 \delta C F - F_0 C \delta F}{F^2} - \\ \qquad - \delta n_0 \dfrac{F_0 C}{F}, \end{cases}$$

- cambio de variable inverso $\mathcal{T}^{-1}$. Se actualizan las derivadas $\delta\theta$ y δr usando (3.4).

Aunque se han utilizado reglas sencillas, la diferenciación de las ecuaciones continuas es una tarea tediosa que puede ser muy cansada para sistemas de ecuaciones más complejos.

3.4. Diferenciación automática por sobrecarga de los operadores

Algunos softwares de cálculo formal tal como Maple pueden llevar a cabo la diferenciación de ecuaciones matemáticas. El usuario ingresa su función en la forma pedida por el software, y pide la diferenciación de ésta. El resultado es una ecuación derivada en el formalismo del software. Sin embargo, no le es posible derivar códigos. Por otra parte, evaluar derivadas a lo largo de un programa completo requiere no solamente derivar las instrucciones matemáticas (que representan ecuaciones matemáticas), sino también crear el ambiente informático que permite evaluar las derivadas de manera correcta.

La diferenciación automática de programa (Griewank, 2000)y sus herramientas dan una repuesta confiable a la necesidad de evaluar derivadas a lo largo de programas completos. Existen dos tipos de herramientas. Por un lado, se puede construir una biblioteca basada en la técnica de sobrecarga de los operadores bien conocida de los usuarios que desarrollan programas en `Fortran 90` (y posteriores) o `C++`. Por otro lado, las herramientas de diferenciación de fuente a fuente (ver Sec. 3.5) son capaces de generar las instrucciones derivadas de un código usando conceptos de cálculo formal. Como lo veremos en el capítulo 4, éstas involucran frecuentemente el modo adjunto para el cálculo de gradientes. Una revisión completa de estas herramientas se presenta en el sitio Internet `www.autodiff.org`

A continuación presentamos con detalle la sorbrecarga de operadores.

3.4.1. Biblioteca de sobrecarga de operadores

Esta técnica consiste en la definición de un nuevo tipo de variable y de los operadores y funciones matemáticas asociados. Todos esos se agregan para formar una biblioteca de diferenciación. Una vez construida, se compila con el código que se quiere diferenciar. Como veremos en el ejemplo, muy pocos cambios del código original son necesarios para calcular derivadas. Aquí, consideramos programas escritos en `Fortran 90`.

Nótese que sólo las variables de tipo `REAL` o `COMPLEX` (cualquiera que sea la precisión) pueden ser diferenciadas. De hecho, no hay razón para diferenciar una variable entera ya que su valor más una pequeña perturbación no resulta en un entero. Si tal fuera el caso en un código, por ejemplo en la definición de un parámetro, y se quisiera calcular sensibilidades con respecto a este parámetro, se tendría que cambiar su tipo por `REAL`.

Al principio de toda sobrecarga figura un nuevo tipo. En nuestro caso eligemos el tipo `dreal8`.

```
!-----------------------------------------
!---   PLineal/mtipo.f90              ------
!-----------------------------------------
MODULE mtipo
  IMPLICIT NONE
  TYPE dreal8
!d0: campo dedicado al valor de la variable
!d1: campo dedicado al valor de la derivada
  REAL*8 :: d0, d1
  END TYPE dreal8
END MODULE mtipo
!-----------------------------------------
```

Una variable `x` de tipo `dreal8` tiene sus campos

x%d0 y x%d1 dedicados al valor de la variable y su derivada. Las variables activas de tipo REAL*8 de nuestro código pliniano se cambian al tipo dreal8 escrito en el MODULE mtipo para la propagación de la diferenciación.

Al tipo dreal8, se le asocia una definición de los operadores usuales {+, -, *, /} y otros cuando son necesarios. Cada de esos operadores tiene dos operandos (a excepción del − unitario). Estos pueden ser:

- ambos de tipo dreal8 si las dos variables están activas,
- de tipo dreal8 y REAL*8 si sólo una está activa. En este caso, la biblioteca tendrá que manejar las parejas dreal8/REAL*8 y REAL*8/dreal8.

A consecuencia, cada operador necesita 3 nuevas definiciones para cubrir estas parejas (el − unitario se sobrecarga también). En Fortran, a cada operador se le asocia un módulo. Para el caso de la multiplicación se escribe el MODULE mmultiplicacion. Se observan las reglas usuales de diferenciación del operador de multiplicación en las funciones del módulo. Basta cambiar a%d0 por x, a%d1 por x', b%d0 por y, b%d1 por y', res%d0 por z, res%d1 por z', para encontrar que $z = xy$ se diferencia por $z' = xy' + y'x$.

```
!---------------------------------------
!---   PLineal/mmultiplicacion.f90   ---
!---------------------------------------
MODULE mmultiplicacion
  USE mtipo
  INTERFACE OPERATOR ( * )
    MODULE PROCEDURE multi_dr_dr, &
                     multi_dr_r,  &
                     multi_r_dr
  END INTERFACE
CONTAINS
  FUNCTION multi_dr_dr(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a,b
    TYPE(dreal8)             :: r
    r%d0 = a%d0*b%d0
    r%d1 = a%d0*b%d1 + a%d1*b%d0
  END FUNCTION multi_dr_dr

  FUNCTION multi_dr_r(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a
    REAL*8      , INTENT(IN) :: b
    TYPE(dreal8)             :: r
    r%d0 = a%d0*b
    r%d1 = a%d1*b
  END FUNCTION multi_dr_r

  FUNCTION multi_r_dr(a,b)  RESULT (r)
    REAL*8      , INTENT(IN) :: a
    TYPE(dreal8), INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0 = a*b%d0
    r%d1 = a*b%d1
  END FUNCTION multi_r_dr
END MODULE mmultiplicacion
!---------------------------------------
```

Las instrucciones que permiten diferenciar la división son:

```
!---------------------------------------
!---   PLineal/mdivision.f90         -----
!---------------------------------------
MODULE mdivision
  USE mtipo
  INTERFACE OPERATOR ( / )
    MODULE PROCEDURE division_dr_dr, &
                     division_dr_r,  &
                     division_r_dr
  END INTERFACE
CONTAINS
  FUNCTION division_dr_dr(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a,b
    TYPE(dreal8) :: r
    r%d0=a%d0/b%d0
    r%d1=(a%d1-r%d0*b%d1)/b%d0
  END FUNCTION division_dr_dr

  FUNCTION division_dr_r(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a
    REAL*8      , INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0=a%d0/b
    r%d1=a%d1/b
  END FUNCTION division_dr_r

  FUNCTION division_r_dr(a,b)  RESULT (r)
    REAL*8      , INTENT(IN) :: a
    TYPE(dreal8), INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0=a/b%d0
    r%d1=a/b%d0*(-r%d0*b%d1)
  END FUNCTION division_r_dr
END MODULE mdivision
!---------------------------------------
```

Las instrucciones que permiten diferenciar la suma son:

```
!---------------------------------------
!---   PLineal/msuma.f90             ---
!---------------------------------------
MODULE msuma
  USE mtipo
  INTERFACE OPERATOR ( + )
    MODULE PROCEDURE suma_dr_dr, &
                     suma_dr_r,  &
                     suma_r_dr
  END INTERFACE
CONTAINS
```

```
  FUNCTION suma_dr_dr(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a,b
    TYPE(dreal8) :: r
    r%d0 = a%d0+b%d0
    r%d1 = a%d1+b%d1
  END FUNCTION suma_dr_dr

  FUNCTION suma_dr_r(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a
    REAL*8      , INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0 = a%d0+b
    r%d1 = a%d1
  END FUNCTION suma_dr_r

  FUNCTION suma_r_dr(a,b)  RESULT (r)
    REAL*8      , INTENT(IN) :: a
    TYPE(dreal8), INTENT(IN) :: b
    TYPE(dreal8):: r
    r%d0 = a+b%d0
    r%d1 =   b%d1
  END FUNCTION suma_r_dr
END MODULE msuma
!--------------------------------------
```

Las instrucciones que permiten diferenciar la resta son:

```
!--------------------------------------
!---   PLineal/mresta.f90             ---
!--------------------------------------
MODULE mresta
  USE mtipo
  INTERFACE OPERATOR ( - )
    MODULE PROCEDURE resta_dr,   &
                     resta_dr_dr,&
                     resta_dr_r, &
                     resta_r_dr
  END INTERFACE
CONTAINS
  FUNCTION resta_dr_dr(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a,b
    TYPE(dreal8) :: r
    r%d0 = a%d0-b%d0
    r%d1 = a%d1-b%d1
  END FUNCTION resta_dr_dr

  FUNCTION resta_dr_r(a,b)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a
    REAL*8      , INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0 = a%d0-b
    r%d1 = a%d1
  END FUNCTION resta_dr_r

  FUNCTION resta_r_dr(a,b)  RESULT (r)
    REAL*8      , INTENT(IN) :: a
    TYPE(dreal8), INTENT(IN) :: b
    TYPE(dreal8) :: r
    r%d0 = a-b%d0
    r%d1 =  -b%d1
  END FUNCTION resta_r_dr

  FUNCTION resta_dr(a)  RESULT (r)
! - unitario
    TYPE(dreal8), INTENT(IN) :: a
    TYPE(dreal8) :: r
    r%d0 = -a%d0
    r%d1 = -a%d1
  END FUNCTION resta_dr
END MODULE mresta
!--------------------------------------
```

A estos módulos se agregan otros para definir todos los operadores dando cuenta de todas las parejas de tipos posibles. Por ejemplo, se tiene que considerar las parejas dreal8–INTEGER. Para ser completo se añaden módulos para las funciones matemáticas elementales como la raíz cuadrada o la potencia con exponente "2".

```
!--------------------------------------
!---   PLineal/msqrt.f90              ---
!--------------------------------------
MODULE msqrt
  USE mtipo
  INTERFACE SQRT
    MODULE PROCEDURE sqrt_dr
  END INTERFACE
CONTAINS
  FUNCTION sqrt_dr(a)  RESULT (r)
  TYPE(dreal8), INTENT(IN) :: a
  TYPE(dreal8) :: r
  r%d0=sqrt(a%d0)
  r%d1=a%d1/(2*r%d0)
  END FUNCTION sqrt_dr
END MODULE msqrt
!--------------------------------------
```

```
!--------------------------------------
!---   PLineal/mpotencia2.f90         ---
!--------------------------------------
! funciona solo para a**2
MODULE mpotencia2
  USE mtipo
  INTERFACE OPERATOR ( ** )
    MODULE PROCEDURE poten_dr
  END INTERFACE
CONTAINS
  FUNCTION poten_dr(a,i)  RESULT (r)
    TYPE(dreal8), INTENT(IN) :: a
    INTEGER     , INTENT(IN) :: i
    TYPE(dreal8) :: r
    r%d0 = a%d0*a%d0
    r%d1 = 2*a%d0*a%d1
  END FUNCTION poten_dr
END
!--------------------------------------
```

Se pueden requerir también módulos dedicados a las operaciones lógicas que sirven en las pruebas condicionales de un código. En el modelo pliniano se necesita el módulo mge dedicado a la

operación lógica .GE. que determina si el valor de la velocidad es todavía mayor que el criterio establecido para detener el proceso, por ejemplo $10m.s^{-1}$.

```
!---------------------------------------
!---   PLineal/mge.f90                 ---
!---------------------------------------
MODULE mge
  USE mtipo
  IMPLICIT NONE
  INTERFACE OPERATOR ( .GE. )
    MODULE PROCEDURE ge_dr_r
  END INTERFACE
CONTAINS
  FUNCTION ge_dr_r(a,b)  RESULT (r)
  TYPE(dreal8), INTENT(IN) :: a
  REAL*8      , INTENT(IN) :: b
  LOGICAL                  :: r
  r = a%d0.ge.b
  END FUNCTION ge_dr_r
END MODULE mge
!---------------------------------------
```

Finalmente se considera el módulo para la asignación (=) que tiene un rol especial. Se utiliza para inicializar los campos de la variables de tipo dreal8 con valores reales o enteras. En nuestro caso, sirve para la instrucción Cp=Cp0 porque Cp0, al contrario de Cp, no es activa. Se evita introducirlo al principio porque su ausencia permite encontrar las variables activas dependientes examinando los errores de compilación. Una práctica más simple consiste en cambiar todas las variables reales al tipo dreal8.

```
!---------------------------------------
!---   PLineal/migual.f90              ---
!---------------------------------------
MODULE migual
  USE mtipo
  INTERFACE ASSIGNMENT ( = )
    MODULE PROCEDURE igual_r
  END INTERFACE
CONTAINS
  SUBROUTINE igual_r(r,a)
    REAL*8,       INTENT(IN) :: a
    TYPE(dreal8), INTENT(OUT) :: r
    r%d0 = a
    r%d1 = 0.
  END SUBROUTINE igual_r
END MODULE migual
!---------------------------------------
```

Los tipos nuevos junto con los módulos de sobrecarga de operadores y las funciones forman la biblioteca de diferenciación. Tiene que ser completa para poder aplicarse a cualquier código (ejercicio). El trabajo de escritura puede parecer ligeramente largo, pero se lleva a cabo sólo una vez. Una vez validada, la biblioteca de sobrecarga puede usarse confiablemente. En lo sucesivo, el usuario llama a la biblioteca en su programa, reemplazando el tipo de las variables activas (diferenciadas en el proceso) por el nuevo tipo: la diferenciación se hace al nivel del compilador que sobrecarga los operadores. Quienes se interesan por una biblioteca completa se le aconseja Adol-C (Griewank et al., 1996)para códigos en C y C++. Se puede utilizar Rapsodia (Charpentier y Utke, to appear) para códigos en F90.

Es muy sencillo sobrecargar las rutinas del modelo pliniano porque éstas se consiguen:

- haciendo una copia de las rutinas originales,
- añadiendo la instrucción:

  INCLUDE 'modulos.inc'

  a cada rutina por derivar,
- cambiando el tipo de las variables activas,
- (no se modifican las instrucciones de cómputo).

Estos pequeños cambios bastan para que el compilador sobrecargue los operadores.

```
!---------------------------------------
!---   PLineal/modulos.inc             ---
!---------------------------------------
! Paquetes para el typo derivado dreal8
! y la sobrecarga de los operadores
      USE mtipo
      USE msuma
      USE mresta
      USE mmultiplicacion
      USE mdivision
      USE msqrt
      USE mge
      USE migual
!---------------------------------------
```

Se presentan sucesivamente el archivo README que describe el directorio Internet www.lpmm.fr/charpentier/Libro/PLineal, la rutina principal programa_lin, y las rutinas modelo_lin, funcion_M_lin y error_lin (párrafo 3.4.2). Cada una figura en su propio archivo. Como no hay retroalimentación de la columna sobre la atmósfera en el modelo, las variables de atmósfera no están activas para la diferenciación y el archivo atmosfera.f no necesita modificaciones. Se completa por el Makefile.

```
******   PLineal/README   ********************************
El directorio PLineal contiene:
* los archivos de la biblioteca de sobrecarga de los
  operadores
  - mtipo.f90,msuma.f90,mmultiplicacion.f90,mdivision.f90,
  - mresta.f90,migual.f90,mge.f90,msqrt.f90,mpotencia2.f90
  - modulos.inc
* las rutinas del codigo pliniano linealizado
  - programa_lin.f,modelo_lin.f,funcion_M_lin.f,error_lin.f
* las rutinas del codigo original
  - datos.inc
  - atmosfera.f
* un Makefile para el compilador gfortran
  - Makefile
  produce el archivo ejecutable plineal.exe
* un archivo de observaciones
  - observaciones
* un archivo gnuplot para trazar las figuras 3.1 y 3.2
  - curvas_c3.gnu
  produze las figuras postscript del capitulo 3
************************************************************


!-------------------------------------------------------
!---   PLineal/programa_lin.f  ---------------------------
!-------------------------------------------------------
      PROGRAM plineal
! Este programa calcula el radio (r), la velocidad (u),
! la temperatura (theta), la fraccion de gas (n) y la
! densidad (beta) a lo largo de una columna pliniana por
! el modelo de Woods(1988) en la formulacion (2.9)-(2.10).
! Calcula las sensibilidades por sobrecarga
      INCLUDE 'modulos.inc'
      INCLUDE 'datos.inc'
! Condiciones de salida del crater (r0,u0,theta0,n0)
      TYPE(dreal8) r0,u0,theta0,n0
! Condiciones de salida perturbadas
      TYPE(dreal8) r0p,u0p,theta0p,n0p
      TYPE(dreal8) costo
      REAL*8 omega       !paso de perturbacion para Taylor
      TYPE(dreal8) costo_u,costo_up !para Taylor
      INTEGER i,z
!----
! Prueba de Taylor
! Lectura de observaciones (calculadas para u0=330m/s)
      OPEN(1,FILE='observaciones')
      REWIND(1)
      DO i = 1,2000
        READ(1,*) z,robs(i),uobs(i), thetaobs(i)
      ENDDO
      CLOSE(1)
! Valores de salida del crater
      r0%d0=100; u0%d0=300;
      theta0%d0=1000; n0%d0=0.03
! Eleccion de la direccion de perturbacion
      r0%d1=0; u0%d1=1; theta0%d1=0; n0%d1=0
! Llamada al modelo lineal tangente
      CALL modelo(r0,u0,theta0,n0,costo_u)
! Prueba de Taylor
      omega = 10.
      DO i = 1,10
        omega = omega/10
        r0p%d0     = r0%d0     + omega*r0%d1
        u0p%d0     = u0%d0     + omega*u0%d1
        theta0p%d0 = theta0%d0 + omega*theta0%d1
        n0p%d0     = n0%d0     + omega*n0%d1
        CALL modelo(r0p,u0p,theta0p,n0p,costo_up)
! Escritura de los resultados como tabla Latex
        WRITE(*,*) '10$^{-',i,'}$ &',
     *(costo_up%d0-costo_u%d0)/(omega*costo_u%d1),'\\\\'
      ENDDO
!----
!Calculo de las sensibilidades
! Valores de salida
      r0%d0=100; u0%d0=330; theta0%d0=1000; n0%d0 = 0.03
! Inicialisacion del error y de la componente del gradiente
      costo%d0=0.; costo%d1=0.  !no sirve en este programa

! Direccion de perturbacion (10\% de r0)
      r0%d1=10; u0%d1=0; theta0%d1=0; n0%d1=0
! Apertura de la carpeta para escribir las variables
! a lo largo de la columna
      OPEN(1,file='sensr')
! Llamada al modelo lineal
      CALL modelo(r0,u0,theta0,n0,costo)

! Direccion de perturbacion (10\% de u0)
      r0%d1=0; u0%d1=30; theta0%d1=0; n0%d1=0
      OPEN(1,file='sensu')
```

```
! Llamada al modelo lineal
      CALL modelo(r0,u0,theta0,n0,costo)

! Direccion de perturbacion (10\% de theta0)
      r0%d1=0; u0%d1=0; theta0%d1=100; n0%d1=0
      OPEN(1,file='senst')
! Llamada al modelo lineal
      CALL modelo(r0,u0,theta0,n0,costo)

! Direccion de perturbacion (10\% de n0)
      r0%d1=0; u0%d1=0; theta0%d1=0; n0%d1=0.003
      OPEN(1,file='sensn')
! Llamada al modelo lineal
      CALL modelo(r0,u0,theta0,n0,costo)
      END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   PLineal/modelo_lin.f   ------------------------------
!-------------------------------------------------------------
      SUBROUTINE modelo (r0,u0,theta0,n0,costo)
! Modelo de Woods modificado:
      INCLUDE 'modulos.inc'
      INCLUDE 'datos.inc'
! Variables de salida del crater
      TYPE(dreal8) r0,u0,theta0,n0 !entradas del modelo
      TYPE(dreal8) beta, E0, F0
! Costo
      TYPE(dreal8) costo
! Variables de estado
      TYPE(dreal8) r,u,theta,n,beta,Cp,Rg,E,F
! Variables intermedias por RK4
      TYPE(dreal8) yn(4)  !solucion del modelo yn
      TYPE(dreal8) M1(4),M2(4),M3(4),M4(4) !ecuacion (2.20)
      TYPE(dreal8) yy(4)  !var. intermedias
      INTEGER zz
      REAL*8 dzz
! Variables atmosfericas
      REAL*8 t,p,alfa
      INTEGER i,zn

! Initializaciones
      dzz = dz*1.d0
      costo = 0.d0
      Rg = Rg0
      Cp = Cp0
      CALL atmosfera(z0,t,p,alfa)
      beta0 = 1.d0/(((1.d0-n0)/sigma)+(n0*Rg*theta0/P))
      zn=z0+dz
! Cambio de variables
      F0 = u0*r0*r0*beta0
      E0 = theta0*Cp0
      yn(1) = F0; yn(2) = u0; yn(3) = E0; yn(4) = n0
! Desarollo de la columna
! Solucion por Runge Kutta de orden 4
       DO WHILE (yn(2).GE.velmin)
! Calculos (2.19)
        CALL funcion_M(zn,yn,n0,F0,M1)
        zz = zn + dzz*0.5
         DO i = 1,4
          yy(i) = yn(i) + dzz*M1(i)*0.5d0
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M2)
         DO i = 1,4
          yy(i) = yn(i) + dzz*M2(i)*0.5d0
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M3)
        zz = zn + dz
         DO i = 1,4
          yy(i) = yn(i) + dzz*M3(i)
        ENDDO
        CALL funcion_M(zn,yy,n0,F0,M4)
! Ecuacion (2.18)
        DO i = 1,4                          !yn
          yn(i)=yn(i)
     *    +(dzz*(M1(i)+2.d0*M2(i)+2.d0*M3(i)+M4(i))/6.d0)
        ENDDO
! Cambio de variables inverso
        F = yn(1); u = yn(2); E = yn(3); n = yn(4)
! Relacions funcionales
        Cp=ca+((Cp0-ca)*((1.d0-n)/(1.d0-n0)))
        Rg=Ra+ (Rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
        theta=E/Cp
        beta=1.d0/(((1.d0-n)/sigma)+(n*Rg*theta/p))
        r=SQRT(F/(u*beta))
! evaluacion del error
        CALL error(zn,r,u,theta,costo)
! Escritura del archivo de resultados
        write(1,111) zn,r,u,theta,100.d0*n,100.d0*beta
```

```
111   FORMAT(I5,10F12.6)
! Incremento de la altura
        zn=zn+dz
      END DO
      CLOSE(1)
      END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   PLineal/funcion_M_lin.f   -----------------------------
!-------------------------------------------------------------
      SUBROUTINE funcion_M(z,y,n0,F0,M)
! funcion matematica (2.20)
      INCLUDE 'modulos.inc'
      INCLUDE 'datos.inc'
! Variables atmosfericas
      REAL*8 T,P,alfa
      INTEGER z
! Variables de RK4
      TYPE(dreal8) M(4),y(4)
! Variables intermedias
      TYPE(dreal8) C,D,Cp,Rg,theta,r,beta
      TYPE(dreal8) F,u,E,n
! Datos de m
      TYPE(dreal8) n0,F0

! Atmosfera estandar a la altura z
      CALL atmosfera(z,T,P,alfa)
! Cambio de variables
      F = y(1); u = y(2); E = y(3); n = y(4)
! Relaciones funcionales (2.15)
      Cp=ca+((Cp0-ca)*((1.d0-n)/(1.d0-n0)))
      Rg=Ra+ (Rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
      theta=E/Cp
      beta=1.d0/(((1.d0-n)/sigma)+(n*Rg*theta/p))
      r=SQRT(F/(u*beta))
! Variables intermedias
      D = (alfa-beta)/beta
      C = 2.d0*u*eps*r*alfa/F
! Ecuaciones (2.20)
      M(1) = F*C
      M(2) = g*D/u - C*u
      M(3) = c*(Ca*T-E-u*u*.5d0)-u*M(2) - g
      M(4) = (1.d0-n0)*F0*C/F
      END
!-------------------------------------------------------------



!-------------------------------------------------------------
!---   PLineal/error_lin.f   ---------------------------------
!-------------------------------------------------------------
      SUBROUTINE error(z,r,u,theta,costo)
! Evalua una funcion objetivo
!  * error cuadratica entre observaciones y variables
!  * bajo la tropopausa
!  * cada 500m
      INCLUDE 'modulos.inc'
      INCLUDE 'datos.inc'
      TYPE (dreal8) r,u,theta,costo
      INTEGER z,i
      IF ((MOD(z,500).EQ.0).AND.(z.LE.H1)) THEN
        i = (z-z0)/dz
! se podria codear un modulo de sobrecarga para **2
        costo = costo + (r-robs(i))*(r-robs(i))
     *        + (u-uobs(i))*(u-uobs(i))
     *        + (theta-thetaobs(i))*(theta-thetaobs(i))
      ENDIF
      END
!-------------------------------------------------------------



#-------------------------------------------------------------
#---   PLineal/Makefile   ------------------------------------
#-------------------------------------------------------------
# Makefile para linux: compilador gfortran
#! /bin/sh -f
FC= gfortran
LINKER= gfortran
#
FOBJS= mtipo.o msuma.o mmultiplicacion.o mdivision.o \
       mresta.o migual.o mge.o msqrt.o mpotencia2.o \
       programa_lin.o modelo_lin.o funcion_M_lin.o \
       error_lin.o atmosfera.o
#
.SUFFIXES:.o .f90 .f
.f90.o:
       $(FC) -c $(FFLAGS) $<
.f.o:
       $(FC) -c $(FFLAGS) $<
```

```
#
plineal.exe: $(FOBJS)
	$(LINKER) $(LINKFLAGS) -o plineal.exe $(FOBJS)
#
```

```
clean:
	@rm -f *~ *.o *.mod
#-------------------------------------------------------------
```

