Resolución de ecuaciones

La semana pasada veíamos la utilidad de *Mathematica* para ilustrar conceptos como los de convergencia uniforme y puntual así como la aproximación local de una función por sus polinomios de Taylor. En esta práctica empezamos a explorar algunas de las muchas posibilidades de *Mathematica* como herramienta de cálculo. Concretamente, vamos a estudiar la resolución de varios tipos de ecuaciones que por su sencillez conceptual nos han parecido apropiadas como punto de partida, pero cuya dificultad numérica pone de manifiesto la utilidad de *Mathematica* para realizar cálculos laboriosos y facilitar así la resolución de muchos problemas.

Ecuaciones

Recuerda que en *Mathematica* el símbolo de igualdad "=" es la forma usual de escribir el comando `Set` (léase *atribuye* o *asigna*). Así `expr1=expr2` y `Set[expr1,expr2]` son dos formas de decirle a *Mathematica* que evalúe inmediatamente `expr2` y el resultado de esta operación lo asigne en adelante al símbolo `expr1`. Se comprende así que "=" no pueda usarse en *Mathematica* para representar ecuaciones.

Fíjate en que resolver una ecuación, por ejemplo $x^4 - 3x^2 + 7 = 0$, consiste realmente en *comprobar* una igualdad; es decir, se trata de obtener los valores de la variable (o de las variables) que hacen que dicha igualdad sea *verdadera*.

Mathematica dispone de un operador, `Equal`, que sirve para *comprobar* igualdades. Su sintaxis es `Equal[expr1,expr2]` que suele escribirse en la forma más corta `expr1==expr2` y lo que hace es evaluar ambas expresiones devolviendo el valor "`True`" si la igualdad es verdadera o "`False`" en otro caso.

Pues bien, una ecuación en *Mathematica* se representa, precisamente, en la forma `expr1==expr2` donde al menos una de las expresiones `expr1` o `expr2` es simbólica, es decir contiene símbolos (variables) a los que no se han asignado anteriormente valores. Es claro que en tal caso la igualdad no puede comprobarse por lo que `expr1==expr2` da también como salida `expr1==expr2`. A continuación veremos algunas de las muchas funciones de *Mathematica* para trabajar con ecuaciones y resolverlas.

Resolución de ecuaciones en una variable

La orden `Solve[ecuación,var]`, intenta resolver "`ecuación`" en la variable "`var`". Por ejemplo, *Mathematica* conoce las fórmulas algebraicas para las soluciones simbólicas exactas de ecuaciones polinómicas en una variable de grado menor que cinco.

```
In[1]:=
  Solve[a x^2+b x+c == 0,x]
```

Out[1]=

$$\left\{\left\{x \to \frac{-b-\sqrt{b^2-4ac}}{2a}\right\}, \left\{x \to \frac{-b+\sqrt{b^2-4ac}}{2a}\right\}\right\}$$

De la misma forma puedes obtener las soluciones simbólicas de las ecuaciones polinómicas generales de tercer grado, $x^3 + ax^2 + bx + c = 0$, y de cuarto grado $x^4 + ax^3 + bx^2 + cx + d = 0$. Compruébalo ahora mismo (no olvides dejar un espacio entre los coeficientes y la variable `x`). ¿Te explicas por qué, a pesar de conocerse fórmulas generales para resolver dichas ecuaciones, éstas no se enseñan en la escuela?

