1 Utilización del toolbox PDE de Matlab

Este *toolbox* nos permite resolver ecuaciones diferenciales en derivadas parciales utilizando el método de elementos finitos. En particular, con la utilización de la interfaz gráfica denominada **pdetool** veremos que es muy sencillo especificar el dominio 2-D de nuestro problema, la triangulación del dominio, los coeficientes de la EDP y las condiciones de contorno.

A continuación se resumen algunos elementos básicos para trabajar con el toolbox.

◯ Para **comenzar** a trabajar con la interfaz gráfica para EDPs de Matlab, teclearemos **pdetool** en la línea de comandos.

◯ Para **especificar el dominio** de nuestro problema procederemos del siguiente modo: en **Options**, elegir **Axes Limits** para concretar el área (x, y) (por defecto, $x = -1.5 : 1.5$ e $y = -1 : 1$); seleccionar **Grid** para situar un mallado sobre el área que aparece y **Snap** para ajustar automáticamente la figura de nuestro dominio al mallado (esto facilita dibujar regiones simples). Seleccionar **Grid Spacing** si se desea modificar el espaciado del mallado (el que está por defecto es 0.5 en x y 0.2 en y).

Los dominios se construyen en base a la composición (suma y/o resta) de una serie de dominios elementales tales como rectángulos, polígonos y elipses (los que aparecen en el menú de opciones), cuya manipulación es muy sencilla. Se recomienda escoger un par de ejemplos para comprobarlo.

Para formar el área de interés en nuestro problema, escribiremos en el espacio destinado a **Set formula** (que aparece sobre la región gráfica) la "fórmula" que describe nuestro dominio como suma y/o resta de las figuras elementales.

Si queremos guardar este dominio para utilizarlo a posteriori, seleccionaremos **Export geometry description** del menú de la opción **Draw** y elegiremos OK en el cuadro mostrado. De este modo, "exportaremos" los datos de la geometría bajo los nombres de las variables **gd sf ns** (salvo que se cambie). De hecho, sin entrar en detalles, esta especificación de la geometría debe ser procesada antes de poder utilizarla con diferentes comandos Matlab. Es más sencillo esperar a tener especificadas las condiciones de contorno y exportar simultáneamente éstas y la geometría del problema, como comentaremos a continuación.

◯ Para **establecer las condiciones de contorno**, seleccionar en **Boundary** la opción **Boundary mode** o, equivalentemente, pinchar el botón $\partial\Omega$. Seleccionar uno o varios segmentos y en el menú de **Boundary** elegir **Specify Boundary Conditions**. Se pueden especificar condiciones de contorno tipo Dirichlet de la forma h*u=r introduciendo los valores de los parámetros h y r (por defecto, $h = 1$ y $r = 0$). Si elegimos condiciones de contorno tipo

Neumann, deberemos especificar los coeficientes g y q. La forma general de la condición de contorno aparece en la parte superior del cuadro de diálogo (n es el vector unitario normal y c es un coficiente en la EDP).

Para guardar las condiciones de contorno en una forma que permita ser utilizada en un programa Matlab, seleccionar **Export Decomposed Geometry, Boundary Cond's** en el menú de **Bundary** y elegir OK en el cuadro. De este modo, exportaremos los datos de geometría y condiciones de contorno con los nombres g b (salvo que los cambiemos).

◯ Para **especificar la EDP** a resolver, seleccionaremos **PDE specification** en el menú de la opción **PDE** y escogeremos un problema prototipo (por defecto, elíptico, de la forma -div(c*grad(u))+ a*u =f). Introducir los coeficientes c y a de la EDP y la función f del lado derecho de la ecuación. Los coeficientes pueden ser funciones de x, y y para problemas no lineales también pueden ser función de u, ux, uy. Utilizaremos entonces x, y, u, ux, uy en la expresión a introducir, dándonos cuenta de que Matlab interpreta estas cantidades como vectores. Teniendo en cuenta esto, si queremos introducir la función xy, teclearemos x.*y en lugar de x*y.

◯ Seleccionando **Parameters** en el menú de la opción **Mesh** estableceremos (opcionalmente) los parámetros de la **triangulación inicial** de nuestro problema de elementos finitos. Para generar la triangulación, pincharemos en el botón con el triángulo o, alternativamente, elegiremos **Initialize Mesh** en el menú de **Mesh**.