En el caso de la rutina principal, se inicializan el valor de las condiciones de salidas y la dirección de perturbación en los campos `%d0` y `%d1` de las variables de tipo `dreal8`. Se nota la presencia de una prueba de Taylor (párrafo 3.4.2) para verificar la validez de la diferenciación. *Los cambios debidos a la diferenciaciíon por sobrecarga de los operadores figuran con caracteres marcados en itálico.*

3.4.2. Pruebas de validez

Prueba de Taylor

Los códigos conseguidos por una herramienta automática son generalmente correctos. Sin embargo, es aconsejable verificarlos utilizando la prueba sencilla del desarrollo de Taylor. Como este último ofrece un método para calcular derivadas parciales aproximadas, se puede utilizar para verificar los códigos generados por diferenciación.

Sea el modelo $\mathcal{M}$ con una entrada $u_0 \in IR^m$ y una salida $u \in IR^n$. La prueba consiste en:

1. la elección de una función objetivo $\mathcal{J}$ $IR^n \rightarrow IR$ (puede ser la norma Euclidiana como en el archivo `error_lin.f`),
2. la introduccíon de una llamada a `error` en la rutina `modelo`,
3. la diferenciación de la función compuesta $\mathcal{J} \circ \mathcal{M}$ (diferenciación del conjunto de rutinas por una herramienta),
4. la elección de una dirección de perturbación $\delta u_0 \in IR^m$,
5. el cálculo del ratio r_ω para parámetros ω cada vez más pequeños:

$$r_\omega = \frac{|\mathcal{J} \circ \mathcal{M}(u_0 + \omega \delta u_0)) - \mathcal{J} \circ \mathcal{M}(u_0)|}{|\nabla(\mathcal{J} \circ \mathcal{M}(u_0).\delta u_0|}. \tag{3.5}$$

En teoría, si la relación r_ω converge linealmente hacia 1 para cada dirección δu_0, es decir:

$$\lim_{\omega \to 0} r_\omega = 1, \tag{3.6}$$

entonces la prueba de Taylor está satisfecha: el código diferenciado es correcto.

Tabla 3.1

Comportamiento de la relación r_ω con respecto a parámetros ω.

ω	r_ω
10^{-1}	1.00416669422641
10^{-2}	1.00041740279683
10^{-3}	1.00004174812573
10^{-4}	1.00000417997004
10^{-5}	1.00000046071453
10^{-6}	1.00000017579322
10^{-7}	1.00000238496574
10^{-8}	1.00001601163741
10^{-9}	1.00015640764856
10^{-10}	1.00259269137142

En la práctica, el archivo `error_lin.f` puede implementar una función objetivo que calcula desviaciones entre observaciones y las variables r, u y θ. Aquí, los cálculos se hacen cada $500m$ bajo la tropopausa.

Se consiguen tablas como la tabla 3.1 que presenta un ejemplo de comportamiento de la relación r_ω con respecto a parámetros ω elegidos como potencias negativas decrecientes de 10. Se nota una convergencia lineal de la relación hacia 1 hasta el paso óptimo $\omega = 10^{-6}$. Luego se observa una divergencia que se explica como sigue:

1. pequeños pasos ω son necesarios para la minimización del error de truncamiento proveniente de la aproximación por desarrollo de Taylor en (3.5),
2. pasos ω demasiado pequeños generan un error de redondeo (cancelación) en este desarollo por la substracción de dos números muy cercanos.

La prueba de Taylor se satisface cuando la relación (3.6) tiende primero hacia 1, ya que luego diverge porque el error de cancelación domina el error de truncamiento.

En los códigos largos no es necesario verificar todas las direcciones de perturbación ya que es suficiente verificar la prueba de Taylor en un conjunto de direcciones de perturbación que recorran todas las ramas del código. Se nota que los valores de la Tabla 3.1 pueden variar un poco

de una máquina a otra, lo que cuenta es el comportamiento con respecto a ω.

Cualquier fracaso en la prueba permite apuntar hacia alguno de los siguientes obstáculos:

1. un problema en la diferenciación: códigos diferenciados o modificados a mano, o error de la herramienta,
2. un problema de diferenciabilidad conocido o no por el usuario.

Debajo figuran las instrucciones que permiten hacer la prueba de Taylor. Fueron introducidas en el archivo `programa_lin.f`.

```
!-- instrucciones para hacer una prueba de Taylor --
! Prueba de Taylor
! Valores de salida del crater
      u0%d0  = 300; theta0%d0 = 1000
      r0%d0  = 100; n0%d0  = 0.03
! Eleccion de la direccion de perturbacion
      u0%d1 = 1; theta0%d1 = 0;
      r0%d1 = 0; n0%d1 = 0
! Llamada al modelo lineal tangente
      CALL modelo(u0,theta0,r0,n0,costo_u)
! Prueba de Taylor
      omega = 10.
      DO i = 1,12
        omega = omega/10
        u0p%d0      = u0%d0      + omega*u0%d1
        theta0p%d0 = theta0%d0 + omega*theta0%d1
        r0p%d0      = r0%d0      + omega*r0%d1
        n0p%d0      = n0%d0      + omega*n0%d1
        CALL modelo(u0p,theta0p,r0p,n0p,costo_up)
! Escritura de los resultados como tabla Latex
        WRITE(*,*) '10$^{-',i,'}$ &',
     *  (costo_up%d0-costo_u%d0)/(omega*costo_u%d1),
     *  '\\\\'
      ENDDO
!-------------------------------------------------
```

Hipótesis de la aproximación lineal tangente

Un modelo lineal tangente es una linealización del modelo original. Como toda linealización, ésta es válida sólo en la vecindad del punto donde fue linealizado el modelo. Para los modelos que no son lineales hay que verificar la "hipótesis de la aproximación lineal tangente" buscando en particular su duración de validez.

Sea el modelo pliniano $\mathcal{M}$ que permite evaluar la variable de estado u en cualquier momento calculando:

$$\frac{d\mathrm{u}}{dt} = M(\mathrm{u}, t).$$

Sea u^0 la solución conseguida utilizando la condición inicial $\mathrm{u}(0) = \mathrm{u}_0$, y u^δ la solución conseguida utilizando la condición inicial $\mathrm{u}(0) = \mathrm{u}_0 + \delta$. Se puede deducir:

$$\frac{d\delta}{dt} = \frac{d(\mathrm{u}^\delta - \mathrm{u}^0)}{dt} = M(\mathrm{u}^\delta, t) - M(\mathrm{u}^0, t).$$

Como M es diferenciable con respecto a u, un desarrollo de Taylor nos dice que:

$$\begin{aligned}\frac{d\delta}{dt} &= [M(\mathrm{u}^0, t) + \frac{\partial M}{\partial \mathrm{u}}(\mathrm{u}^0, t).\delta + O(||\delta||^2)] - M(\mathrm{u}^0, t) \\ &\simeq M_l(\mathrm{u}^0, t).\delta,\end{aligned}$$

donde $M_l = \dfrac{\partial M}{\partial \mathrm{u}}$ es el operador lineal tangente relacionado a M.

Con esa fórmula tenemos una aproximación de la propagación de la perturbación δ en función del tiempo. Es decir que en cada momento t, se le puede comparar con el valor exacto $M(\mathrm{u}^\delta) - M(\mathrm{u}^0)$ de la perturbación estudiando la relación:

$$\rho_t = \frac{||(M(\mathrm{u}^\delta, t) - M(\mathrm{u}^0, t)) - M_l(\mathrm{u}^0, t).\delta||}{||M_l(\mathrm{u}^0, t).\delta||}.$$

La hipótesis lineal tangente se satisface sobre el periodo $(0, T)$ si para cada momento $t \in (0, T)$ se verifica $\rho_t << 1$. Procediendo así, se evalúa la importancia de los términos de 2o orden (y mayores) que despreciamos con el uso de la diferenciación de primer orden.

Complejidad algorítmica

Según los límites teóricos establecidos por Morgenstern (1985),el costo en términos del número de operaciones aritméticas del cómputo del modelo lineal $\mathcal{M}_l$ en una dirección es, en el peor de los casos, 4 veces mayor que para el modelo original $\mathcal{M}$. Se puede comprobar esto contando las operaciones (* y /) que se utilizan en la diferenciación de la división.

3.5. Diferenciación automática de fuente a fuente

Dada la fuente del código original y un conjunto de variables activas independientes, una herramienta "fuente a fuente" diferencia el código dando como resultado la fuente del código diferenciado. Es una herramienta de tipo "cálculo formal".

3.5.1. Ejemplos

Para ilustrar la diferenciación fuente a fuente, consideramos las relaciones formales calculando las variables r y C:

$$\begin{cases} r = \sqrt{\dfrac{F}{\beta u}}, \\ C = 2u\varepsilon r\alpha/F. \end{cases}$$

Se programan por las instrucciones:

$$\texttt{r} = \texttt{sqrt(F/(beta} * \texttt{u))}$$
$$\texttt{C} = 2 * \texttt{u} * \texttt{eps} * \texttt{r} * \texttt{alfa/F}.$$

Para diferenciarlas, una herramienta automática las ejecuta en 4 etapas (en la práctica, están ocultas al usuario):

1. descomposición de las instrucciones en otras más elementales, introduciendo variables intermedias (empiezan por una `s`):

```
s1 = beta*u
s2 = F/s1
r = sqrt(s2)
!fin de la primera instrucción

s3 = 2*u
s4 = s3*eps
s5 = s4*r
s6 = s5*alfa
C = s5/F
!fin de la segunda instrucción
```

2. diferenciación de las instrucciones elementales utilizando variables derivadas con sufijo `d` (se hace la diferenciación con respecto a las variables activas de $\mathcal{A}$):

```
!linealización de la primera instrucción
s1d = betad*u + beta*ud
s2d = Fd/s1 - F*s1d/s1**2
rd = s2d/(2*sqrt(s2))

!linealización de la segunda instrucción
s3d = 2*ud
s4d = s3d*eps
s5d = s4d*r + s4*rd
s6d = s5d*alfa
Cd = s5d/F - s5*Fd/F**2
```

3. recomposición de las instrucciones elementales para eliminar algunas de las variables intermedias, y

4. adición de la trayectoria para la evaluación de las derivadas. Al final tenemos:

```
s2d = ... ejercicio ...
s2 = F/(beta*u)          !trayectoria
rd = s2d/(2*SQRT(s2))
r = SQRT(s2)             !trayectoria

Cd = ... ejercicio ...
C = (2*u*eps*r*alfa)/F   !trayectoria
```

Para entender la importancia de la trayectoria, se notará en estas últimas instrucciones que el cálculo de `C` se hace con el valor más reciente de r.

Una herramienta "fuente a fuente" debe ser capaz también de manejar la estructura del programa. Por eso, tiene habilidades para:

1. reproducir las instrucciones de control (pruebas y bucles) presentes en el código original. Por ejemplo, este pedazo de `modelo`:

```
DO WHILE (y(2).GE.10)
... instrucciones
... del modelo
ENDDO
```

   se transforma en el pedazo de código lineal tangente:

```
DO WHILE (y(2).GE.10)
... instrucciones
... del modelo lineal tangente
... y del modelo
ENDDO
```

2. manejar las llamadas a rutinas en el programa. En el programa original tenemos:

```
modelo  ->  atmosfera
        ->  m          ->  atmosfera
        ->  error
```

   lo que conduce a la llamada de las rutinas siguientes en el programa lineal tangente:

```
modelo_d  ->  atmosfera
          ->  m_d        ->  atmosfera
          ->  error_d
```

   Se notan las rutinas lineales tangentes con sufix `_d`. Se nota también que la rutina `atmosfera` no está diferenciada.

Como los softwares de diferenciación "fuente a fuente" saben distinguir las instrucciones de afectación de las instrucciones de control y de las rutinas, el programa completo puede ser diferenciado.

3.5.2. Código lineal tangente producido por Tapenade

El lector interesado puede probar el software `Tapenade` propuesto por el Proyecto Tropics del INRIA en su dirección de Internet `tapenade.inria.fr:8080/tapenade/index.jsp`

Para diferenciar con `Tapenade` se toma en cuenta un cálculo del error, tal que propuesto en el archivo `error.f`, para verificar la prueba de Taylor y luego hacer el análisis de sensibilidad. Se utiliza `Tapenade` como sigue:

1. se "cargan" como fuente (source): `modelo.f`, `funcion_M.f`, `error.f`, `atmosfera.f`, y como "include": `datos.inc`, son copias de los archivos del directorio Internet `www.lpmm.fr/charpentier/Libro/Pliniano`,
2. se elige la rutina principal (top routine) `modelo`,
3. se eligen las variables independientes `r0`, `u0`, `theta0` y `n0`,
4. se diferencia en modo tangente.