Observa que la salida de `Solve` es una *lista* cuyos elementos son listas de *reglas*, cada una de las cuales representa una solución. Una expresión de la forma `a->b` se llama en *Mathematica* una *regla*. La utilidad de las reglas es que sirven para realizar sustituciones. El operador de reemplazamiento `/.` indica

que la regla o lista de reglas que hay a su derecha debe ser sustituida una vez en cada parte de la expresión que está a su izquierda. Los siguientes ejemplos se explican por sí mismos.

```
In[2]:=
 x^2+y^x /.x->3
Out[2]=
 9+y^3
In[3]:=
 x^y /.{x->y,y->x}
Out[3]=
 y^x
In[4]:=
 x+y /.{{x->1},{y->2}}
Out[4]=
 {1+y,2+x}
```

La forma que tiene la salida de `Solve` permite sustituir fácilmente en una función las soluciones de una ecuación. Procura entender lo que sigue.

```
In[5]:=
 sol=Solve[x^3-5x^2+2x+8==0,x]
Out[5]=
 {{x->-1},{x->2},{x->4}}
In[6]:=
 poly=1/4 x^4 -5/3 x^3 +x^2 +8x+17; poly/.sol
Out[6]=
 {143/12, 83/3, 67/3}
In[7]:=
 poly/.sol[[2]]
Out[7]=
 83
 --
 3
In[8]:=
 poly/.sol[[{1,3}]]
Out[8]=
 {143/12, 67/3}
```

Ejercicio 1
Calcula los puntos críticos de $f(x) = x^5 - 2x^4 + 3x^3 - x - 1$ y el valor de f en cada uno de ellos. Calcula el valor numérico de los puntos críticos y de los valores críticos. Usa la derivada segunda para clasificar los puntos críticos reales. Halla los puntos de inflexión de f. Haz la gráfica de f en un intervalo apropiado y confirma los resultados obtenidos.

Una función de *Mathematica* parecida a `Solve` es `Roots[ecuación,var]`. Las diferencias son que `Roots` sólo puede usarse para ecuaciones polinómicas y proporciona como salida una *disyuntiva lógica* entre ecuaciones que representan las soluciones de dicha ecuación. La disyuntiva lógica "o" se representa en *Mathematica* por "`||`". Observa la diferencia.

```
In[9]:=
 Solve[2+3x-4x^2+x^3==0,x]
Out[9]=
 {{x->2},{x->1-√2},{x->1+√2}}
In[10]:=
 Roots[2+3x-4x^2+x^3==0,x]
Out[10]=
 x==1-√2 || x==1+√2 || x==2
```

Y, ya puestos, ¿qué pasa con las ecuaciones polinómicas de grado mayor o igual que cinco? Bueno, como seguramente sabes, Evariste Galois (1811-1832) demostró que no pueden, en general, resolverse por radicales; es decir, que es matemáticamente imposible encontrar fórmulas explícitas para sus soluciones. Por ello, no te extrañes de que con frecuencia *Mathematica* no pueda resolverlas de forma exacta.

```
In[11]:=
  Solve[70+66x+29x^2-39x^3-34x^4+3x^5+x^6==0,x]
Out[11]=
  {{x->-7},{x->5},{x->-(-1)^(1/3)},{x->(-1)^(2/3)},{x->-√2},
  {x->√2}}
In[12]:=
  Solve[3+3x-7x^2-x^3+2x^4+3x^7-3x^8-x^9+x^10==0,x]
Out[12]=
  {{x->1},{x->-√3},{x->√3},{x->Root[1+ 2 #1+#1^7, &,1]},
  {x->Root[1+2 #1+#1^7, &,2]}, {x->Root[1+2 #1+#1^7, &,3]},
  {x->Root[1+2 #1+#1^7, &,4]}, {x->Root[1+2 #1+#1^7, &,5]},
  {x->Root[1+2 #1+#1^7, &,6]}, {x->Root[1+2 #1+#1^7, &,7]}}
```

En el primer caso hemos tenido suerte: `Solve` ha sido capaz de encontrar todas las soluciones de la ecuación (observa que dos de ellas son complejas). En el segundo *Mathematica* no ha podido encontrar todas las raíces de un polinomio tan complicado. Ha encontrado tres raíces exactas 1, $-\sqrt{3}$ y $\sqrt{3}$; y nos dice que las restantes siete son las raíces del polinomio $1+2x+x^7$. En los casos en que `Solve` no puede dar soluciones exactas, podemos conseguir soluciones numéricas aproximadas con el comando `NSolve[ecuación,var]`.

```
In[13]:=
  NSolve[3+3x-7x^2-x^3+2x^4+3x^7-3x^8-x^9+x^10==0,x]
Out[13]=
  {{x->1.},{x->-1.73205},{x->1.73205},{x->-0.496292},
  {x->-0.868688-0.585282 i},{x->-0.868688+0.585282 i},
  {x->0.0763556-1.14095 i},{x->0.0763556+1.14095 i},
  {x->1.04048-0.56735 i},{x->1.04048+0.5673 i}}
```