Para guardar la triangulación de forma que pueda ser utilizada en un programa Matlab, seleccionaremos **Export Mesh** en el menú de **Mesh** y pincharemos OK en el cuadro de diálogo. Esto permite exportar los vértices, los bordes y la ordenación de los triángulos con los nombres: p e t (salvo cambio), respectivamente. Démonos cuenta de que necesitaremos esta información si queremos comparar la solución exacta y la aproximada en los vértices o calcular diversas normas del error que se comete.

◯ Para **refinar la triangulación** pincharemos el triángulo dividido o seleccionaremos **Refine Mesh** en el menú de **Mesh**. El método de refinamiento por defecto es regular (para cambiarlo, iremos a **Parameters**). La solución numérica de la EDP puede realizarse de manera adaptativa seleccionando primero **Solve Parameters** y luego **Adaptive Mode**, en el menú de la opción **Solve**: introduciremos el máximo número de triángulos que deseamos o el máximo n'umero de pasos de refinamiento.

◯ Para **resolver la EDP** y **dibujar la solución**, seleccionaremos **Solve PDE** en el menú de **Solve**. Para exportar la solución al espacio de trabajo principal de Matlab, elegiremos **Export Solution**. Disponemos de varias opciones de

gráficos para representar la solución: seleccionaremos **Parameters** del menú de **Plot**.

1.1 Trabajando desde la línea de comandos de Matlab

Una vez resuelto nuestro problema de contorno utilizando **pdetool** y exportados los datos descritos con anterioridad, podemos procesar esta información desde la línea de comandos de Matlab. En primer lugar, podría ser interesante guardar la descomposición de la geometría de nuestro problema (g) en un fichero (prob1g.m, por ejemplo) que podamos utilizar más adelante. Una vez que hemos elegido en *Boundary* la opción *Export Decomposed Geometry, Boundary Conds*, escribiremos en la línea de comandos **fid=wgeom(g,'prob1g')**. Para hacer lo mismo con la información sobre el contorno (b) que queremos guardar en un fichero (prob1b.m, por ejemplo), escribiremos **fid=wbound(b,'prob1b')**.

De hecho, con estos dos ficheros es bastante fácil resolver un problema de contorno sencillo sin necesidad de recurrir a la interfaz gráfica. Para ello, establecemos en primer lugar una triangulación sobre la geometría definida en el fichero prob1g.m, escribiendo **[p,e,t]=initmesh('prob1g')**; el comando **initmesh** permite especificar determinados parámetros relativos a las característica de la triangulación. Podemos visualizar la triangulación con **pdemesh(p,e,t)**. Para refinar la triangulación, escribiremos **[p,e,t] = refinemesh('prob1g',p,e,t)**. Por otro lado, los coeficientes de la EDP a resolver pueden ser especificados directamente. Por ejemplo, la ecuación escalar genérica es de la forma -div(c grad u) + a u = f. Por tanto, para resolver el problema con $c = 1$, $a = 0$ y $f = 8 - 16(x^2 + y^2)$, escribiríamos

```
c= '1'; a= '0'; f = '8 - 16.*(x.^2 + y.^2)'
```

Para obtener entonces la solución aproximada a nuestro problema de contorno, con condiciones de contorno especificadas en el fichero prob1b.m, tecleamos **u = assempde('prob1b',p,e,t,c,a,f)**. Para representar gráficamente la solución u, consideramos **pdesurf(p,t,u)**.

1.2 Post-procesado de la solución

Hay varios comandos Matlab que son útiles para obtener información sobre la solución aproximada y para el cálculo de errores (en problemas modelo) cuando la solución exacta es conocida. Escribiendo **[ux,uy] = pdegrad(p,t,u)** obtenemos el gradiente de u en cada triángulo (la primera componente está en el vector ux y la segunda, en uy). Los valores de u pueden obtenerse asimismo en puntos que no sean vértices utilizando la función **tri2grid**. Por ejemplo, **v=tri2grid(p,t,u,x,y)** calcula los valores de u sobre el mallado definido por los vectores x e y (asumiendo que u es conocido en los vértices).