El código diferenciado generado por `Tapenade` a partir del modelo de columna es `modelo_d-all.f.tex`. Se localiza en el directorio Internet `www.lpmm.fr/charpentier/Libro/PLinTap` junto a un `Makefile` y al programa principal `programa_tap.f`. Para calcular las sensibilidades de la figura 3.1, se necesita modificar la instrucción de escritura de las variables a lo largo de la columna como indicado en *itálico* en la rutina `modelo_d`. Se nota que Tapenade puede hacer errores de diferenciación (es poco frecuente).

```
******   PLinTap/README   *************************
El directorio PLinTap contiene:
* el programa principal programa_tap.f que permite
  hacer una prueba de Taylor y calcular
  sensibilidades
* las rutinas del codigo pliniano original:
  - datos.inc
  - modelo.f
  - m.f
  - atmosfera.f
* la rutina del modelo diferenciado por Tapenade
  - modelo_d-all.f
* un Makefile para el compilador gfortran
  - Makefile
  produce el archivo ejecutable plintap.exe
* un archivo gnuplot para trazar figuras
  - curvas_c3_tap.gnu
  produze las figuras postscript del capitulo 3
************************************************


!-------------------------------------------------------
!---   PLineal/programa_tap.f  ---------------------------
!-------------------------------------------------------
      PROGRAM plintap
! Este programa calcula el radio (r), la velocidad (u),
! la temperatura (theta), la fraccion de gas (n) y la
! densidad (beta) a lo largo de una columna pliniana por
! el modelo de Woods(1988) en la formulacion (2.9)-(2.10).
! Calcula las sensibilidades por Tapenade
      INCLUDE 'datos.inc'

      REAL*8 r0,u0,theta0,n0     !condiciones de salida
      REAL*8 r0d,u0d,theta0d,n0d !derivadas
      REAL*8 r0p,u0p,theta0p,n0p !cond. salida perturbadas
      REAL*8 costo               !desviacion J
      REAL*8 costod              !derivada
      REAL*8 omega               !paso para Taylor
      REAL*8 costo_u,costo_up    !costo para Taylor
      REAL*8 costo_ud
      INTEGER i
      REAL*8 z
!----
! Prueba de Taylor
! Lectura de observaciones (calculadas para u0=330m/s,
!                           otros valores de salida estan debajo)
      OPEN(1,FILE='observaciones')
      REWIND(1)
      DO i = 1,2000
        READ(1,*) z, robs(i),uobs(i),thetaobs(i)
      ENDDO
      CLOSE(1)
! Valores de salida del crater
      r0=100; u0=300; theta0=1000; n0=0.03
! Eleccion de la direccion de perturbacion
      r0d=0; u0d=1; theta0d=0; n0d=0
! Llamada al modelo lineal tangente
      CALL modelo_d(r0,r0d,u0,u0d,theta0,theta0d,n0,n0d,
     *              costo_u,costo_ud)
! Prueba de Taylor
      omega = 10.
      DO i = 1,10
        omega = omega/10
        r0p     = r0     + omega*r0d
        u0p     = u0     + omega*u0d
        theta0p = theta0 + omega*theta0d
        n0p     = n0     + omega*n0d
        CALL modelo(r0p,u0p,theta0p,n0p,costo_up)
        WRITE(*,*) '10$^{-',i,'}$ &',
     *   (costo_up-costo_u)/(omega*costo_ud),'\\\\'
      ENDDO
!----
!Calculo de las sensibilidades
! Valores de salida
      r0=100; u0=300; theta0=1000; n0=0.03
! Inicialisacion del error y de la componente del gradiente
      costo=0.; costod=0.  !no sirve en este programa

! Direccion de perturbacion (10\% de r0)
      r0d=10; u0d=0; theta0d=0; n0d=0
! Apertura del archivo para escribir las variables
! a lo largo de la columna
      OPEN(1,file='sensr')
! Llamada al modelo lineal
      CALL modelo_d(r0,r0d,u0,u0d,theta0,theta0d,n0,n0d,
     *              costo_u,costo_ud)

! Direccion de perturbacion (10\% de u0)
      r0d=0; u0d=30; theta0d=0; n0d=0
      OPEN(1,file='sensu')
! Llamada al modelo lineal
      CALL modelo_d(r0,r0d,u0,u0d,theta0,theta0d,n0,n0d,
     *              costo_u,costo_ud)

! Direccion de perturbacion (10\% de theta0)
      r0d=0; u0d=0; theta0d=100; n0d=0
      OPEN(1,file='senst')
! Llamada al modelo lineal
      CALL modelo_d(r0,r0d,u0,u0d,theta0,theta0d,n0,n0d,
     *              costo_u,costo_ud)

! Direccion de perturbacion (10\% de n0)
      r0d=0; u0d=0; theta0d=0; n0d=0.003
```

```
      OPEN(1,file='sensn')  !aqui se cambia su nombre
! Llamada al modelo lineal
      CALL modelo_d(r0,r0d,u0,u0d,theta0,theta0d,n0,n0d,
     *              costo_u,costo_ud)
      END

!-------------------------------------------------------------
```

```
C        Generated by TAPENADE     (INRIA, Tropics team)
C  Tapenade 2.2.4 (r2308) - 03/04/2008 10:04
C
C  Differentiation of modelo in forward (tangent) mode:
C   variations  of output variables: costo
C   with respect to input variables: theta0 n0 u0 r0
C-----------------------------------------------------------
C---   Pliniano/modelo.f   ---------------------------------
C-----------------------------------------------------------
      SUBROUTINE MODELO_D(r0, r0d, u0, u0d, theta0, theta0d, n0, n0d,
     +                    costo, costod)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
C Variables de salida del crater
Centradas del modelo
      REAL*8 r0, u0, theta0, n0
      REAL*8 r0d, u0d, theta0d, n0d
      REAL*8 beta0, e0, f0
      REAL*8 beta0d, e0d, f0d
C Costo
      REAL*8 costo
      REAL*8 costod
C Variables de esta DO
      REAL*8 u, theta, r, n, beta, cp, rg, e, f
      REAL*8 ud, thetad, rd, nd, betad, cpd, rgd, ed, fd
C Variables intermedias por RK4
Ccontiene la solucion del modelo yn
      REAL*8 yn(4)
      REAL*8 ynd(4)
Cecuacion (2.20)
      REAL*8 m1(4), m2(4), m3(4), m4(4)
      REAL*8 m1d(4), m2d(4), m3d(4), m4d(4)
Cvar. intermedias
      REAL*8 yy(4)
      REAL*8 yyd(4)
      INTEGER zz
      REAL*8 dzz
C Variables atmosfericas
      REAL*8 t, p, alfa
      INTEGER i, zn
      REAL*8 arg1
      REAL*8 arg1d
      INTEGER ii1
      INTRINSIC SQRT
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
C
C Initializaciones
      dzz = dz*1.d0
      costo = 0.d0
      rg = rg0
      cp = cp0
      CALL ATMOSFERA(z0, t, p, alfa)
      beta0d = -(((-n0d)/sigma+rg*(n0d*theta0+n0*theta0d)/p)/((1.d0-n0)/
     +  sigma+n0*rg*theta0/p)**2)
      beta0 = 1.d0/((1.d0-n0)/sigma+n0*rg*theta0/p)
      zn = z0 + dz
C Cambio de variables
      f0d = (u0d*r0+u0*r0d)*r0*beta0 + u0*r0*(r0d*beta0+r0*beta0d)
      f0 = u0*r0*r0*beta0
      e0d = cp0*theta0d
      e0 = theta0*cp0
      ynd(1) = f0d
      yn(1) = f0
      ynd(2) = u0d
      yn(2) = u0
      ynd(3) = e0d
      yn(3) = e0
      ynd(4) = n0d
      yn(4) = n0
      costod = 0.0
      DO ii1=1,4
        m1d(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m2d(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m3d(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m4d(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        yyd(ii1) = 0.0
```

```
      ENDDO
C Desarollo de la columna
C Solucion por Runge Kutta de orden 4
      DO WHILE (yn(2) .GE. velmin)
C Calculos (2.19)
        CALL FUNCION_M_D(zn, yn, ynd, n0, n0d, f0, f0d, m1, m1d)
        zz = zn + dzz*0.5
        DO i=1,4
          yyd(i) = ynd(i) + dzz*0.5d0*m1d(i)
          yy(i) = yn(i) + dzz*m1(i)*0.5d0
        ENDDO
        CALL FUNCION_M_D(zn, yy, yyd, n0, n0d, f0, f0d, m2, m2d)
        DO i=1,4
          yyd(i) = ynd(i) + dzz*0.5d0*m2d(i)
          yy(i) = yn(i) + dzz*m2(i)*0.5d0
        ENDDO
        CALL FUNCION_M_D(zn, yy, yyd, n0, n0d, f0, f0d, m3, m3d)
        zz = zn + dz
        DO i=1,4
          yyd(i) = ynd(i) + dzz*m3d(i)
          yy(i) = yn(i) + dzz*m3(i)
        ENDDO
        CALL FUNCION_M_D(zn, yy, yyd, n0, n0d, f0, f0d, m4, m4d)
C Ecuacion (2.18)
Cyn
        DO i=1,4
          ynd(i) = ynd(i) + dzz*(m1d(i)+2.d0*m2d(i)+2.d0*m3d(i)+m4d(i))/
     +      6.d0
          yn(i) = yn(i) + dzz*(m1(i)+2.d0*m2(i)+2.d0*m3(i)+m4(i))/6.d0
        ENDDO
C Cambio de variables inverso
        fd = ynd(1)
        f = yn(1)
        ud = ynd(2)
        u = yn(2)
        ed = ynd(3)
        e = yn(3)
        nd = ynd(4)
        n = yn(4)
C Relacions funcionales
        cpd = (cp0-ca)*((1.d0-n)*n0d-nd*(1.d0-n0))/(1.d0-n0)**2
        cp = ca + (cp0-ca)*((1.d0-n)/(1.d0-n0))
        rgd = ((((rg0-ra)*n0d*(1.d0-n0)+(rg0-ra)*n0*n0d)*(1.d0-n)/(1.d0-
     +    n0)**2-(rg0-ra)*n0*nd/(1.d0-n0))*n-(rg0-ra)*n0*(1.d0-n)*nd/(
     +    1.d0-n0))/n**2
        rg = ra + (rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
        thetad = (ed*cp-e*cpd)/cp**2
        theta = e/cp
        betad = -(((-nd)/sigma+((nd*rg+n*rgd)*theta+n*rg*thetad)/p)/((
     +    1.d0-n)/sigma+n*rg*theta/p)**2)
        beta = 1.d0/((1.d0-n)/sigma+n*rg*theta/p)
        arg1d = (fd*u*beta-f*(ud*beta+u*betad))/(u*beta)**2
        arg1 = f/(u*beta)
        IF (arg1 .EQ. 0.0) THEN
          rd = 0.0
        ELSE
          rd = arg1d/(2.0*SQRT(arg1))
        END IF
        r = SQRT(arg1)
C evaluacion del error
        CALL ERROR_D(zn, r, rd, u, ud, theta, thetad, costo, costod)
C Escritura del archivo de resultados
        WRITE(1, 111) zn, r, u, theta, 100*n, 100*beta
C Incremento de la altura
        zn = zn + dz
      ENDDO
      CLOSE(1)
 111  FORMAT(i5,10f12.6)
      END
C--------------------------------------------------------------

C  Differentiation of error in forward (tangent) mode:
C   variations  of output variables: costo
C   with respect to input variables: costo r u theta
C--------------------------------------------------------------
C---   Pliniano/error.f   ------------------------------------
C--------------------------------------------------------------
      SUBROUTINE ERROR_D(z, r, rd, u, ud, theta, thetad, costo, costod)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
      REAL*8 r, u, theta, costo
      REAL*8 rd, ud, thetad, costod
      INTEGER z, i
      INTRINSIC MOD
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
```

```
C
      IF (MOD(z, 500) .EQ. 0 .AND. z .LE. h1) THEN
        i = (z-z0)/dz
        costod = costod + rd*(r-robs(i)) + (r-robs(i))*rd + ud*(u-uobs(i
     +    )) + (u-uobs(i))*ud + thetad*(theta-thetaobs(i)) + (theta-
     +    thetaobs(i))*thetad
        costo = costo + (r-robs(i))*(r-robs(i)) + (u-uobs(i))*(u-uobs(i)
     +    ) + (theta-thetaobs(i))*(theta-thetaobs(i))
       END IF
       END
C-----------------------------------------------------------------

C  Differentiation of funcion_m in forward (tangent) mode:
C   variations  of output variables: m
C   with respect to input variables: m f0 y n0
C-----------------------------------------------------------------
C---   Pliniano/funcion_M.f   ------------------------------------
C-----------------------------------------------------------------
      SUBROUTINE FUNCION_M_D(z, y, yd, n0, n0d, f0, f0d, m, md)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
C Variables atmosfericas,
      REAL*8 t, p, alfa
      INTEGER z
C Variables de RK4
      REAL*8 m(4), y(4)
      REAL*8 md(4), yd(4)
C Variables intermedias
      REAL*8 c, d, cp, rg, theta, r, beta
      REAL*8 cd, dd, cpd, rgd, thetad, rd, betad
      REAL*8 f, u, e, n
      REAL*8 fd, ud, ed, nd
      REAL*8 n0, f0
      REAL*8 arg1
      REAL*8 arg1d
      INTRINSIC SQRT
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
      REAL*8 n0d, f0d
C
C Atmosfera estandar a la altura z
      CALL ATMOSFERA(z, t, p, alfa)
C Cambio de variables
      fd = yd(1)
      f = y(1)
      ud = yd(2)
      u = y(2)
      ed = yd(3)
      e = y(3)
      nd = yd(4)
      n = y(4)
C Relaciones funcionales (2.15)
      cpd = (cp0-ca)*((1.d0-n)*n0d-nd*(1.d0-n0))/(1.d0-n0)**2
      cp = ca + (cp0-ca)*((1.d0-n)/(1.d0-n0))
      rgd = ((((rg0-ra)*n0d*(1.d0-n0)+(rg0-ra)*n0*n0d)*(1.d0-n)/(1.d0-n0
     +  )**2-(rg0-ra)*n0*nd/(1.d0-n0))*n-(rg0-ra)*n0*(1.d0-n)*nd/(1.d0-
     +  n0))/n**2
      rg = ra + (rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
      thetad = (ed*cp-e*cpd)/cp**2
      theta = e/cp
      betad = -(((-nd)/sigma+((nd*rg+n*rgd)*theta+n*rg*thetad)/p)/((1.d0
     +  -n)/sigma+n*rg*theta/p)**2)
      beta = 1.d0/((1.d0-n)/sigma+n*rg*theta/p)
      arg1d = (fd*u*beta-f*(ud*beta+u*betad))/(u*beta)**2
      arg1 = f/(u*beta)
      IF (arg1 .EQ. 0.0) THEN
        rd = 0.0
      ELSE
        rd = arg1d/(2.0*SQRT(arg1))
      END IF
      r = SQRT(arg1)
C Variables intermedias
      dd = (-(betad*beta)-(alfa-beta)*betad)/beta**2
      d = (alfa-beta)/beta
      cd = (2.d0*eps*alfa*(ud*r+u*rd)*f-2.d0*u*eps*r*alfa*fd)/f**2
      c = 2.d0*u*eps*r*alfa/f
C Ecuaciones (2.20)
      md(1) = fd*c + f*cd
      m(1) = f*c
      md(2) = (g*dd*u-g*d*ud)/u**2 - cd*u - c*ud
      m(2) = g*d/u - c*u
      md(3) = cd*(ca*t-e-u*u*.5d0) + c*(-ed-.5d0*(ud*u+u*ud)) - ud*m(2)
     +  - u*md(2)
      m(3) = c*(ca*t-e-u*u*.5d0) - u*m(2) - g
      md(4) = (((1.d0-n0)*(f0d*c+f0*cd)-n0d*f0*c)*f-(1.d0-n0)*f0*c*fd)/f
     +  **2
```

```
      m(4) = (1.d0-n0)*f0*c/f
      END
C-----------------------------------------------------------
```

4. Modelación inversa

Una de las características de la investigación científica es su afán de predicción. En este sentido, quizá una de las áreas en que es muy patente este esfuerzo es en el de la predicción meteorológica. En esta área la disciplina ha tenido un desarrollo que puede calificarse como exitoso. Tal éxito ha resultado de los esfuerzos concentrados en tres direcciones: la observación, la modelación y la informática.

La observación de los fenómenos naturales comenzó en una etapa muy temprana del desarrollo humano. Desde tiempos prehistóricos muchas y diversas culturas construyeron calendarios solares con el fin de establecer las épocas de siembra. Con el advienimiento de la ciencia moderna se fundamentó la observación como base esencial para la determinación de las leyes de la naturaleza, pero se advirtió que su interpretación plena requería de un marco teórico (o modelo) del fenómeno estudiado. Tales modelos requieren la definición precisa de las variables observables y muy a menudo la introducción de otras no contempladas con anterioridad. De esta manera, la investigación moderna se basa en métodos que conjuntan observaciones y modelos. Un ejemplo de esta metodología lo constituye el trabajo de Gauss en Astronomía que introdujo el método de los "mínimos cuadrados" en su modelo de órbitas planetarias para encontrar algunos parámetros. A su modelo, agregó una función objetivo (tal `error`, en nuestro caso) midiendo la desviación cuadrada entre sus órbitas parametrizadas y sus datos.

La discusión anterior es de vital importancia para comprender la diferencia entre un problema físico expresado en términos matemáticos y un problema solamente matemático. La diferencia estriba precisamente en que los valores asociados a las observaciones, es decir los datos, no son exactos (aunque pueden ser muy precisos). Un ejemplo simple puede ilustrar más fácilmente la diferencia. Supongamos que un modelo nos establece que las variables A, B y C están relacionadas por la siguiente ecuación:

$$\frac{A}{B} = C.$$

Esta ecuación puede ser vista como una relación matemática en la que A y B tienen valores exactos y por lo tanto C tendrá un valor exacto aun si es un número racional. En un ejemplo trivial si A=12 y B=3 se sigue que C=4. Supongamos ahora que conocemos los valores de A y C y deseamos conocer el valor de B. Si conocemos A y C, éste es el problema inverso del anterior y desde luego su solución es así mismo trivial ya que:

$$\frac{A}{C} = B.$$

Y por lo tanto B=3. Supongamos ahora que conocemos A y C a través de observaciones, en este caso los valores de A y C no serán exactos sino de la forma $A = 12 \pm \delta_A$ y $C = 4 \pm \delta_C$ y el problema de obtener B es menos trivial que el problema anterior. Con este ejemplo no se pretende argumentar que los problemas matemáticos sean triviales comparados a los planteados por los modelos físicos, sino que el manejo de datos introduce un elemento adicional por determinar y que, por otro lado, requiere de ampliar los métodos matemáticos empleados en la solución de estos problemas inversos.

Podemos definir de manera general, un problema inverso con tratamiento de datos como la búsqueda de las variables de entrada del modelo que minimizan la distancia entre los datos y el resultado del modelo. En términos matemáticos, tal problema involucra genéricamente los siguientes ingredientes.

1. Datos v^o. Se supone que los errores de medición tienen un error Gaussiano (o ruido blanco).

2. Un modelo $\mathcal{M}$. Se expone el método usando el modelo general:

$$(\mathcal{M}) \quad \begin{cases} \dfrac{d\mathrm{u}}{dt} = M(\mathrm{u}) \text{ en } (z_0, z_{top}), \\ \mathrm{u}(0) = \mathrm{u}_0. \end{cases} \tag{4.1}$$

3. La variable de control. Ésta contiene las entradas $\mathrm{u}_0 \in \mathcal{U}_0$ que queremos encontrar. Se le llama variable de control porque se utiliza para manejar numéricamente la solución del problema inverso. Al espacio $\mathcal{U}_0$ se le llama espacio de los controles.

4. La función objetivo $\mathcal{J}$. Ésta mide la desviación cuadrada entre los datos observados v^o y los resultados u del modelo. Se escribe:

$$\begin{aligned} \mathcal{J}(\mathrm{u}) &= \frac{1}{2}\int_{z_0}^{z_{top}} |\mathcal{H}\mathrm{u} - \mathrm{v}^o|^2, = \\ &= \frac{1}{2}\int_{z_0}^{z_{top}} (\mathcal{H}\mathrm{u} - \mathrm{v}^o)^*.(\mathcal{H}\mathrm{u} - \mathrm{v}^o), \end{aligned}$$

donde $\mathcal{H}$ es un operador lineal de observación que permite mapear el conjunto $\mathcal{U}$ de los estados admisibles que puede tener la variable u al conjunto $\mathcal{U}^{obs}$ de los datos colectados observando esta misma variable.

5. El problema inverso (o problema de minimización). Éste se escribe como:

$$\begin{aligned} &\textit{Buscar } \mathrm{u}_0^* \textit{ tal como} \\ &\min_{\mathrm{u}\in\mathcal{U}} \mathcal{J}o\mathcal{M}(\mathrm{u}_0) = \mathcal{J}o\mathcal{M}(\mathrm{u}_0^*), \end{aligned} \tag{4.2}$$

donde u_0^* es la variable de entrada óptima (o control óptimo) tal que la solución u^* del modelo calculado desde u_0^* minimiza la distancia $\mathcal{J}$ con los datos.

Una vez propuesto el problema inverso, son necesarias herramientas matemáticas e informáticas para resolverlo. Al problema inverso (4.2) se le llama también problema de "asimilación de datos"por lo que la función objetivo que minimizar involucra datos.

Históricamente, los desarrollos metodológicos en matemática e informática se sucedieron a un ritmo increíble aprovechando los recursos informáticos nuevos en dos direcciones: aumento de la complejidad de los modelos (se puede ver a Textor et al., 2005, para una revisión de los modelos de erupciones plinianas), y/o nuevos objetivos científicos tales como la solución de problemas inversos. Estos últimos pueden ser realizados por medio de:

- métodos estadísticos. Algunos están basados sobre la teoría estadística del filtrado lineal de Kalman (1960). Se desarrollaron mejoras tal como el filtro de Kalman extendido reducido (Pham, Verron y and Roubaud, 1998) y filtros de conjuntos (Evensen, 2003).Hoy en día se utilizan principalmente en oceanología.