Y usamos `NSolve[ecuación,var,n]` si queremos las soluciones con `n` dígitos de precisión. Haz la prueba en el ejemplo anterior con `n=30`. También puedes usar `NRoots[ecuación,var,n]` cuando las raíces no pueden obtenerse simbólicamente. La diferencia entre `NRoots` y `NSolve` es análoga a la indicada entre `Roots` y `Solve`.

Hasta ahora hemos considerado ecuaciones polinómicas pero `Solve[ecuación,var]` también puede resolver otros tipos de ecuaciones. Por ejemplo, teniendo en cuenta que `var` puede ser cualquier símbolo que aparezca en `ecuación`, podemos resolver respecto de `var=f[x]` ecuaciones de la forma `p[f[x]]==0`, donde `p` es una función polinómica.

```
In[14]:=
 Solve[3-15Log[x]+7Log[x]^2==0,Log[x]]
Out[14]=
 {{Log[x]->1/14 (15-Sqrt[141])},{Log[x]->1/14 (15+Sqrt[141])}}
```

A veces `Solve` puede resolver ecuaciones en las que intervienen funciones trascendentes.

```
In[15]:=
 Solve[ArcCos[x]-ArcTan[x]==0,x]
Out[15]=
 {{x -> Sqrt[-1/2+Sqrt[5]/2]}}
In[16]:=
 Solve[Log[x+(a+x^2)^(1/2)]==b,x]
Out[16]=
 {{x -> 1/2 E^-b (-a+E^(2b))}}
```

Pero lo más frecuente es que dichas ecuaciones no puedan ser resueltas exactamente en forma simbólica o que no puedan obtenerse todas las soluciones posibles de la ecuación. *Mathematica* suele dar en estos casos mensajes de advertencia. Por ejemplo, cuando al resolver una ecuación *Mathematica* utiliza alguna función inversa, lo indica con un mensaje en el que también advierte de que puede haber más soluciones.

```
In[17]:=
 Solve[Sin[x]Cos[x]==0,x]
 Solve::ifun: Inverse functions are being used by Solve, so some solutions may not be found.
Out[17]=
 {{x -> 0},{x -> -π/2},{x -> π/2}}
```

`Reduce[ecuación,var]` es la función análoga a `Roots` para ecuaciones no polinómicas. `Reduce` transforma la ecuación inicial en un conjunto de ecuaciones más simples equivalentes a ella; dichas ecuaciones vendrán expresadas usando "operadores lógicos": `&&` es la conjunción "y"; `||` es la disyunción "o"; `!=` es el símbolo de desigualdad. Además, si la ecuación tiene parámetros, `Reduce` presenta los distintos casos que pueden darse según los valores de los parámetros. Por el contrario, `Solve` sólo proporciona soluciones genéricas y descarta cualquier solución cuya existencia dependa de valores particulares de los parámetros. El siguiente ejemplo es claro.

```
In[18]:=
 Reduce[a x^2+b x+c == 0,x]
Out[18]=
 a≠0 && x==(-b-Sqrt[b^2-4ac])/(2a) || a≠0 && x== (-b+Sqrt[b^2-4ac])/(2a) || c==0 &&
 b==0 && a==0 || b≠0 && x==-c/b && a==0
```

Pero también puede ocurrir que una ecuación no pueda resolverse con ninguno de los comandos que hemos visto hasta ahora.

```
In[19]:=
 NSolve[Cos[x] == x,x]
Solve::tdep: The equations appear to involve the variables to be solved
for in an essentially non-algebraic way.
Out[19]=
 NSolve[Cos[x] == x,x]
```