- métodos determinísticos. Le Dimet y Talagrand (1986) y Lewis y Derber (1985) propusieron el método de la "asimilación variacional de datos". Le llamaron "asimilación de datos"para apuntar el hecho que los datos se vuelven parte de la modelación, mientras el término "variacional"indica su liga a la teoría de control óptimo (Lions, 1971). La solución del problema de minimización se calcula por un método de optimización basado en un cálculo de gradiente. El punto fundamental del método de asimilación variacional de datos está en la escritura de un modelo adjunto que permite conseguir gradientes de bajo costo, aún la variable de control es un vector de dimensión grande. Es el método actualmente utilizado en los centros de meteorología operacional.

El lector puede referirse al sitio NEOS (`www-neos.mcs.anl.gov`) para ver una revisión bastante completa de técnicas y herramientas informáticas de optimización.

A continuación estudiamos el segundo enfoque con mayor detalle. Comenzaremos con experimentos numéricos de asimilación variacional de datos. En particular presentamos cómo se usa el algoritmo de optimización cuasi-Newton (Gilbert y Lemaréchal, 1989; Byrd, Lu y Nocedal, 1995).Posteriormente presentamos las teorías para construir el modelo adjunto y el código adjunto. Como lo veremos, permiten calcular los gradientes de una manera barata.

4.1. Identificación de condiciones de salida

4.1.1. Contexto experimental

Hay que reconocer que las columnas plinianas no son fenómenos lo bastante frecuentes, al menos durante el tiempo en que las observaciones señaladas han sido posibles, como para permitir crear grandes archivos de columna. De hecho, las erupciones plinianas han sido más conocidas por los depósitos que han dejado que por las imágenes de vídeo de sus columnas.

Al seguir, presentamos unos medios para conseguir datos de columna. Luego describimos un experimento gemelo en el cual los datos son producidos por el modelo estudiado.

Observaciones

Los radares Doppler dispuestos en los aeropuertos para supervisar el tráfico permiten seguir ciertas actividades volcánicas tal como las nubes de ceniza (Casadevall, T.J., 1993; ICAO, 2001).Se nota también el monitoreo bastante completo de la erupción del Pinatubo (1991) realizado por Scott Oswalt, Nichols y O'Hara (1996)por medio de viejos dispositivos militares. Hoy en día, existen pequeños dispositivos para la observación de las características volcánicas tales como las velocidades horizontales y verticales, las nubes de cenizas y gases (Harris et al., 1981), el tamaño de los clastos (Harris y Rose, 1983). Dispositivos infrarrojos (Patrick, 2007) pueden dar acceso a datos de temperatura.

El análisis de secuencias de imágenes, estéreoscópicas o infrarrojas, resulta muy interesante para adquirir información global tal como son el contorno de la columna o una estimación de su velocidad. Se pueden imaginar procesos de tratamiento de imágenes para la detección y el seguimiento de vórtices de turbulencia. Otra posibilidad es la de aprovechar la coherencia temporal de la columna durante un periodo pequeño (determinado por la distancia entre 2 vórtices de turbulencia y la velocidad de la mezcla volcánica) para construir una envolvente de la columna que tendría menos ruido que un contorno instantáneo. Tal secuencia de imagen permite también una estimación de la velocidad de la mezcla (ver al párrafo 4.2).

En general, los datos de campo, cuando existen, son generalmente parciales ya que no es posible observar la columna en cada punto del espacio y a cada instante. Además, es posible que las variables físicas que se observan con los instrumentos de medida no sean las que se usan directamente en la modelación. Aunque los avances tecnológicos proporcionan nuevos dispositivos para el monitoreo, gran parte de la información sobre una columna permanece oculta.

Experimentos gemelos

Llamamos experimentos gemelos a aquellos en los cuales los datos son producidos por el modelo estudiado. Los datos sintéticos, es decir los producidos por el modelo, llevan obviamente información sobre las variables de la modelación. Se conocen todas y todas se pueden utilizar. En este caso, la identificación de la variable de control (condición de salida por ejemplo) tiene que ser exacta. Si no, significa que el procedimiento de optimización no es correcto. Finalmente, se puede agregar un ruido blanco a los datos sintéticos para cambiarlas en "datos con sabor real" y probar el procedimiento. En tal caso, tanto el error de modelación como el error de medida están bajo el control del científico. Un experimento gemelo se construye como sigue:

1. Elección del modelo $\mathcal{M}$ y de entradas óptimas u_0^*,
2. Generación de los datos sintéticos:
   - se calcula $\mathcal{M}(u_0)$ para producir u^*,
   - se agrega un ruido Gausiano a u para tener datos perturbados u^p,
   - se elige un operador de observación $\mathcal{H}$,
   - se diezman los datos para tener datos estilo real $u^{obs} = \mathcal{H}u^p$,

   Para agregar un ruido blanco, se puede utilizar las rutinas `gausspolar` y `creeobs` presentadas en anexo A.1.
3. Elección de la función objetivo:

$$\mathcal{J}(u) = \int_{z_0}^{z_{top}} |\mathcal{H}u - u^{obs}|,$$

para la comparación de cualquier solución u del modelo con los observaciones u^{obs},

4. Elección de un iterado inicial u_0^0 (el índice superior indica el número de la iteración del proceso de descenso),

5. Uso del método iterativo de optimización (realizado desde u_0^0) para encontrar las entradas $u_0^\#$ que minimizan $\mathcal{J}o\mathcal{M}$,

6. Evaluación del rendimiento del método de minimización por su capacidad para aproximar $u_0^\#$ a la verdadera condición inicial u_0^*. Aquí es muy importante que el ruido sea blanco.

No se pierde tiempo haciendo este tipo de experimentos ya que si el proceso no funciona con datos sintéticos, generalmente no funciona con datos reales.

4.1.2. Modelación pliniana inversa

Presentamos los elementos de nuestro problema inverso. Consideramos:

1. el modelo pliniano $\mathcal{M} = \mathcal{P}o\mathcal{T}$ en su forma modificada (2.7)–(2.10). Se supone que todo está bien definido desde un punto de vista matemático y que el sistema (4.1) tiene una solución única.

2. la variable de control $u_0 = (r_0, u_0, \theta_0, n_0)$,

3. datos $v^{obs} = (r^{obs}, u^{obs}, \theta^{obs})$,

4. la función objetivo:

$$\mathcal{J}(u) = \int_{z_0}^{z_{tropo}} |\mathcal{H}u - v^{obs}|^2,$$

es decir:

$$\begin{aligned}\mathcal{J}(r,u,\theta,n) &= \int_{z_0}^{z_{tropo}} |\mathcal{H}_r r - r^{obs}|^2 + \\ &+ \int_{z_0}^{z_{tropo}} |\mathcal{H}_u u - u^{obs}|^2 + \\ &+ \int_{z_0}^{z_{tropo}} |\mathcal{H}_\theta \theta - \theta^{obs}|^2.\end{aligned}$$

Nótese que las integrales se pueden calcular desde el cráter (de altura z_0) hasta la tropopausa (de altura z_{tropo}). Este límite superior tiene dos ventajas. Primero, las columnas plinianas que nos interesan tienen alturas superiores a la de la tropopausa. Segundo, la altura de la columna varía con respecto a las condiciones de salida. En el proceso iterativo de optimización no se podría comparar $\mathcal{J}(r_1, u_1, \theta_1, n_1)$ a $\mathcal{J}(r_2, u_2, \theta_2, n_2)$ de una manera justa si la altura de la columna difiere del conjunto $(r_1, u_1, \theta_1, n_1)$ al conjunto $(r_2, u_2, \theta_2, n_2)$. Se nota también que, con respecto a los análisis de sensibilidades del párrafo 3.1, no es necesario utilizar datos de la fracción de gas (Charpentier, 2008). En el cálculo numérico, las observaciones v^{obs} y la solución u son conocidas en puntos discretos distribuidos a pasos regulares (por el uso de los operadores de observación $\mathcal{H}$). La función objetivo discreta puede ser escrita como:

$$\mathcal{J}(u) = \sum_p |u(z_p) - v^{obs}(z_p)|^2, \qquad (4.3)$$

donde las alturas z_p pueden ser elegidas cada $500m$ por ejemplo. En un experimento real, se comparan variables y observaciones en los puntos de medida o en puntos interpolados.

5. el espacio de control $\mathcal{U}_0$ tal que $r_0 \in (50m, 200m)$, $u_0 \in (200m.s^{-1}, 450m.s^{-1})$, $\theta_0 \in (800K, 1300K)$ y $n_0 \in (0{,}02, 0{,}06)$. Estos intervalos permiten calcular columnas que no se colapsan.

6. el problema inverso:

$$\begin{aligned}&Buscar\ \ u_0^* = (r_0^*, u_0^*, \theta_0^*, n_0^*)\ \ tal\ que \\ &\min_{u \in \mathcal{U}} \mathcal{J}o\mathcal{M}(u_0) = \mathcal{J}o\mathcal{M}(u_0^*),\end{aligned}$$

7. el método de asimilación variacional de datos. Buscamos la solución u_0^* como un punto de $\mathcal{U}_0$ donde el gradiente $\nabla\mathcal{J}o\mathcal{M}$ es cero, es decir:

$$\begin{aligned}&Buscar\ \ u_0^* = (r_0^*, u_0^*, \theta_0^*, n_0^*)\ \ tal\ que \\ &\nabla\mathcal{J}o\mathcal{M}(r_0^*, u_0^*, \theta_0^*, n_0^*) = 0.\end{aligned} \qquad (4.4)$$

Cuando la función $\mathcal{J}o\mathcal{M}$ es convexa (tal como la parábola x^2 por ejemplo), existe un único mínimo. En general no es el caso y sólo se puede encontrar un mínimo local.

4.1.3. Cálculo del gradiente

En modo lineal tangente

En el capitulo 3, realizamos análisis de sensibilidades derivando el modelo con respecto a una

de sus entradas. Corresponde al cálculo:

$$\frac{\partial \mathcal{M}}{\partial \mathrm{u}_0}(\mathrm{u}_0).\delta \mathrm{u}_0.$$

Se puede también calcular la sensibilidad de $\mathcal{J}o\mathcal{M}$:

$$\frac{\partial \mathcal{J}o\mathcal{M}}{\partial \mathrm{u}_0}(\mathrm{u}_0).\delta \mathrm{u}_0 = \frac{\partial \mathcal{J}}{\partial \mathrm{u}}(\mathcal{M}(\mathrm{u}_0)).\frac{\partial \mathcal{M}}{\partial \mathrm{u}_0}(\mathrm{u}_0).\delta \mathrm{u}_0,$$

tal como en la prueba de Taylor. El gradiente no es más que el vector de las sensibilidades de $\mathcal{J}o\mathcal{M}$ calculado con respecto a cada elemento de la variable de control. Es decir:

$$\nabla(\mathcal{J}o\mathcal{M}) = \\ = \left(\frac{\partial \mathcal{J}o\mathcal{M}}{\partial r_0}, \frac{\partial \mathcal{J}o\mathcal{M}}{\partial u_0}, \frac{\partial \mathcal{J}o\mathcal{M}}{\partial \theta_0}, \frac{\partial \mathcal{J}o\mathcal{M}}{\partial n_0}\right).$$

El modo lineal tangente de la diferenciación automática permite obtener códigos para cálculos de gradiente. Se puede:

1. utilizar los códigos presentados en el capitulo 3 llamándolos en las 4 direcciones de perturbación de nuestra variable de control,

2. diferenciar los códigos en modo lineal tangente vectorial (capítulo 3) por:

   a) `Tapenade` usando el modo lineal tangente vectorial (ejercicio), o

   b) sobrecarga de los operadores cambiando el tipo y las carpetas para tomar en cuenta 4 direcciones de perturbación al mismo tiempo (ejercicio).

En modo adjunto

Se puede resumir la idea en pocas palabras. Se busca la perturbación $\delta_f \in IR$ que agregar a la salida $f(x) = \mathcal{J}o\mathcal{M}(x) \in IR$ para conseguir la perturbación $\delta \mathrm{u}_0$ de las entradas u_0 del modelo. En consecuencia, su uso es muy interesante cuando la dimensión de la variable de control es importante (del orden de un millón en meteorología), porque su costo es independiente del número de direcciones de perturbación (Morgenstern, 1985).

Desde el punto de vista teórico se puede explicar como en Talagrand y Courtier (1987).Sean $\mathcal{Y}$ y $\mathcal{W}$ dos espacios de Hilbert asociados con los productos escalares $< .,. >$ y $[.,.]$ respectivamente. Sean A un operador lineal continuo de $\mathcal{Y}$ en $\mathcal{W}$ y A^* su operador transpuesto (u operador adjunto aquí), se puede escribir:

$$\begin{aligned} &\forall \delta y \in \mathcal{Y},\ \forall \delta w \in \mathcal{W}, \\ &< \delta w, A.\delta y > = \delta w^*.A.\delta y = \\ &\qquad = (A^*.\delta w)^*.\delta y = \\ &\qquad = [A^*.\delta w, \delta y]. \end{aligned} \tag{4.5}$$

Se aplica la fórmula (4.5) a los productos de operadores. Así, cualquiera que sea el operador B lineal continuo de $\mathcal{W}$ en cualquier otro espacio de Hilbert, se prueba que:

$$(B.A)^* = A^*.B^*. \tag{4.6}$$

Nuestros operadores lineales $\nabla\mathcal{J}$ y $\nabla\mathcal{M}$ satisfacen esta hipótesis. Sea $\delta w = \nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0).\delta \mathrm{u}_0$ una perturbación de $w = \mathcal{J}o\mathcal{M}(\mathrm{u}_0) \in IR$. Haciendo un cambio de escala, uno puede elegir $\delta w = 1$ para escribir:

$$\begin{aligned} 1 &= (\delta w, \delta w) = \\ &= (\delta w, \nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0).\delta \mathrm{u}_0) = \\ &= \delta w^*.\nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0).\delta \mathrm{u}_0 = \\ &= (\nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0))^*.\delta w)^*.\delta \mathrm{u}_0 = \\ &= [(\nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0))^*.\delta w, \delta \mathrm{u}_0] = \\ &= [\hat{\mathrm{u}}(z_0), \delta \mathrm{u}_0], \end{aligned}$$

donde $\hat{\mathrm{u}} = (\nabla(\mathcal{J}o\mathcal{M})(\mathrm{u}_0))^*$ es la variable adjunta, pertenece a $\mathcal{U}$.

En la prueba original, Le Dimet y Talagrand (1986) se utilizó el método Lagrangiano para evidenciar el modelo adjunto. Tal técnica se utiliza en numerosas pruebas matemáticas (control óptimo o descomposición de dominio por ejemplo) ya que permite tratar problemas inversos escribiendo un problema de minimización bajo una restricción por medio de una función $\mathcal{L}$ llamada Lagrangiano. Tal función presenta, por construcción, un punto silla cuyas propiedades dan informaciones sobre el mínimo del problema inverso original. Para imaginarse un punto silla, se puede pensar en una silla de caballo o el punto "paso de Cortés"que es a la vez el punto de mayor altura en el camino montañoso que usó el conquistador Hernán Cortés para ir de Puebla al Valle de México, y también el punto de altura mínima entre los volcanes Popocatépetl e Iztaccíhuatl.

En la asimilación de datos, la función $\mathcal{L}$ se usa para escribir el problema de minimización (4.2) bajo la constricción del modelo (4.1):

$$\begin{aligned}
&\mathcal{L}(\mathrm{u}_0, \mathrm{u}, q) = \\
&= \mathcal{J}o\mathcal{M}(\mathrm{u}_0) + \\
&\quad + \int_{z_0}^{z_{tropo}} < q, \frac{d\mathrm{u}}{dz} - M(\mathrm{u}) > dz = \\
&= \mathcal{J}(\mathrm{u}) + \int_{z_0}^{z_{tropo}} < q, \frac{d\mathrm{u}}{dz} - M(\mathrm{u}) > dz = \\
&= \mathcal{J}(\mathrm{u}) - \\
&\quad - \int_{z_0}^{z_{tropo}} < \frac{dq}{dz}, \mathrm{u} > dz - \\
&\quad - \int_{z_0}^{z_{tropo}} < q, M(\mathrm{u}) > dz + \\
&\quad + < q(z_{tropo}), \mathrm{u}(z_{tropo}) > - \\
&\quad - < q(z_0), \mathrm{u}_0 > .
\end{aligned} \tag{4.7}$$

En esta ecuación, la condición de salida u_0 es la variable de control a identificar, u es la variable de estado y q es una nueva variable, dicha variable adjunta. El modelo adjunto se consigue buscando las relaciones que tienen que satisfacer q para que u_0^* sea el mínimo de la función $\mathcal{J}o\mathcal{M}$.

Teorema 1 *Si el punto (u_0^*, u^*, q^*) es "punto silla" de la función $\mathcal{L}$, satisface:*

$$\begin{aligned}
&\forall (u_0, u), \quad \forall q, \\
&\mathcal{L}(u_0^*, u^*, q) \leq \mathcal{L}(u_0^*, u^*, q^*) \leq \mathcal{L}(u_0, u, q^*),
\end{aligned}$$

entonces u_0^* es la solución del problema de minimización (4.2).
La prueba figura en el anexo A.2.

Las propiedades del punto de silla de $\mathcal{L}$ permiten caracterizar el mínimo u_0^* de $\mathcal{J}o\mathcal{M}$.

1. La propiedad "$\mathcal{L}$ es mínimo en u^*" sirve en la construcción del modelo adjunto.
   La función $\mathcal{L}$ está derivada con respecto a u y evaluada en el punto $(\mathrm{u}_0^*, \mathrm{u}^*, q^*)$. El mínimo tiene que satisfacer:

   $$< \frac{\partial \mathcal{L}}{\partial \mathrm{u}}(\mathrm{u}_0^*, \mathrm{u}^*, q^*), \phi >= 0, \quad \forall \phi.$$

   Desarrollando $\mathcal{J}(\mathrm{u})$, se consigue:

   $$\begin{aligned}
   &\int_{z_0}^{z_{tropo}} < \mathcal{H}\mathrm{u}^* - \mathrm{u}_{obs}, \mathcal{H}\phi > dz - \\
   &- \int_{z_0}^{z_{tropo}} < \frac{dq^*}{dz}, \phi > dz - \\
   &- \int_{z_0}^{z_{tropo}} < q^*, \left[\frac{\partial M}{\partial \mathrm{u}}(\mathrm{u}^*)\right].\phi > dz + \\
   &+ < q^*(z_{tropo}), \phi(z_{tropo}) >= 0, \quad \forall \phi,
   \end{aligned}$$

   porque la derivada de u_0 con respecto a u es cero.

   La construcción del modelo adjunto se hace eligiendo $q^*(z_{tropo}) = 0$ y aislando ϕ. Eso supone transponer unos operadores. La ecuación se escribe:

   $$\begin{aligned}
   &\forall \phi, \\
   &\int_{z_0}^{z_{tropo}} < \mathcal{H}^T(\mathcal{H}\mathrm{u}^* - \mathrm{u}_{obs}) - \frac{dq^*}{dz} - \\
   &\qquad - \left[\frac{\partial M}{\partial \mathrm{u}}(\mathrm{u}^*)\right]^T .q^*, \phi > dz = 0.
   \end{aligned} \tag{4.8}$$

   Se deduce el modelo adjunto $\mathcal{M}_a$:

   $$\begin{cases}
   z \in (z_0, z_{tropo}), \\
   -\frac{dq^*}{dz} = \left[x\frac{\partial M}{\partial \mathrm{u}}(\mathrm{u}^*)\right]^T .q^* - \\
   \qquad\qquad -\mathcal{H}^T(\mathcal{H}\mathrm{u}^* - \mathrm{u}_{obs}), \\
   q^*(z_{tropo}) = 0.
   \end{cases} \tag{4.9}$$

   Es un sistema "retrógrado" integrable de z_{tropo} hasta z_0.

2. "$\mathcal{L}$ es mínimo en u_0^*" sirve al cálculo del gradiente.
   La función $\mathcal{L}$ es derivada con respecto a u_0 y evaluada en el punto $(\mathrm{u}_0^*, \mathrm{u}^*, q^*)$. El mínimo tiene que satisfacer:

   $$< \frac{\partial \mathcal{L}}{\partial \mathrm{u}_0}(\mathrm{u}_0^*, \mathrm{u}^*, q^*), \phi >= 0, \quad \forall \phi. \tag{4.10}$$

   La ecuación (4.7) está derivada con respecto a u_0 y simplificada gracias a (4.8):

   $$\begin{aligned}
   &\forall \phi, \\
   &< \frac{\partial \mathcal{L}}{\partial \mathrm{u}}(\mathrm{u}_0^*, \mathrm{u}^*, q^*), \phi >= - < q^*(z_0), \phi > .
   \end{aligned}$$

   Se deduce que:

   $$\frac{\partial \mathcal{L}}{\partial \mathrm{u}_0}(\mathrm{u}_0^*, \mathrm{u}^*, q^*) = -q^*(z_0).$$

   La desigualdad derecha del teorema enseña que si $(\mathrm{u}_0^*, \mathrm{u}^*, q^*)$ es punto silla de $\mathcal{L}$, entonces u_0^* es un mínimo de $\mathcal{L}(\mathrm{u}_0, \mathrm{u}, q^*)$ y de $\mathcal{J}o\mathcal{M}(\mathrm{u}_0)$. En consecuencia, los gradientes de $\mathcal{J}o\mathcal{M}$ y $\mathcal{L}$ con respecto a u_0 son iguales al valor de la variable adjunta q^* al nivel del cráter z_0. Es decir:

   $$\frac{\partial \mathcal{L}}{\partial \mathrm{u}_0}(\mathrm{u}_0^*, \mathrm{u}^*, q^*) = \frac{\partial (\mathcal{J}o\mathcal{M})}{\partial \mathrm{u}_0}(\mathrm{u}_0^*) = -q^*(z_0).$$

Sistema de optimalidad

Una vez construidas las diferentes ecuaciones se reúnen en el sistema de optimalidad que contiene:

1. el modelo $\mathcal{M}$:

$$\begin{cases} \dfrac{d\mathrm{u}}{dz} = M(\mathrm{u}) \quad \text{en } (z_0, z_{tropo}), \\ \mathrm{u}(z_0) = \mathrm{u}_0, \end{cases}$$

2. el modelo adjunto $\mathcal{M}_a$:

$$\begin{cases} z \in (z_0, z_{tropo}), \\ -\dfrac{dq}{dz} = \left[\dfrac{\partial M}{\partial \mathrm{u}}(\mathrm{u})\right]^T .q - \mathcal{H}^T(\mathcal{H}\mathrm{u} - \mathrm{u}_{obs}), \\ q(z_{tropo}) = 0, \end{cases}$$

3. y la fórmula para calcular el gradiente:

$$\frac{\partial(\mathcal{J}o\mathcal{M})}{\partial \mathrm{u}_0}(\mathrm{u}_0) = -q(z_0).$$

Este sistema se utiliza en estudios teóricos y numéricos.

4.1.4. Optimización numérica

Aspectos teóricos

La solución de (4.4) se calcula por un "algoritmo de descenso". Existen varios muy sofisticados, en ellos figura el Cuasi-Newtoniano. Dada su complejidad se presenta un algoritmo general señalando los lugares donde se pueden hacer mejoras. Un algoritmo de descenso tiene la forma siguiente:

0. Se elige una iteración inicial u_0^0 (el índice superior indica el número de la iteración del proceso de descenso) supuesta bastante cercana a la condición de salida óptima u_0^*.

El conocimiento de la condición de salida u_0^k a la iteración k permite estimar la condición inicial u_0^{k+1} como sigue:

1. Se calculan:
   - el problema directo integrando el modelo $\mathcal{M}$ a partir de $\mathrm{u}(z_0) = \mathrm{u}_0^k$ para conocer $\mathrm{u}^k(z)$ sobre el intervalo (z_0, z_{tropo}),
   - la desviación $\mathcal{J}(\mathrm{u}^k)$ con respecto a las observaciones.
   - el gradiente de $\mathcal{J}$ como en el párrafo 4.1.3 por ejemplo,
2. Se prueba el interés de u_0^k:
   a) si $\mathcal{J}(\mathrm{u}^k)$ es poco diferente de $\mathcal{J}(\mathrm{u}^{k-1})$, y/o $\nabla\mathcal{J}(\mathrm{u}^k)$ cerca de 0, el algoritmo se detiene con la estimación $\mathrm{u}_0^{\#} = \mathrm{u}_0^k$ de u_0^*,
   b) en otro caso, se construye una dirección de descenso $d^k \simeq \nabla\mathcal{J}(\mathrm{u}^k)$, se estima

$$\mathrm{u}_0^{k+1} = \mathrm{u}_0^k + \rho^k d^k,$$

   y se regresa al punto 1 con la condición inicial u_0^{k+1} para efectuar la iteración $k+1$ del algoritmo. El parámetro de descenso ρ^k se calcula de manera específica para cada algoritmo de optimización.

Cuando se trabaja con modelos no lineales, la función $\mathcal{J}o\mathcal{M}$ no es convexa es decir tiene varios mínimos locales donde el gradiente se anula. Entonces la elección del punto de inicio u_0^0 es muy importante. Si está demasiado lejos del óptimo u_0^*, es probable que el algoritmo de descenso encuentre un mínimo local del cual no pudiera salir para alcanzar u_0^*. Para remediar este problema, se puede elegir u_0^0 como lo sugiere el algoritmo y/o trabajar con funciones objetivo más sofisticadas (Ver el anexo A.3).

En la práctica, utilizamos el algoritmo Cuasi-Newton `L-BGS-B` propuesto por Lu, Nocedal y Zhu, 1995.Evaluamos $\mathcal{J}$ y su gradiente por medio de la rutina `simulad`. Sus argumentos son:

- el vector de control `x` $=\mathrm{u}_0$,
- la variable escalar `f`$=\mathcal{J}o\mathcal{M}(\mathrm{u}_0)$, y
- el (vector) gradiente `g` $= \nabla\mathcal{J}o\mathcal{M}(\mathrm{u}_0)$.

Se calculan usando la rutina `modelo` y la rutina adjunta `modelo_b` presentada en el párrafo 4.1.5

```
!-----------------------------------------------
!---   PliniAD/simulad.f   ---------------------
!-----------------------------------------------
       SUBROUTINE simulad(x,f,grad)
! Se utiliza para calcular JoM y su gradiente en el
! formalismo de L-BFGS-B
       INCLUDE 'datos.inc'
       REAL*8 x(4),f,grad(4)
       REAL*8 r0,u0,theta0,n0
       REAL*8 r0b,u0b,theta0b,n0b,fb
```

```
      r0=x(1);u0=x(2);theta0=x(3);n0=x(4);f=0
      u0b=0;theta0b=0;r0b=0;n0b=0;fb=1
      CALL modelo_b(r0,r0b,u0,u0b,theta0,theta0b,
     *              n0,n0b,f,fb)
      grad(1)=r0b;grad(2)=u0b;grad(3)=theta0b;
      grad(4)=n0b
      CALL modelo(r0, u0, theta0, n0, f)
      END
!-------------------------------------------------
```

Se tiene que descargar L-BGS-B desde la red para aprovechar la rutina `routines.f` en el directorio. No se tocará a esta carpeta. Sólo se cambia la rutina `driver.f` que maneja la optimización.

El programa pliniano que implementa el experimento gemelo figura en el directorio `www.lpmm.fr/charpentier/Libro/PliniAD`. Este ultimo contiene un `Makefile` y todos los archivos necesarios para reproducir el experimento gemelo exacto.

```
!-------------------------------------------------------
!---   PliniAD/programa_ad.f   -------------------------
!-------------------------------------------------------
      PROGRAM experimento_gemelo_exacto
      INCLUDE 'datos.inc'

      REAL*8 u0,theta0,r0,n0      !condiciones de salida
      REAL*8 j                    !discrepancy
      REAL*8 z
      INTEGER  i

! Condiciones de salida exactas
      r0=100; u0=300; theta0=1000; n0=0.03
      j=0.
! Creacion del archivo de datos sinteticos
      OPEN(1,file='observacion')
      CALL modelo(r0,u0,theta0,n0,j)
      CLOSE(1)
! Lectura de las observaciones
      OPEN(1,file='observacion')
      z = 0
      i = 1
      DO WHILE (z.LE.H1) !observacion bajo la tropopausa
        READ(1,*) z,robs(i),uobs(i),thetaobs(i)
        i = i+1
      ENDDO
      CLOSE(1)

! Experimento gemelo exacto
      r0=130; u0=250; theta0=900; n0=0.04
      CALL driver(r0, u0, theta0, n0)
      END
!-------------------------------------------------------


#-------------------------------------------------------
#---   PliniAD/Makefile   ------------------------------
#-------------------------------------------------------
FC = gfortran -g
CC=gcc
FFLAGS =  -C

all: clean adBuffer.o adStack.o pliniad.exe

adBuffer.o : adBuffer.f
        $(FC) -c adBuffer.f
adStack.o : adStack.c
        $(CC) -c adStack.c

#Experimento gemelo perfecto
TWINf= programa_ad.f modelo_b-all.f \
        modelo.f error.f funcion_M.f atmosfera.f \
        simulad.f driver.f routines.f
TWINo= programa_ad.o modelo_b-all.o \
        modelo.o error.o funcion_M.o atmosfera.o \
        simulad.o driver.o routines.o

pliniad.exe:
        $(FC) $(FFLAGS) -c $(TWINf)
        $(FC) adBuffer.o adStack.o $(TWINo) -o \
        pliniad.exe

clean:
        rm -f *.o pliniad.exe fort.* iterate.dat
#-------------------------------------------------------
```

```
!-------------------------------------------------------------
!---   PliniAD/driver.f   ------------------------------------
!-------------------------------------------------------------
      SUBROUTINE driver(r0,u0,theta0,n0)
! implementa el algoritmo de optimizacion L-BFGS-B
! [1] R. H. Byrd, P. Lu, J. Nocedal and C. Zhu, ``A limited
! memory algorithm for bound constrained optimization'',
! SIAM J. Scientific Computing 16 (1995), no. 5, pp. 1190--1208.
!
! Sus variables y parametros de trabajo son
      INCLUDE 'datos.inc'
      INTEGER   mmax, lenwa, nmax
      PARAMETER (mmax = 17, nmax = 4)
      PARAMETER (lenwa = 2*mmax*nmax +  4*nmax
     *                 + 11*mmax*mmax + 8*mmax)
      CHARACTER*60 task, csave
      LOGICAL lsave(4),boo
      INTEGER n, m, iprint,
     *        nbd(nmax), iwa(3*nmax), isave(44)
      REAL*8 f, x(nmax), l(nmax), u(nmax), grad(nmax),
     *       dsave(29), wa(lenwa)
      REAL*8 factr, pgtol

       real*8 u0,theta0,r0,n0
       real*8 u0b,theta0b,r0b,n0b,fb
       integer i,niter

! Parametros de la optimizacion
      iprint = 1      !iprint=0: no impresiones intermedias
                      !iprint=1: impresiones
      factr = 1d+1    !tolerencias en los criterios de parada
      pgtol = 0.
      n = 4           !dimension de la variable de control
      m = 5           !numero de corecciones guardadas
! Initializaciones
      x(1)=r0; x(2)=u0; x(3)=theta0; x(4)=n0
! espacio de control
      nbd(1)=2;l(1)=50; u(1)=200  !limites inf y sup para r0
      nbd(2)=2;l(2)=200;u(2)=450  !limites inf y sup para u0
      nbd(3)=2;l(3)=800;u(3)=1300 !limites inf y sup theta0
      nbd(4)=2;l(4)=.02;u(4)=.07  !limites inf y sup para n0

      task = 'START' !tarea inicial
! Principio del bucle iterativo de optimizacion
  111 continue
      write(*,*)(x(i),i=1,4)
      !llamada al codigo L-BFGS-B
      call setulb(n,m,x,l,u,nbd,f,grad,factr,pgtol,
     *     wa,iwa,task,iprint,csave,lsave,isave,dsave)

      if (task(1:2) .eq. 'FG') then
        !el optimizor quiere la funcion y su gradiente en x
        call simulad(x,f,grad)
        goto 111
      elseif (task(1:5) .eq. 'NEW_X') then
        !el optimizor produjo un nuevo iterado
        !los criterios de parada no son satisfechos
        !continua la optimizacion
        goto 111
      else
        !los criterios de parada son satisfechos o
        !el optimizor no encuentra una solucion y produce
        !un error
        if (iprint.le.-1.and.task(1:4).ne.'STOP')
     *    write(6,*) task
      endif
! Fin del bucle iterativo
! Impresion del resultado
      write(*,*) 'valor del control',(x(i),i=1,n)
      write(*,*) 'costo',f
      end
!-------------------------------------------------------------
```

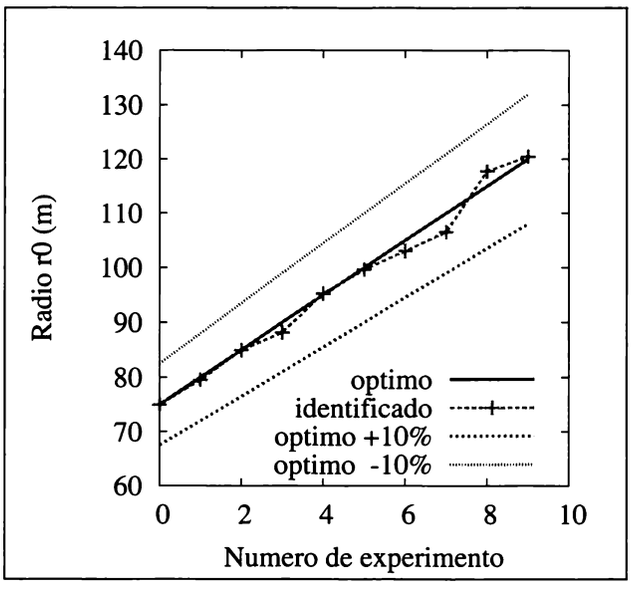

Fig. 4.1 Valores óptimos e identificados para el radio (m).

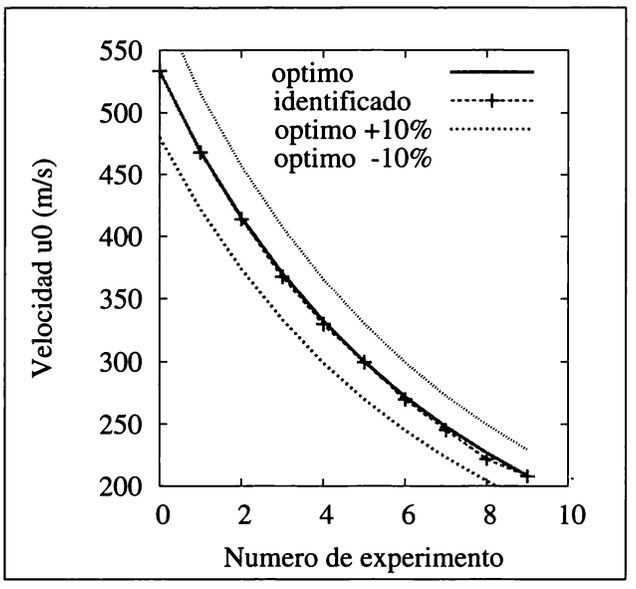

Fig. 4.2 Valores óptimos e identificados para la velocidad ($m.s^{-1}$).

Resultados numéricos

La pertinencia del proceso de asimilación de datos se puede estudiar por medio de experimentos gemelos.

En el experimento gemelo exacto propuesto en el directorio `PliniAD`, utilizamos condiciones de salida $\mathrm{u}_0^* = (r_0^*, u_0^*\theta_0^*, n_0^*) = (100m; 300m.s^{-1}; 1000K; 0{,}03)$ para construir datos de observacion sinteticos. El iterado inicial es $\mathrm{u}_0^0 = (130m, 250m.s^{-1}, 1000K, 0{,}04)$. El método de optimización converge en 61 iteraciones hasta el punto $\mathrm{u}_0^\# = (r_0^\#, u_0^\#, \theta_0^\#, n_0^\#)$ tal que:

$$\begin{cases} r_0^\# = 99{,}9999998m, \\ u_0^\# = 300{,}0000003m.s^{-1}, \\ \theta_0^\# = 1000{,}0000002K, \\ n_0^\# = 0{,}0299999992. \end{cases}$$

que corresponde casi exactamente al punto u_0^* eligido para producir los datos sinteticos. Nuestra herramienta de optimización es correcta.

En (Charpentier, 2008), el proceso de asimilación de datos se aplica a la identificación de las condiciones de salida en el curso de la fase eruptiva. Como utilizamos el modelo estacionario de Woods (1988),se consideran datos adquiridos en diversos tiempos para identificar las condiciones de salida correspondientes. Tal conocimiento presenta dos aspectos interesantes. Por un lado, estas condiciones pueden sustituirse por el uso de los modelos de cámara/conducto magmático (Folch y Felpeto, 2005; Textor et al., 2005) para proporcionar condiciones de salida al modelo pliniano. Por otro lado, estas condiciones podrían ser utilizadas como observaciones en un proceso de asimilación de datos implicando el modelo del conducto. Para representar una evolución temporal de la columna, el modelo fue ejecutado a partir de 10 conjuntos de entradas escogidas de tal manera que la masa $F_0 = \beta_0 u_0 r_0^2$ del flujo de salida sea constante. Considerando que la temperatura ($\theta_0 = 1000k$) y la fracción del gas ($n_0 = 0{,}03$) son constantes al nivel del cráter z_0, entonces una disminución de la velocidad límite se relaciona con un aumento en el radio de salida. Los otros parámetros están descritos en (Charpentier, 2008). Usamos los valores $r_0 = 80 + 5im$ ($i = 0, 9$) para figurar la erosión del conducto, para producir 10 conjuntos de observaciones. Los datos así generados son perturbados usando ruido blanco con desviaciones estándar iguales a $(15.m.s^{-1}, 5K, 30m)$ respectivamente. La observación, a través de la elección de operador de observación, se refiere a la parte troposférica de la columna. Según lo presentado en las figuras 4.1–4.4, el proceso de asimilación de datos permite la identificación de las condiciones de salida aúnque los conjuntos de datos sean bastante parecidos. Eso sugiere que podría ser posible seguir la evolu-

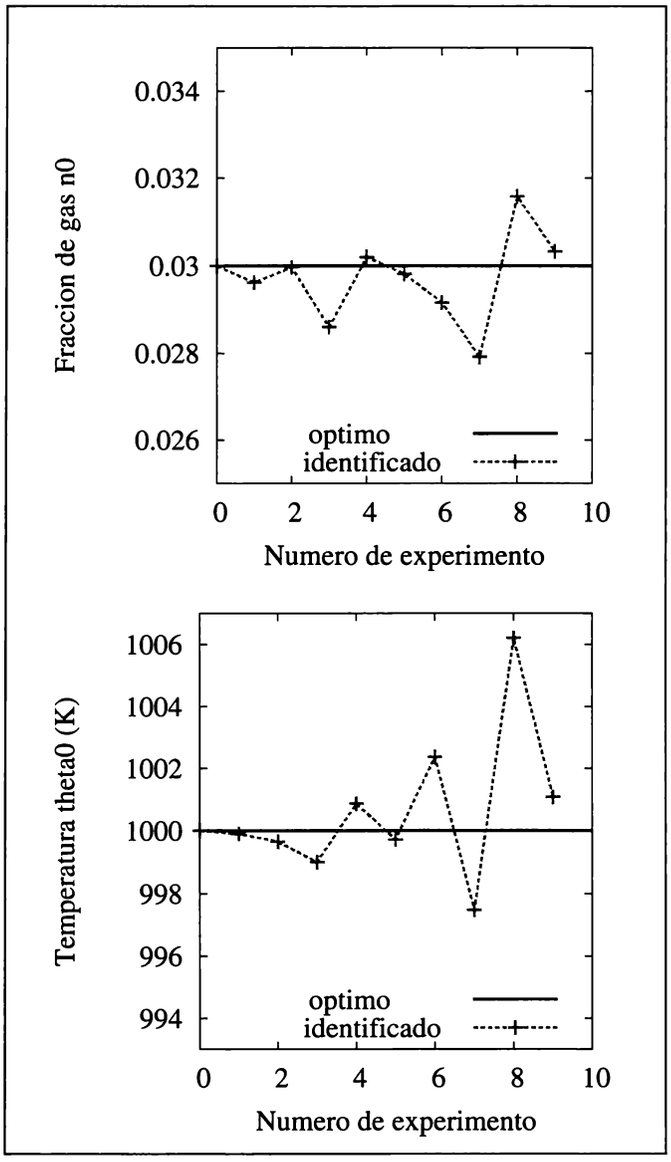

Fig. 4.3 Valores óptimos e identificados para la fracción de gas y la temperatura (K).