En estos casos podemos todavía encontrar una solución numérica aproximada usando el comando `FindRoot[función,{var,valor inicial}]` el cual utiliza el método de Newton para obtener un cero de `función` partiendo del `valor inicial` de la variable `var`. Este comando también admite la sintaxis `FindRoot[ecuación,{var,valor inicial}]`.

```
In[20]:=
 FindRoot[Cos[x] == x,{x,1}]
Out[20]=
 {x->0.739085}
```

El valor inicial debe ser un punto que esté "más o menos" cerca del cero de la función que queremos buscar. En cualquier caso hay que tener en cuenta que aun escogiendo valores iniciales cercanos el resultado puede ser muy distinto. Para elegir los valores iniciales una representación gráfica suele ser de gran utilidad.

Ejercicio 2

Calcula las soluciones de $8\operatorname{sen} x + 1 - \frac{1}{3}x^2 = 0$.

Ejercicio 3

Calcula la solución de $\tan(x) = \frac{1}{x}$ en el intervalo $]0,\pi/2[$.

Ejercicio 4

Partiendo del valor inicial $0{,}02$, usa `FindRoot` para calcular una raíz de x^4, con seis dígitos exactos. Usa para ello las opciones de `FindRoot` que creas oportunas.

Resolución de sistemas de ecuaciones

Los comandos vistos, con la excepción de `Roots` pueden usarse también para calcular las soluciones de sistemas de ecuaciones en varias variables. Lo único que hay que hacer es sustituir en cada caso `ecuación` por una lista de ecuaciones `{ecuac1,ecuac2,...}` y especificar también en una lista las variables respecto de las que queremos resolver el sistema `{var1,var2,...}`. Al igual que antes, `Solve` y `NSolve` sirven principalmente para resolver sistemas de ecuaciones polinómicas. El primero sólo resuelve sistemas de grado bajo pero con soluciones exactas, expresadas mediante radicales. El segundo da soluciones aproximadas pero sirve para cualquier grado. Como puedes suponer, el comando `Reduce` cuando se aplica a sistemas de ecuaciones que contienen parámetros realiza una discusión de las soluciones del sistema según los posibles valores de los parámetros. Es particularmente útil para sistemas de ecuaciones lineales con parámetros.

También disponemos de `Eliminate[{ecuac1,ecuac2,...},{var1,var2,...}]` que algunas veces permite eliminar ciertas variables del sistema de ecuaciones. Compara.

```
In[21]:=
 Solve[{a x^2+b y==5, x-b y==1},{x,y}]

In[22]:=
 Reduce[{a x^2+b y==5, x-b y==1},{x,y}]

In[23]:=
 Eliminate[{a x^2+b y==5, x-b y==1},{y}]

In[24]:=
 FindRoot[{x^2+y^2==1, x Exp[x]==y},{x,0.1},{y,0.5}]

In[25]:=
 Solve[{(a-b) x^2+c==0, a^2==3, b^2==3},x]

In[26]:=
 Reduce[{(a-b) x^2+c==0, a^2==3, b^2==3},x]
```

Es interesante saber que `Solve[ecuaciones, variables, elimina]` trata de resolver el sistema de ecuaciones en las variables indicadas eliminando las variables `elimina`. También puede usarse la misma estructura con `NSolve`. Esto puede ser útil para, por ejemplo, calcular la intersección de dos curvas planas en paramétricas.

Ejercicio 5

Encontrar una solución del siguiente sistema de ecuaciones "cerca" de $(0,0)$:

$$\begin{cases} \operatorname{sen} x \cos y = \dfrac{1}{4} \\ xy = 1 \end{cases}$$

Ejercicio 6

Estudiar los extremos de la función $f(x) = x^2 - 10x - 40 + \dfrac{1}{10x^2 - 100x + 251}$.