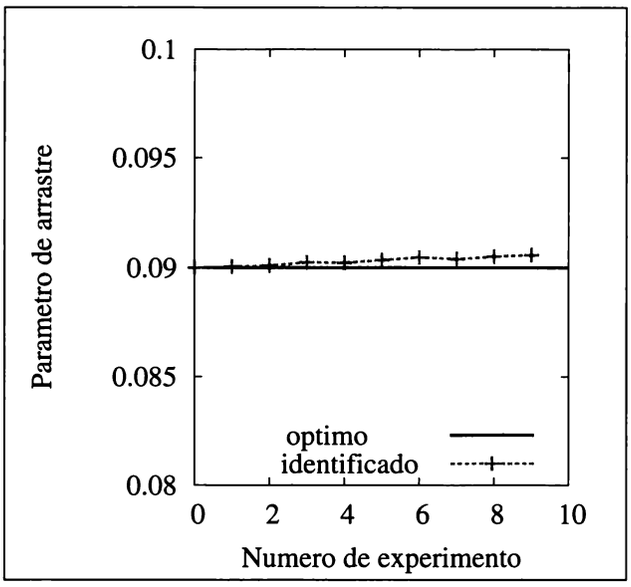

Fig. 4.4 Valores óptimos e identificados para el parámetro de arrastre.

ción de las condiciones de límite durante la erupción.

4.1.5. Código adjunto

Hemos presentado la diferenciación en modo adjunto como el método que permite calcular cuál sería la perturbación de entrada δ_e que tendría que ponerse para conseguir la perturbación de salida δ_s deseada. Escrito de una manera diferente, consiste en buscar el operador $\diamond$ que permite calcular la perturbación de entrada $\widehat{\delta_e}$ en función de la perturbación de salida $\widehat{\delta_s}$ como:

$$\widehat{\delta_e} = \diamond \widehat{\delta_s}.$$

Para esto, recordemos la propiedad de los productos escalares (4.5) en los espacios de Hilbert.

Se puede aprovechar esa propiedad para diferenciar en modo adjunto, ya que nuestros espacios discretos reales son espacios de Hilbert y nuestros Jacobianos son constantes con respecto a las variables derivadas. Así, el operador $\diamond$ es exactamente el Jacobiano transpuesto.

En el caso de la instrucción `z=x*y` que corresponde a la función matemática $f(x, y, z) = (x, y, x/y)$, evaluamos $\widehat{\delta_e} = J_f^T . \widehat{\delta_s}$ de la manera

siguiente:

$$\begin{aligned}
\widehat{\delta_e} &= J_f^T.\widehat{\delta_s} = \\
&= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{1}{y} & -\frac{x}{y^2} & 0 \end{pmatrix}^T \cdot \begin{pmatrix} \widehat{\delta_{xs}} \\ \widehat{\delta_{ys}} \\ \widehat{\delta_{zs}} \end{pmatrix} = \\
&= \begin{pmatrix} 1 & 0 & \frac{1}{y} \\ 0 & 1 & -\frac{x}{y^2} \\ 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \widehat{\delta_{xs}} \\ \widehat{\delta_{ys}} \\ \widehat{\delta_{zs}} \end{pmatrix} = \\
&= \begin{pmatrix} \widehat{\delta_{xs}} + \widehat{\delta_{zs}} y/y^2 \\ \widehat{\delta_{ys}} - \widehat{\delta_{zs}} x/y^2 \\ 0 \end{pmatrix}.
\end{aligned}$$

Aúnque aquí $\widehat{\delta_s}$ es finalmente un vector de entrada, conserva la `s` para señalar que estamos utilizando una dirección de perturbación de la variable de salida del modelo.

Cambiando los vectores de las variables matemáticas $\widehat{\delta_s}$ y $\widehat{\delta_e}$ en las variables informáticas `xb`, `yb` y `zb` (la `b` se utiliza para indicar la diferenciación en modo adjunto (backward)), se deducen las instrucciones adjuntas (derivadas en modo adjunto):

```
xb=xb+zb*y/y**2
yb=yb-zb*x/y**2
zb=0
```

Todas contienen información. De la misma manera que para la transposición del producto de operadores lineales, las instrucciones del código son diferenciadas en el orden opuesto. En modo adjunto se tiene que diferenciar y evaluar la ultima instrucción del código original antes que todas las otras. Obviamente, la evaluación de las derivadas necesita una primera ejecución completa del código para conocer los últimos elementos de la trayectoria. El manejo de la trayectoria es el aspecto difícil de la diferenciación en modo adjunto. No lo explicaremos con detalles ya que las herramientas lo proporcionan bien.

Sobrecarga de los operadores

Una herramienta como `Adol-C` (Griewanket al., 1996)permite diferenciar en modo adjunto los códigos escritos en `C` o `C++`. Para simplificar, se puede ver la ejecución del código sobrecargado en 2 etapas:

1. se recoge el código guardando los operadores en columnas, las variables y su valor actual, los índices de tablas, los resultados de pruebas, es decir todo lo que forma la trayectoria de evaluación del código,
2. se calculan las derivadas operación por operación (del último operador guardado hasta el primero).

Requiere mucho más trabajo escribir una biblioteca para esta diferenciación en modo adjunto.

Fuente a Fuente

Se utiliza por ejemplo la herramienta `Tapenade` pidiéndole una diferenciación en modo adjunto. Se nota que la trayectoria está manejada por pedazos. Cada rutina adjunta se presenta en 3 partes:

1. definiciones de las variables e inicialización de ciertas variables adjuntas,
2. cálculo de la trayectoria cuyas valores se guardan en variables intermedias,
3. evaluación de las variables adjuntas usando la trayectoria previamente guardada.

Se obtiene el código adjunto siguiente (ver el archivo `modelo_b-all.f` en la página Internet `www.lpmm.fr/charpentier/Libro/PliniAD`). Se tendria que verificarlo haciendo una prueba de Taylor tal como en el capítulo 3 (ejercicio).

```
C        Generated by TAPENADE     (INRIA, Tropics team)
C  Tapenade 2.2.4 (r2308) - 03/04/2008 10:04
C
C  Differentiation of modelo in reverse (adjoint) mode:
C   gradient, with respect to input variables: costo theta0 n0
C                u0 r0
C   of linear combination of output variables: costo theta0 n0
C                u0 r0
C-----------------------------------------------------------
C---   Pliniano/modelo.f   ---------------------------------
C-----------------------------------------------------------
      SUBROUTINE MODELO_B(r0, r0b, u0, u0b, theta0, theta0b, n0, n0b,
     +                    costo, costob)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
C Variables de salida del crater
Centradas del modelo
      REAL*8 r0, u0, theta0, n0
      REAL*8 r0b, u0b, theta0b, n0b
      REAL*8 beta0, e0, f0
      REAL*8 beta0b, e0b, f0b
C Costo
      REAL*8 costo
      REAL*8 costob
C Variables de esta DO
      REAL*8 u, theta, r, n, beta, cp, rg, e, f
      REAL*8 ub, thetab, rb, nb, betab, cpb, rgb, eb, fb
C Variables intermedias por RK4
Ccontiene la solucion del modelo yn
      REAL*8 yn(4)
      REAL*8 ynb(4)
Cecuacion (2.20)
      REAL*8 m1(4), m2(4), m3(4), m4(4)
      REAL*8 m1b(4), m2b(4), m3b(4), m4b(4)
Cvar. intermedias
      REAL*8 yy(4)
      REAL*8 yyb(4)
      INTEGER zz
      REAL*8 dzz
C Variables atmosfericas
      REAL*8 t, p, alfa
      INTEGER i, zn
      DOUBLE PRECISION temp
      DOUBLE PRECISION temp0
      REAL*8 temp1
      REAL*8 temp2
      REAL*8 temp3
      REAL*8 temp0b
      DOUBLE PRECISION tempb0
      REAL*8 temp3b
      DOUBLE PRECISION tempb
      REAL*8 temp0b4
      REAL*8 temp0b3
      REAL*8 temp0b2
      DOUBLE PRECISION temp0b1
      REAL*8 temp0b0
      REAL*8 temp2b
      INTEGER ad_count
      INTEGER ii1
      INTRINSIC SQRT
      INTEGER i0
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
C
C Initializaciones
      dzz = dz*1.d0
      rg = rg0
      CALL ATMOSFERA(z0, t, p, alfa)
      beta0 = 1.d0/((1.d0-n0)/sigma+n0*rg*theta0/p)
      zn = z0 + dz
C Cambio de variables
      f0 = u0*r0*r0*beta0
      e0 = theta0*cp0
      yn(1) = f0
      yn(2) = u0
      yn(3) = e0
      yn(4) = n0
      ad_count = 0
C Desarollo de la columna
C Solucion por Runge Kutta de orden 4
      DO WHILE (yn(2) .GE. velmin)
        CALL PUSHREAL8ARRAY(m1, 4)
C Calculos (2.19)
        CALL FUNCION_M(zn, yn, n0, f0, m1)
        DO i=1,4
```

```
          CALL PUSHREAL8(yy(i))
          yy(i) = yn(i) + dzz*m1(i)*0.5d0
        ENDDO
        CALL PUSHREAL8ARRAY(m2, 4)
        CALL FUNCION_M(zn, yy, n0, f0, m2)
        DO i=1,4
          CALL PUSHREAL8(yy(i))
          yy(i) = yn(i) + dzz*m2(i)*0.5d0
        ENDDO
        CALL PUSHREAL8ARRAY(m3, 4)
        CALL FUNCION_M(zn, yy, n0, f0, m3)
        DO i=1,4
          CALL PUSHREAL8(yy(i))
          yy(i) = yn(i) + dzz*m3(i)
        ENDDO
        CALL PUSHREAL8ARRAY(m4, 4)
        CALL FUNCION_M(zn, yy, n0, f0, m4)
C Ecuacion (2.18)
Cyn
        DO i=1,4
          CALL PUSHREAL8(yn(i))
          yn(i) = yn(i) + dzz*(m1(i)+2.d0*m2(i)+2.d0*m3(i)+m4(i))/6.d0
        ENDDO
        CALL PUSHREAL8(f)
C Cambio de variables inverso
        f = yn(1)
        CALL PUSHREAL8(u)
        u = yn(2)
        CALL PUSHREAL8(e)
        e = yn(3)
        CALL PUSHREAL8(n)
        n = yn(4)
        CALL PUSHREAL8(cp)
C Relacions funcionales
        cp = ca + (cp0-ca)*((1.d0-n)/(1.d0-n0))
        CALL PUSHREAL8(rg)
        rg = ra + (rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
        CALL PUSHREAL8(theta)
        theta = e/cp
        CALL PUSHREAL8(beta)
        beta = 1.d0/((1.d0-n)/sigma+n*rg*theta/p)
        CALL PUSHREAL8(r)
        r = SQRT(f/(u*beta))
        CALL PUSHINTEGER4(zn)
C evaluacion del error
C Escritura del archivo de resultados
C Incremento de la altura
        zn = zn + dz
        ad_count = ad_count + 1
      ENDDO
      CALL PUSHINTEGER4(ad_count)
      f0b = 0.0
      DO ii1=1,4
        m1b(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m2b(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m3b(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        m4b(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        ynb(ii1) = 0.0
      ENDDO
      DO ii1=1,4
        yyb(ii1) = 0.0
      ENDDO
      CALL POPINTEGER4(ad_count)
      DO i0=1,ad_count
        CALL POPINTEGER4(zn)
        CALL ERROR_B(zn, r, rb, u, ub, theta, thetab, costo, costob)
        CALL POPREAL8(r)
        temp2 = u*beta
        temp3 = f/temp2
        IF (temp3 .EQ. 0.0) THEN
          temp3b = 0.0
        ELSE
          temp3b = rb/(2.0*SQRT(temp3)*temp2)
        END IF
        temp2b = -(temp3*temp3b)
        fb = temp3b
        ub = ub + beta*temp2b
        betab = u*temp2b
        CALL POPREAL8(beta)
        temp1 = theta/p
        temp0 = (-n+1.d0)/sigma + n*rg*temp1
        temp0b1 = -(betab/temp0**2)
        rgb = temp1*n*temp0b1
        thetab = thetab + n*rg*temp0b1/p
        cpb = -(e*thetab/cp**2)
        temp0b2 = (rg0-ra)*rgb/((1.d0-n0)*n)
        temp0b3 = -(n0*(1.d0-n)*temp0b2/((1.d0-n0)*n))
        temp0b4 = (cp0-ca)*cpb/(1.d0-n0)
        nb = (1.d0-n0)*temp0b3 - n0*temp0b2 - temp0b4 + (temp1*rg-1.0/
     +    sigma)*temp0b1
        CALL POPREAL8(theta)
        eb = thetab/cp
        CALL POPREAL8(rg)
        n0b = n0b + (1.d0-n)*temp0b4/(1.d0-n0) - n*temp0b3 + (1.d0-n)*
     +    temp0b2
        CALL POPREAL8(cp)
```

```
  CALL POPREAL8(n)
  ynb(4) = ynb(4) + nb
  CALL POPREAL8(e)
  ynb(3) = ynb(3) + eb
  CALL POPREAL8(u)
  ynb(2) = ynb(2) + ub
  CALL POPREAL8(f)
  ynb(1) = ynb(1) + fb
  DO i=4,1,-1
    CALL POPREAL8(yn(i))
    temp0b0 = dzz*ynb(i)/6.d0
    m1b(i) = m1b(i) + temp0b0
    m2b(i) = m2b(i) + 2.d0*temp0b0
    m3b(i) = m3b(i) + 2.d0*temp0b0
    m4b(i) = m4b(i) + temp0b0
  ENDDO
  CALL POPREAL8ARRAY(m4, 4)
  CALL FUNCION_M_B(zn, yy, yyb, n0, n0b, f0, f0b, m4, m4b)
  DO i=4,1,-1
    CALL POPREAL8(yy(i))
    ynb(i) = ynb(i) + yyb(i)
    m3b(i) = m3b(i) + dzz*yyb(i)
    yyb(i) = 0.0
  ENDDO
  CALL POPREAL8ARRAY(m3, 4)
  CALL FUNCION_M_B(zn, yy, yyb, n0, n0b, f0, f0b, m3, m3b)
  DO i=4,1,-1
    CALL POPREAL8(yy(i))
    ynb(i) = ynb(i) + yyb(i)
    m2b(i) = m2b(i) + dzz*0.5d0*yyb(i)
    yyb(i) = 0.0
  ENDDO
  CALL POPREAL8ARRAY(m2, 4)
  CALL FUNCION_M_B(zn, yy, yyb, n0, n0b, f0, f0b, m2, m2b)
  DO i=4,1,-1
    CALL POPREAL8(yy(i))
    ynb(i) = ynb(i) + yyb(i)
    m1b(i) = m1b(i) + dzz*0.5d0*yyb(i)
    yyb(i) = 0.0
  ENDDO
  CALL POPREAL8ARRAY(m1, 4)
  CALL FUNCION_M_B(zn, yn, ynb, n0, n0b, f0, f0b, m1, m1b)
ENDDO
n0b = n0b + ynb(4)
ynb(4) = 0.0
e0b = ynb(3)
ynb(3) = 0.0
u0b = u0b + ynb(2)
ynb(2) = 0.0
f0b = f0b + ynb(1)
temp0b = r0**2*f0b
beta0b = u0*temp0b
      temp = (-n0+1.d0)/sigma + rg*n0*theta0/p
      tempb = -(beta0b/temp**2)
      tempb0 = rg*tempb/p
      theta0b = theta0b + n0*tempb0 + cp0*e0b
      r0b = r0b + u0*beta0*2*r0*f0b
      u0b = u0b + beta0*temp0b
      n0b = n0b + theta0*tempb0 - tempb/sigma
      costob = 0.0
 111  FORMAT(i5,10f12.6)
      END
C--------------------------------------------------------------

C  Differentiation of error in reverse (adjoint) mode:
C   gradient, with respect to input variables: costo r u theta
C   of linear combination of output variables: costo
C--------------------------------------------------------------
C---   Pliniano/error.f   ------------------------------------
C--------------------------------------------------------------
      SUBROUTINE ERROR_B(z, r, rb, u, ub, theta, thetab, costo, costob)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
      REAL*8 r, u, theta, costo
      REAL*8 rb, ub, thetab, costob
      INTEGER z, i
      INTRINSIC MOD
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
C
      IF (MOD(z, 500) .EQ. 0 .AND. z .LE. h1) THEN
        i = (z-z0)/dz
        rb = 2*(r-robs(i))*costob
        ub = 2*(u-uobs(i))*costob
        thetab = 2*(theta-thetaobs(i))*costob
      ELSE
```

```
        rb = 0.0
        ub = 0.0
        thetab = 0.0
      END IF
      END
C-----------------------------------------------------------

C  Differentiation of funcion_m in reverse (adjoint) mode:
C   gradient, with respect to input variables: m f0 y n0
C   of linear combination of output variables: m f0 y n0
C-----------------------------------------------------------
C---   Pliniano/funcion_M.f   -----------------------------
C-----------------------------------------------------------
      SUBROUTINE FUNCION_M_B(z, y, yb, n0, n0b, f0, f0b, m, mb)
      IMPLICIT NONE
C Temperatura (T0) y presion (P0) al nivel del mar
C Constantes del aire (Ca,Ra)
C Alturas de la tropausa (h1), de la estratosfera baja (h2)
C Razones de lapso (l0,l1,l2), gravedad (g)
      REAL*8 t0, p0, ca, ra, h1, h2, l0, l1, l2, g
C Altura del crater (z0)
C Datos de la erupcion (Cp0,Rg0,sigma)
C Parametro de arrastre
      REAL*8 cp0, rg0, sigma, eps
      INTEGER z0, dz
C Paso de calculo (dz), limite de parada (velmin)
      REAL*8 velmin
C Observaciones
      REAL*8 uobs(6000), thetaobs(6000), robs(6000)
      COMMON /obs/ uobs, thetaobs, robs
C Variables atmosfericas,
      REAL*8 t, p, alfa
      INTEGER z
C Variables de RK4
      REAL*8 m(4), y(4)
      REAL*8 mb(4), yb(4)
C Variables intermedias
      REAL*8 c, d, cp, rg, theta, r, beta
      REAL*8 cb, db, cpb, rgb, thetab, rb, betab
      REAL*8 f, u, e, n
      REAL*8 fb, ub, eb, nb
      REAL*8 n0, f0
      REAL*8 n0b, f0b
      DOUBLE PRECISION temp
      REAL*8 temp0
      REAL*8 temp1
      REAL*8 temp2
      REAL*8 temp3
      REAL*8 tempb2
      REAL*8 tempb1
      REAL*8 tempb0
      REAL*8 temp3b
      DOUBLE PRECISION tempb
      REAL*8 temp2b
      REAL*8 temp3b1
      REAL*8 temp3b0
      REAL*8 temp1b
      INTRINSIC SQRT
      DATA t0, p0, ca, ra /288.15, 101320., 998., 287.54/
      DATA h1, h2, g /14000., 20000., 9.81/
      DATA l0, l1, l2 /.0065, .0, -.002/
      DATA cp0, rg0, sigma, eps /1617., 462., 1617., 0.09/
      DATA z0, dz /5425, 5/
      DATA velmin /10./
C
C Atmosfera estandar a la altura z
      CALL ATMOSFERA(z, t, p, alfa)
C Cambio de variables
      f = y(1)
      u = y(2)
      e = y(3)
      n = y(4)
C Relaciones funcionales (2.15)
      cp = ca + (cp0-ca)*((1.d0-n)/(1.d0-n0))
      rg = ra + (rg0-ra)*n0/(1.d0-n0)*(1.d0-n)/n
      theta = e/cp
      beta = 1.d0/((1.d0-n)/sigma+n*rg*theta/p)
      r = SQRT(f/(u*beta))
C Variables intermedias
      d = (alfa-beta)/beta
      c = 2.d0*u*eps*r*alfa/f
C Ecuaciones (2.20)
      m(1) = f*c
      m(2) = g*d/u - c*u
      temp3b = f0*c*mb(4)/f
      temp3 = (-n0+1.d0)/f
      f0b = f0b + temp3*c*mb(4)
      cb = temp3*f0*mb(4)
      mb(4) = 0.0
      cb = cb + (ca*t-e-.5d0*u**2)*mb(3)
      eb = -(c*mb(3))
      ub = (-m(2)-c*.5d0*2*u)*mb(3)
      mb(2) = mb(2) - u*mb(3)
      mb(3) = 0.0
      temp3b0 = g*mb(2)/u
      db = temp3b0
      ub = ub - c*mb(2) - d*temp3b0/u
      cb = cb - u*mb(2)
      mb(2) = 0.0
      cb = cb + f*mb(1)
      temp3b1 = eps*2.d0*alfa*cb/f
      rb = u*temp3b1
      temp1 = u*beta
      temp2 = f/temp1
```

```
      IF (temp2 .EQ. 0.0) THEN
        temp2b = 0.0
      ELSE
        temp2b = rb/(2.0*SQRT(temp2)*temp1)
      END IF
      temp1b = -(temp2*temp2b)
      betab = u*temp1b + (-((alfa-beta)/beta**2)-1/beta)*db
      temp0 = theta/p
      temp = (-n+1.d0)/sigma + n*rg*temp0
      tempb = -(betab/temp**2)
      rgb = temp0*n*tempb
      thetab = n*rg*tempb/p
      cpb = -(e*thetab/cp**2)
      tempb0 = (rg0-ra)*rgb/((1.d0-n0)*n)
      tempb1 = -(n0*(1.d0-n)*tempb0/((1.d0-n0)*n))
      tempb2 = (cp0-ca)*cpb/(1.d0-n0)
      n0b = n0b + (1.d0-n)*tempb0 - n*tempb1 + (1.d0-n)*tempb2/(1.d0-n0)
     +  - temp3b
      fb = c*mb(1) + temp2b - u*r*temp3b1/f - temp3*temp3b
      mb(1) = 0.0
      ub = ub + beta*temp1b + r*temp3b1
      nb = (1.d0-n0)*tempb1 - n0*tempb0 - tempb2 + (temp0*rg-1.0/sigma)*
     +  tempb
      eb = eb + thetab/cp
      yb(4) = yb(4) + nb
      yb(3) = yb(3) + eb
      yb(2) = yb(2) + ub
      yb(1) = yb(1) + fb
      END
C-----------------------------------------------------------
```

4.2. Identificación de la función de arrastre con datos actuales

Tanto la modelación como la observación de las columnas eruptivas conllevan errores. Por un lado, el modelo unidimensional que estudiamos puede aparecer inadecuado para el tratamiento de los datos recogidos en una columna turbulenta tridimensional afectada por el viento. Aunque la modelación puede ser mejorada por medio de la función de arrastre (Morton, Taylor y Turner, 1956),o del uso de una parametrización del viento, el modelo es aún muy simple. El uso de una atmósfera estándar es también una aproximación cruda en el estudio de una erupción real. Por otro lado, la colección y la interpretación de los datos pueden variar de un dispositivo a otro. La sobre o subestimación de los datos son consecuencias bien conocidas de las técnicas de medición. Por ejemplo, la altura de la columna evaluada usando la temperatura del brillo (Holasek, Self y Woods, 1996) depende de las condiciones atmosféricas. En la práctica, los datos reales se pueden tratar de la siguiente manera:

- en un preproceso: calculando la columna vertical equivalente, quitando el ruido de los datos, corrigiendo cualquier sobre o subestimación,
- durante el proceso de asimilación de datos, por medio de parametrizaciones cuyos parámetros serían parte de la variable de control.

Según Sparks et al. (1997), la columna y el aire ambiente tienen que ser ambos observados. Se nota que las variables atmosféricas pueden también estimarse por simulaciones meteorológicas.

4.2.1. Contexto experimental

En la erupción del 7 de agosto del año 1980, cuatro co-ignimbritas fueron generadas por flujos piroclásticos sobre el Mount St. Helens. Calder, Sparks y Woods (1997) los describieron y midieron a partir del trabajo de Hoblitt (1986). Usando los perfiles de crecimiento de estas columnas, estimaron velocidades y los radios verticales para las dos primeras (A y B), y solamente las velocidades para las otras dos (C y D). En las columnas A y D el flujo primero acelera antes de decelerar. Al contrario, los flujos de B y C aceleran durante todo el período de observación (véase Fig. 4.6). Con respecto a los resultados experimentales de Turner (1962), los datos de velocidad y de radio de una pluma estable se pueden deducir a partir de la velocidad y del radio de una pluma en formación aplicando cambios de escala de 1/0.6 y 0.8 respectivamente. Se dedujeron tazas de expansión, de 11.3° para la pluma A y 12° para la pluma B. Otras condiciones de salida fueron estimadas. El artículo de Calder, Sparks, y Woods (1997)contiene una comparación entre los datos reales y dos modelos, a saber un modelo de la pluma de la co-ignimbrita estacionaria (Woods y Wohletz, 1991)y un modelo discreto de nube co-ignimbrítica (Woods y Kienle, 1994).

Dentro de su modelo de columna co-ignimbrítica, probaron varias funciones de arrastre incluyendo $\varepsilon(z) = 0$, $\varepsilon(z) = 0{,}09$ que corresponde al arrastre completo de (Morton, Taylor y Turner, 1956),y una función que escala el arrastre con respecto a la altura, es decir:

$$\begin{cases} \varepsilon(z) = \lambda z / r \lambda z < r, \\ \varepsilon(z) = 0{,}09 \quad para \quad \lambda z \geq r. \end{cases}$$

Ningunas de estas funciones permiten una predicción de la taza de expansión de la columna co-ignimbrítica real. Se puede explicar teóricamente puesto que (ver Ishimine (2006)por ejemplo) el coeficiente $\varepsilon_c (= \varepsilon(z))$ del arrastre puede deducirse del gradiente k del radio de la columna como:

$$\varepsilon_c = \frac{5}{6} k = \frac{5}{6} \tan(s), \tag{4.11}$$

donde $s = \arctan(k)$ es la taza de expansión. De acuerdo con lo usualmente encontrado en la literatura, un gradiente $k = 0{,}108$, que conduce al $\varepsilon(z) = 0{,}09$ clásico, corresponde a una taza de $s = 6{,}16°$ que está bastante cerca al ángulo de 6.5° presentado en (Calder, Sparks y Woods, 1997).Usando la fórmula (4.11), uno deduce que, para estas plumas co-ignimbríticas, un coeficiente ε_c del orden de 0.17 sería más apropiado para calcular radios con una taza de expansión cercana a la real.

4.2.2. Modelación inversa

El objetivo de este experimento es enseñar como se pueden identificar funciones de arrastre (Charpentier, 2008) tales que las columnas

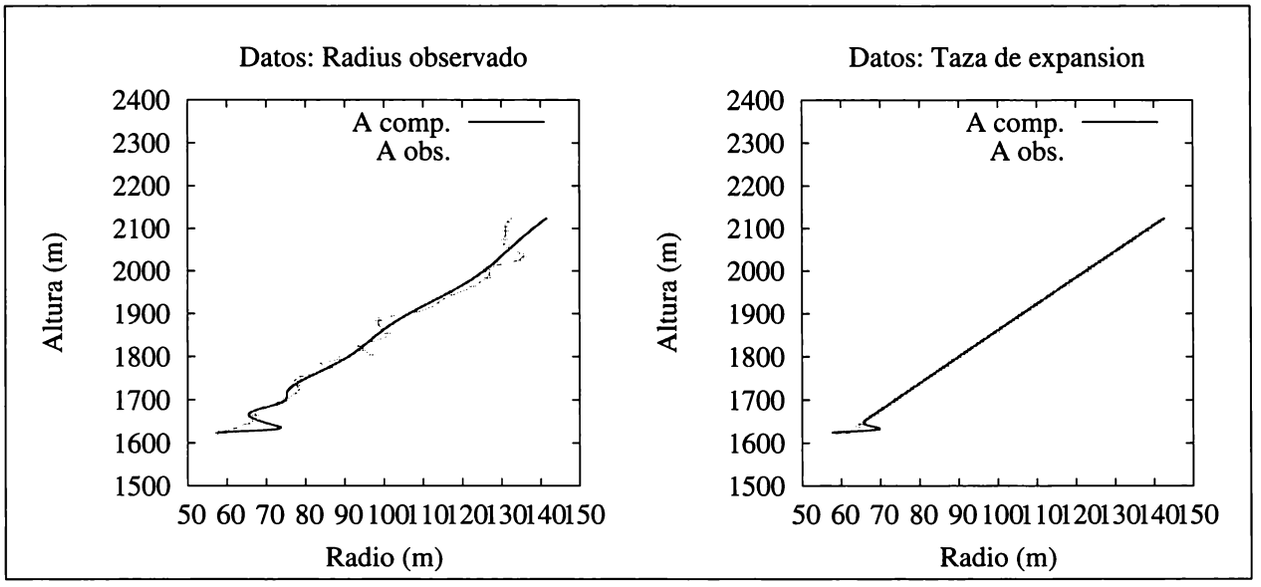

Fig. 4.5 Radios identificados y observados (después del cambio de escala) para la pluma co-ignimbrítica A.

numéricas tengan tazas de expansión dadas. Los elementos de este problema inverso son:

1. el modelo pliniano $\mathcal{M} = \mathcal{P}o\mathcal{T}$ cuyas entradas son:
   - las condiciones de salida (o más bien de la base de la columna) supuestas iguales a $u_0 = 3m.s^{-1}$, $\theta_0 = 600K$ y $n_0 = 0{,}6$ para las columnas A y B. El radio basal depende de la columna ($r_0 = 60$ para A, y $r_0 = 100$ para B). No se van identificar las condiciones de salida durante la optimización.
   - la función de arrastre $\varepsilon(z)$ se escribe como esplines cúbicos definidos por medio de 14 puntos de control $\{\varepsilon_j\}_{j=1,14}$ en las alturas 1, 10, $\{20i\}_{i=1,4}$, $\{100i\}_{i=1,8}$ (en m).
2. la variable de control $\{\varepsilon_j\}_{j=1,14}$ pertenece al espacio de control $\mathcal{E}$ tal que $\{\varepsilon_j\}_{j=1,14}$,
3. la función objetivo discreta puede ser escrita como:

$$\mathcal{J}_r = \sum_{p=0}^{46} |r(z_p) - r^{obs}(z_p)|^2, \qquad (4.12)$$

considerando 47 alturas $z_p = \{40 + 10i\}_{i=0,46}$. Esta función no considera la base de la columna porque, según lo descrito en (Calder, Sparks y Woods, 1997),el radio de la pluma se contrae. Tal comportamiento ocurre en el cómputo porque no se ingiere bastante aire en la base de la pluma. A mayor velocidad de la mezcla de cenizas y gas, más aire se captura lo que permite una expansión de la columna.

4. el problema inverso:

$$\textit{Buscar } \{\varepsilon_j^*\}_{j=1,14} \textit{ tal que } \min_{\{\varepsilon_j\}_{j=1,14}} \mathcal{J}o\mathcal{M}(\mathrm{u}_0, \{\varepsilon_j\}) = \mathcal{J}o\mathcal{M}(\mathrm{u}_0, \{\varepsilon_j^*\}),$$

donde $\{\varepsilon_j^*\}_{j=1,14}$ es el control óptimo y $\mathrm{u}_0 = (r_0, u_0, \theta_0, n_0)$ son condiciones de salida conocidas,

5. el método de asimilación variacional de datos. Buscamos la solución $\{\varepsilon_j^*\}_j$ donde el gradiente $\nabla\mathcal{J}o\mathcal{M}$ es cero, es decir:

$$\textit{Buscar } \{\varepsilon_j^*\}_{j=1,14} \textit{ tal que } \nabla\mathcal{J}o\mathcal{M}(r_0, u_0, \theta_0, n_0, \{\varepsilon_j^*\}_j) = 0.$$

En consecuencia, se necesita diferenciar el modelo con respecto a los parámetros de la función de arrastre. No presentamos los códigos relativos a este ejemplo.

Modelo adjunto

La variable de control $\{\varepsilon_j\}_j$ es un vector con 14 componentes. Se puede todavía pensar

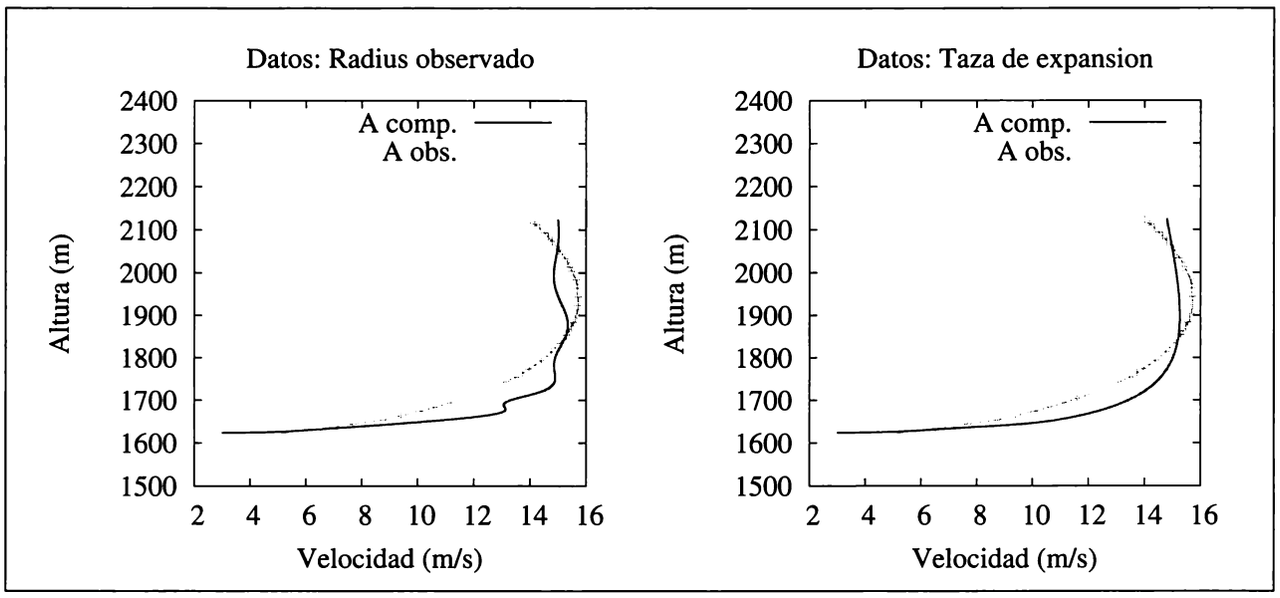

Fig. 4.6 Velocidades identificadas y observadas (después del cambio de escala) para la pluma co-ignimbrítica A.

en el uso de la diferenciación en modo lineal tangente, pero por la complejidad algorítmica lineal debida al número de direcciones (en este caso 14), el costo de tal cálculo se vuelve caro.

La técnica Lagrangiana se applica (ejercicio). Cuando uno quiere identificar parámetros y condiciones de salida al mismo tiempo, la teoría de la asimilación variacional de datos permite calcular el gradiente de $\mathcal{J}o\mathcal{M}$ como:

$$\begin{cases} \dfrac{\partial(\mathcal{J}o\mathcal{M})}{\partial \mathrm{u}_0}(\mathrm{u}_0, \{\varepsilon_j\}_j) = -q(0), \\ \dfrac{\partial(\mathcal{J}o\mathcal{M})}{\partial \{\varepsilon_j\}_j}(\mathrm{u}_0, \{\varepsilon_j\}_j) = \\ \quad = -\int_{z_0}^{z_{top}} \left[\frac{\partial M}{\partial \{\varepsilon_j\}_j}(\mathrm{u}_0, \{\varepsilon_j\}_j)\right]^T .q, \end{cases}$$

donde q es solución del problema adjunto $(\mathcal{M}_a)$:

$$\begin{cases} \text{en } (z_0, z_{top}), \\ -\dfrac{dq}{dt} = \left[\dfrac{\partial M}{\partial \mathrm{u}_0}(\mathrm{u}_0, \{\varepsilon_j\}_j)\right]^T .q + \\ \qquad + \left[\dfrac{\partial M}{\partial \{\varepsilon_j\}_j}(\mathrm{u}_0, \{\varepsilon_j\}_j)\right]^T .q - \\ \qquad - \sum_p (r(z_p) - r^{obs}(z_p))^2, \\ q(z_{top}) = 0. \end{cases}$$

Se nota que el sistema anterior es un sistema retrógrado que se integra de la cima de la

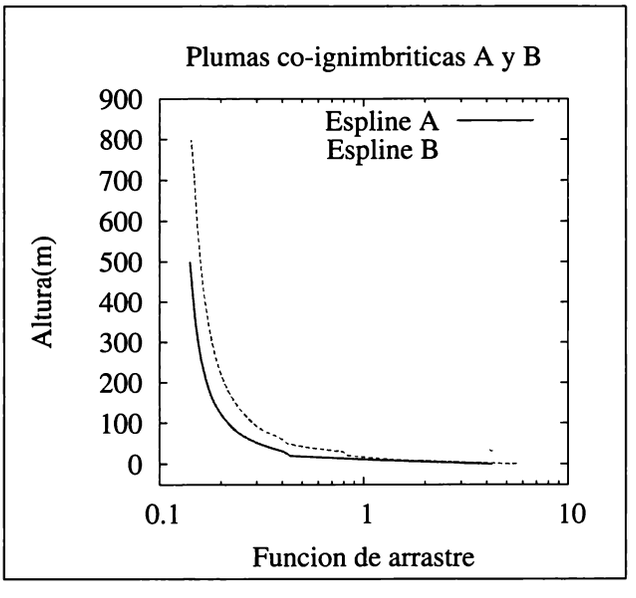

Fig. 4.7 Funciones de arrastre identificadas para las plumas co-ignimbríticas A y B.

columna z_{top} hasta z_0. Es posible hacerlo porque, al contrario del sistema (4.1), el término derecho es lineal en la variable adjunta q.

4.2.3. Resultados numéricos

Experimentos de asimilación de datos son realizados usando observaciones del radio y observaciones deducidas de la taza de expansión (son datos de radio sin ruido). Los dos conjuntos de datos están corregidos por el cambio de escala de Turner (1962).Se aplica una interpolación con esplines cúbicos para deducir un acercamiento regular del radio a lo largo de la columna. La figura 4.5 enseña una comparación entre los radios calculados y los radios observados para la pluma A. En los dos casos, las tazas de expansión están bien predichas. En particular, se nota que las curvas son casi iguales cuando se usan datos de tazas de expansión. Se podría también obtener tal resultado con datos de radio agregando el término de penalización $\nu|\Delta r|^2$ ($\nu > 0$) a la función objetivo (4.12) porque este último se volvería 0 para un radio que crece linealmente. Las plumas calculadas todavía presentan una contracción al nivel del cráter (z_0 a $z_0 + 40$), pero la funciones de arrastre que fueron identificadas permiten reducir su amplitud. Las velocidades identificadas (Fig. 4.6) presentan una buena adecuación con respecto a las velocidades observadas.

Estos experimentos permiten validar el proceso de asimilación de datos reales, pero no son suficientes para deducir elementos de vulcanología ya que no consideramos bastantes juegos de datos.

5. Conclusiones

Este documento presenta las etapas de construcción de una herramienta de optimización que permite introducir datos en la modelación. Se ha utilizado el caso de las columnas plinianas como ilustración. Sin embargo, los métodos son generales y fueron utilizados desde hace unos 20 años en varios campos de la Investigación en Ciencias de la Tierra.

Para darle un tono general hemos escrito cada capítulo introduciendo paulatinamente los elementos de física, informática y matemática necesarios para la modelación, el análisis de sensibilidad y la optimización. En cada uno de estos pasos se ha expuesto la metodología general y su aplicación al caso del conjunto de ecuaciones que constituyen el modelo unidimensional de Woods. Es de esperarse que su aplicación a otros problemas de Ciencias de la Tierra resulte claro a través del ejemplo de las columnas eruptivas plinianas

Hay que reconocer que las columnas plinianas son fenómenos puntuales cuyas características son más conocidas por los depósitos que han dejado que por las imágenes de video de sus columnas. Por consiguiente es necesario encaminar estos métodos para la modelación de columnas en términos de los datos obtenidos de los depósitos, más abundantes y más duraderos que las columnas en sí. En este sentido, en la presente obra se han mostrado los métodos utilizados usando el modelo unidimensional de Woods (1998);sin embargo se continúa trabajando en modelos tridimensionales (Costa, Macedonio y Folch, 2006) que modelan los depósitos de caída. De esta manera se podrán utilizar los métodos que introdujimos en la presente obra para la inversión de mapas de isopacas e isopletas.

A. Apéndices

A.1. Cómo generar ruido blanco

Para agregar ruido blanco, se puede utilizar la rutina `gausspolar`. Posteriormente se construyen las variables Z_1 y Z_2 con una desviación estándar σ aplicando la fórmula $Z_i = \sigma Y_i + m$ como se presenta en la rutina `creeobservaciones`.

```
!---   PliniAD/gausspolar.f   -------------------
      SUBROUTINE gausspolar(y1,y2)
!-----------------------------------------------
! Forma polar de la tranformacion Box-Muller para
! conseguir variables de funcion de distribucion
! normal a partir de variables
! de funcion de distribucion uniforma
      IMPLICIT NONE
      REAL x1,x2,y1,y2
      REAL w,rand
!
      w = 2.
      DO WHILE (w.GE.1)
        x1 = 2.*rand(0)-1
        x2 = 2.*rand(0)-1
        w = x1**2+x2**2
      ENDDO
      w = SQRT(-2.*log(w)/w)
      y1 = x1*w
      y2 = x2*w
      END
!-----------------------------------------------
```

```
!---   PliniAD/creeobs.f   ----------------------
      SUBROUTINE creeobservaciones
     *           (sigmau,sigmat,sigmar)
!-----------------------------------------------
! Permite agregar una perturbacion normal de
! ley N(0,sigmar) a los datos de
! radio robs. Idem para uobs y thetaobs.
! Los resultados estan escritos en un archivo.
      INCLUDE 'datos.inc'
      INTEGER i,zn
      REAL y1,y2
      REAL sigmau,sigmat,sigmar

      zn = z0
      DO i = 1,numobs
        CALL gausspolar(y1,y2)
        uobsp(i) = uobs(i)+sigmau*y1
        thetaobsp(i)= thetaobs(i)+sigmat*y2
        CALL gausspolar(y1,y2)
        robsp(i) = robs(i) + sigmar*y1
        zn = zn+dz
        WRITE(3,112)zn,uobsp(i),uobs(i),
     *                thetaobsp(i),thetaobs(i),
     *                robsp(i),robs(i)
      ENDDO
112   FORMAT(I5,6F12.6)
      END
!-----------------------------------------------
```

A.2. Método Lagrangiano

La función $\mathcal{L}$ sirve en la expresión del problema de minimización (4.2):

$$\begin{array}{l} \textit{Buscar } \mathrm{u}_0^* \textit{ tal como} \\ \min_{\mathrm{u}\in\mathcal{U}} \mathcal{J}o\mathcal{M}(\mathrm{u}_0) = \mathcal{J}o\mathcal{M}(\mathrm{u}_0^*), \end{array}$$

bajo la constricción del modelo (4.1):

$$\begin{cases} \dfrac{d\mathrm{u}}{dt} = M(\mathrm{u}) \text{ en } (z_0, z_t), \\ \mathrm{u}(0) = \mathrm{u}_0. \end{cases}$$

Cuando la condición de salida u_0 es el control a identificar, tiene la forma:

$$\begin{aligned} &\mathcal{L}(\mathrm{u}_0, \mathrm{u}, q) = \\ &= \mathcal{J}o\mathcal{M}(\mathrm{u}_0) + \int_{z_0}^{z_t} < q, \frac{d\mathrm{u}}{dz} - M(\mathrm{u}) >= \\ &= \mathcal{J}(\mathrm{u}) + \int_{z_0}^{z_t} < q, \frac{d\mathrm{u}}{dz} - M(\mathrm{u}) >= \\ &= \mathcal{J}(\mathrm{u}) - \int_{z_0}^{z_t} < \frac{dq}{dz}, \mathrm{u} > - \int_{z_0}^{z_t} < q, M(\mathrm{u}) > + \\ &\quad + < q(z_t), \mathrm{u}(z_t) > - < q(z_0), \mathrm{u}_0 > . \end{aligned}$$

Vamos a probar el Teorema 1:

Si el punto (u_0^*, u^*, q^*) es "punto silla" de la función $\mathcal{L}$, satisface:

$$\begin{aligned} &\forall(u_0, u), \quad \forall q, \\ &\mathcal{L}(u_0^*, u^*, q) \leq \mathcal{L}(u_0^*, u^*, q^*) \leq \mathcal{L}(u_0, u, q^*), \end{aligned} \tag{A.1}$$

y entonces u_0^* es la solución del problema de minimización (4.2).

Prueba 1 *Las dos desigualdades se tratan de manera diferente.*

1. *La desigualdad izquierda implica que* (u_0^*, u^*) *es solución de (4.1).*
   Se escribe "$\mathcal{L}(u_0^*, u^*, q) \leq \mathcal{L}(u_0^*, u^*, q^*)$*" como:*

$$\forall q,$$
$$\mathcal{J}(u^*) + \int_{z_0}^{z_t} < q, \frac{du^*}{dz} - M(u^*) > \leq$$
$$\leq \mathcal{J}(u^*) + \int_{z_0}^{z_t} < q^*, \frac{du^*}{dz} - M(u^*) > .$$

   Se simplifica en:

$$\int_{z_0}^{z_t} < q - q^*, \frac{du^*}{dz} - M(u^*) > \leq 0, \quad \forall q,$$

   y se integra por partes para deducir:

$$-\int_{z_0}^{z_t} < \frac{d(q-q^*)}{dz}, u^* > +$$
$$+ < (q-q^*)(z_t), u^*(z_t) > -$$
$$- < (q-q^*)(z_0), u_0^* > -$$
$$-\int_{z_0}^{z_t} < q - q^*, M(u^*) > \quad \leq 0, \quad \forall q. \tag{A.2}$$

   Sea una función de prueba ϕ *cualquiera tal que* $\phi(z_t) = \phi(z_0) = 0$*. Como (A.2) es verdadera para cualquier* q*, las elecciones de* $q = q^* + \phi$ *y* $q = q^* - \phi$ *conducen a:*

$$\pm \int_{z_0}^{z_t} < \frac{d\phi}{dz}, u^* > \pm \int_{z_0}^{z_t} < \phi, M(u^*) > \leq 0.$$

   Significa que u^* *satisface la primera ecuación del sistema (4.1) en el sentido de las distribuciones (es una teoría matemática bien conocida en la modelación de los elementos finitos por ejemplo). Nos queda demostrar que la condición de salida es realmente* $u^*(z_0) = u_0^*$*.*

   Para eso se integra por partes la ecuación (A.2):

$$\int_{z_0}^{z_t} < (q-q^*), \frac{du^*}{dz} - M(u^*) > -$$
$$- < (q-q^*)(z_t), u^*(z_t) > +$$
$$+ < (q-q^*)(z_0), u(z_0)^* > +$$
$$+ < (q-q^*)(z_t), u^*(z_t) > -$$
$$- < (q-q^*)(z_0), u_0^* > \quad \leq 0, \quad \forall q.$$

   Como u^* *satisface la primera ecuación del sistema (4.1) en el sentido de las distribuciones, conduce a:*

$$< (q-q^*)(z_0), u_0^* - u^*(z_0) > \quad \leq 0, \quad \forall q. \tag{A.3}$$

   Esta vez, la elección de $q = q^* \pm \phi$ *permite deducir que* (u_0^*, u^*) *es solución del problema (4.1).*

2. *La desigualdad derecha implica que* u^* *es el mínimo de* $\mathcal{J}$*, lo que significa que* u_0^* *minimiza* $\mathcal{J}o\mathcal{M}$*.*
   Se desarrolla "$\mathcal{L}(u_0^*, u^*, q^*) \leq \mathcal{L}(u_0, u, q^*)$*" como:*

$$\forall (u_0, u),$$
$$\mathcal{J}(u^*) - \int_{z_0}^{z_t} < \frac{dq^*}{dz}, u^* > - \int_{z_0}^{z_t} < q^*, M(u^*) >$$
$$+ < q(z_t), u^*(z_t) > - < q(z_0), u_0^* >$$
$$\leq \mathcal{J}(u) - \int_{z_0}^{z_t} < \frac{dq^*}{dz}, u > - \int_{z_0}^{z_t} < q^*, M(u) >$$
$$+ < q^*(z_t), u(z_t) > - < q^*(z_0), u_0 > .$$

   Entonces, $\forall (u_0, u)$*, tenemos:*

$$\mathcal{J}(u^*) - \mathcal{J}(u) \leq$$
$$\leq -\int_{z_0}^{z_t} < \frac{dq^*}{dz}, (u-u^*) > +$$
$$+ \int_{z_0}^{z_t} < q^*, M(u^*) - M(u) > +$$
$$+ < q^*(z_t), (u-u^*)(z_t) > -$$
$$- < q^*(z_0), u_0 - u_0^* >$$
$$\leq \int_{z_0}^{z_t} < q^*, \frac{d(u-u^*)}{dz} > +$$
$$+ \int_{z_0}^{z_t} < q^*, M(u^*) - M(u) > -$$
$$- < q^*(z_t), (u-u^*)(z_t) > +$$
$$+ < q^*(z_0), u(z_0) - u^*(z_0) > +$$
$$+ < q^*(z_t), (u-u^*)(z_t) > -$$
$$- < q^*(z_0), u_0 - u_0^* > .$$

   que se simplifica gracias a (A.2) y (A.3) en:

$$\forall (u_0, u),$$
$$\mathcal{J}(u^*) - \mathcal{J}(u) \leq$$
$$\leq -\int_{z_0}^{z_t} < q^*, \frac{du}{dz} - M(u) > -$$
$$- < q^*(z_0), u_0 - u(z_0) > .$$

   La elección de una pareja (u_0, u) *que satisface (4.1), así satisface* $\mathcal{M}(u_0) = u$*, permite obtener:*

$$\mathcal{J}(u^*) - \mathcal{J}(u) \leq 0, \quad \forall u_0,$$

   es decir:

$$\mathcal{J}o\mathcal{M}(u_0^*) - \mathcal{J}o\mathcal{M}(u_0) \leq 0, \quad \forall u_0.$$

   Lo cual significa que u_0^* *es la solución del problema de minimización (4.2).*
   Fin de la prueba.

A.3. Existencia y unicidad

Numerosos fenómenos naturales son descritos por ecuaciones basadas en consideraciones físicas. A veces, no son completamente estudiadas desde el punto de vista matemático por lo que las ecuaciones complejas no son tan fáciles de manejar. Así, existen muchos sistemas de ecuaciones (como las de Navier-Stokes) para las cuales la existencia y unicidad de una solución (cuando existe) no han sido probadas aún. Sin embargo, se usan en el cálculo numérico. El problema es todavía más fuerte cuando uno se interesa en la solución de problemas inversos.

Un problema continuo está "bien planteado" en el sentido de Hadamard cuando sus ecuaciones permiten probar:

1. la existencia de una solución,
2. su unicidad y
3. su estabilidad con respecto a sus datos 'variables de entradas del problema).

Al revés un problema está "mal planteado" si una de estas condiciones no está satisfecha. En general, un problema inverso está mal planteado, aunque el problema directo describiendo el fenomeno esté bien planteado.

Para construir, problemas inversos bien planteados, es posible actuar en la función objetivo aumentándola con términos de regularización o de penalización. Así cuando la variable de control del problema inverso es la condición de salida, se puede elegir una función objetivo tal que:

$$\mathcal{J}_{\beta,\epsilon}(\mathcal{M}(u_0)) = \mathcal{J}(u) + \beta||u_0 - u_{eb}||^2 + \epsilon||\Delta u_0||^2,$$

donde β y ϵ son dos pesos positivos. El término de esbozo $||u_0 - u_{eb}||^2$ significa que el control u_0 está buscado cerca de u_{eb}. El término $||\Delta u_0||^2$ no tiene sentido en el caso del modelo pliniano porque buscamos valores individuales. Se usa frecuentemente cuando se buscan condiciones iniciales de problemas espacio-temporales por lo que asegura una cierta regularidad de la variable buscada.

Bibliografía

Byrd, R.H., Lu, P. y Nocedal, J., 1995. A limited memory algorithm for bound constrained optimization, SIAM Journal on Scientific and Statistical Computing, 16: 1190–1208.

Calder, E.S., Sparks, R.S.J. y Woods, A.W., 1997. Dynamics of co-ignimbrite plumes generated from pyroclastic flows of Mount St. Helens (7 August 1980),Bulletin of Volcanolology, 58: 432–440.

Casadevall, T.J., 1993. Volcanic ash and aviation safety, U.S. Geological Survey Bulletin, 2047: 1–6.

Charpentier, I., 2007. Adjoint modelling experiments on eruptive columns, Geophysical Journal International, 169: 1356–1365.

Charpentier, I., 2008. Variational coupling of Plinian column models and data: Application to El Chichón Volcano, Journal of Volcanology and Geothermal Research, 175: 501–508.

Charpentier, I. y Espíndola, J.M., 2005a. Local Sensitivity Analysis of a Numerical Model of Volcanic Plinian Columns through Automatic Differentiation, Mathematical Geology, 37: 95–113.

Charpentier, I. y Espíndola, J.M., 2005b. A study of the entrainment function in models of plinian columns: characteristics and calibration, Geophysical Journal International, 160: 1123–1130.

Charpentier, I. y Utke, J., to appear. Fast Higher-Order Derivarive Tensors with Rapsodia, Optimization Methods and Software.

Costa, A., Macedonio, G. y Folch, A., 2006. A three-dimensional Eulerian model for volcanic ash dispersion and deposition, Earth and Planetary Science Letters, 241: 634–647.

Dobran, F., Neri, A. y Macedonio, G., 1993. Numerical simulation of collapsing volcanic columns, *Journal of Geophysical Research*, 98: 4231–4259.

Evensen, G., 2003. The Ensemble Kalman Filter: Theoretical Formulation and Practical Implementation, Ocean Dynamics, 53: 343–367.

Folch, A. y Felpeto, A., 2005. A coupled model for dispersal of tephra during sustained explosive eruptions. Journal of Volcanology and Geothermal Research, 145: 337–349.
Recipes for adjoint code construction, ACM Transactions on Mathematical Software, 24: 437–474.

Gilbert, J.-Ch. y Lemaréchal, C., 1989. Some numerical experiments with variable storage quasi-Newton algorithms, Mathematical programming, 45: 407–435.

Glaze, L.S., Baloga, S.M. y Wilson, L., 1997. Transport of atmospheric water vapor by volcanic eruption columns, Journal of Geophysical Research, 102(D5): 6099–6108.

Griewank, A., Juedes, D., Srinivasan, J. y Tyner, C., 1996. ADOL-C, A Package for the Automatic Differentiation of Algorithms Written in C/C++, Algor.755, ACM Transactions on Mathematical Software, 22: 131–167.

Griewank, A., 2000. Evaluating Derivatives: principles and techniques of algorithmic differentiation, Frontiers in Appl. Math. 19, SIAM, Philadelphia, 369pp.

Harris, D.M., Rose, W.I. Jr, Roe, R. y Thompson, M.R., 1981. Radar observations of ash eruptions, U.S. Geoleological Survey Professional Paper, 1250: 323–333.

Harris, D.M. y Rose, W.I. Jr, 1983. Estimating particels size, concentrations, and total mass of ash in volcanic clouds using weather radar, Journal of Geophysical Research, 88: 10969–10983.

Herzog, M., Graf, H.F., Textor, C. y Oberhuber, J., 1998. The effect of phase changes of water on the development of volcanic plumes, Journal of Volcanology and Geothermal Research, 87: 55-74

Hoblitt, R.P., 1986. Observations of the eruptions of July 22 and August 7, 1980, at Mount St.

Helens, Washington: U.S. Geological Survey Professional Paper 1335: 1–44.

Holasek, R.E., Self, S. y Woods, A.W., 1996. Satellite observations and interpretation of the 1991 Mount Pinatubo eruption plumes, Journal of Geophysical Research, 101: 27635–27665.

ICAO (International Civil Aviation Organization), 2001. Manual on volcanic ash, radioactive material, and toxic chemical clouds, Doc 9691-AN/954.

Ishimine, Y., 2006. Sensitivity of the dynamics of volcanic eruption columns to their shape, Bulletin of Volcanology, 68: 516–537.

Kalman, R.E., 1960. A new approach to linear filter and prediction problems, Transaction of the ASME: Journal of Basic Engineering, 82: 35–45.

Le Dimet, F.-X. y Talagrand, O., 1986. Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects, Tellus, 38: 97–110.

Lewis, J. y Derber, J., 1985. The use of the adjoint equations to solve a variational adjustment problem with advective constraints, Tellus, 37: 309–327.

Lions, J.-L., 1971. Optimal control of systems governed by partial differential equations, Springer-Verlag, Berlin, 396 pp.

Lu, P., Nocedal, J. y Zhu, C., 1995. A limited memory algorithm for bound constrained optimization, SIAM Journal of Scientific Computing, 16: 1190–1208.

Morgenstern, J., 1985. How to compute fast a function and all its derivatives, a variation on the theorem of Baur-Strassen, SIGACT News, 16: 60–62.

Morton, B.R., Taylor, G. y Turner, J.S., 1956. Turbulent gravitational convection from maintained and instantaneous sources, Proceedings of the Royal Society A, 234: 1–23.

Oberhuber, J.M., Herzog, M., Graf, H.F. y Schwanke, K., 1998. Volcanic plume simulation on large scales Journal of Volcanology and Geothermal Research, 87: 29-53

Patrick, M.R., 2007. Dynamics of Strombolian ash plumes from thermal video: Motion, morphology, and air entrainment, Journal of Geophysical Research, 112, B06202.

Pham, D.T., Verron, J. y and Roubaud, M.C., 1998. A singular evolutive extended Kalman filter for data assimilation in oceanography, Journal of Marine Systems, 16: 323–340.

Prandtl, L., 1949. The essentials of fluid dynamics, Blackie & *S*on Ltd, Glasgow.

Projet Tropics (INRIA Sophia Antipolis), Tapenade: On-line automatic differentiation Engine.

Scott Oswalt, J., Nichols, W., y O'Hara, J.F., 1996. Meteorological observations of the 1991 Mount Pinatubo eruption, *Fire and Mud*, U.S. Geological Survey, 625–636.

Settle, M., 1978. Volcanic eruption clouds and the thermal power output of explosive eruptions, Journal of Geophysical Research, 3: 309–324.

Sparks, R.S.J., 1986. The dimension and dynamics of volcanic eruption columns, *Journal of Volcanology and Geothermal Research*, 48: 13–15.

Sparks, R.S.J., Bursik, M.I., Carey, S.N., Gilbert, J.S., Glaze, L.S., Sigurdsson, H. y Woods, A.W., 1997. Volcanic plumes, John Wiley and Sons Ltd, 590 pp.

Talagrand, O. y Courtier, P., 1987. Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory, Quaterly Journal of the Royal Meteorological Society, 113: 1311–1328.

Textor, C., Graf, H.-F., Longo, A., Neri, A., Ongaro, T.E., Papale, P., Timmreck, C. y Ernst, G.G.J., 2005. Numerical simulation of explosive volcanic eruptions from the conduit flow to global atmospheric scales,Annals of Geophysics, 48: 817–842.

Turner, J.S., 1962. The starting plume in neutral surroundings, *J. Fluid Mech.*, 13: 356–368.

Valentine, G.A. y Wolhetz, K.H., 1989. Numerical models of Plinian eruption columns and pyroclastic flows, Journal of Geophysical Research, 94: 1867–1887.

Wilson, L., 1976. Explosive volcanic eruptions III. Plinian eruption columns, Geophysical Journal of the Royal Astronomical Society, 45: 543–556.

Wilson, L., Sparks, R.S.J., Huang, T.C. y Watkins, N.D., 1978. The control of volcanic column heights by eruption energetics and dynamics, Journal of Geophysical Research, 83(B4): 1829–1836.