**Instrucciones:**

- Se enviará el fichero de cada práctica a la dirección `avaldes@ucm.es`.
- **El asunto del correo electrónico será:** GC2015 Práctica <poner número aquí>
- Los ficheros adjuntos al correo irán tal cual, sin comprimir o empaquetar de ninguna forma.
- En el cuerpo del mensaje figurarán los nombres, apellidos y correos electrónicos de los alumnos que envían la práctica. Salvo casos muy excepcionales y bien justificados, el grupo de alumnos se mantendrá constante a lo largo de todo el curso.
- El plagio implicará que los alumnos pasarán a ser evaluados de la asignatura mediante la realización del examen final de la misma.
- Si la práctica contiene preguntas, deben responderse en un fichero aparte (PDF, escaneo de manuscrito, etc. No se admiten ficheros word).
- Cada práctica deberá enviarse antes del día y hora indicado como límite. No se aceptarán envíos pasado ese momento.

P6: Dado un intervalo $[a, b]$, junturas $a < \xi_0 < \cdots < \xi_l < b$, coeficientes $= A = [(x_i, y_i)]$ y condiciones de diferenciabilidad $\nu = [\nu_0, \ldots, \nu_l]$ se creará una curva spline plana de grado $< k$ que sea de clase $C^{\nu_i - 1}$ en cada juntura usando la secuencia de nudos descrita en (6) de [1], implementando el algoritmo recursivo (9) de [1] para el cálculo de la curva spline. Para ello, se creará un módulo de Python llamado `splines.py` que contendrá una función con el siguiente perfil:

```
def spline2d(a, b, xi, k, nu, A, num_dots):
    '''Computes a plane spline curve of order k
       defined on the interval [a, b] with knots xi,
       multiplicities nu and coefficients A.
       Parameters:
       a, b -- ends of the interval, real numbers.
       xi -- list of breakpoints, a < xi[0] < ... < xi[-1] < b.
       k -- order of the curve, the degree is <= k - 1.
       nu -- number of smoothness conditions satisfied by the
             curve at each breakpoint.
       A -- list of coefficients of the B-spline basis,
            A = [[x0, y0], [x1, y1],..., [x[N], y[N]]
       num_dots -- number of dots of the spline to be plotted,
                   uniformly spaced along the interval [a, b].
       Returns:
       the spline curve as a numpy array of size (2, num_dots) '''
```

La función devolverá un error en el caso de que las longitudes de las listas no sean adecuadas. Se valorará el que el algoritmo produzca resultados correctos, la claridad y elegancia del código y el tiempo de ejecución. Se publicarán más adelante los tiempos máximos admisibles en las máquinas de referencia.
Opcional: implementar una versión gráfica interactiva del algoritmo en la que se muestre la curva junto con el polígono $A$.

En el fichero `splines.py` podrá haber otras funciones o clases, pero no habrá ningun código ejecutable (en particular, no habrá un `if __name__ == '__main__':`)
Se pueden utilizar los siguientes módulos: `numpy, matplotlib.`

1. Ejercicios:

   (a) Dada la secuencia de nodos $\mathbf{t} = \mathbb{Z}$, calcúlense las funciones B-spline $B_{ik}$ correspondientes.

   (b) Dada una secuencia de nodos arbitraria $\mathbf{t}$, demuéstrese que si $p$ es un polinomio de grado 1 entonces

   $$p = \sum_i B_{ik} p(t_i^\star),$$

   siendo

   $$t_i^\star = (t_{i+1} + \cdots + t_{i+k-1})/(k-1).$$

**Límite para entregar esta práctica:** lunes 4 de mayo.

# References

[1] Carl de Boor, B(asic)-Spline Basics, `ftp://ftp.cs.wisc.edu/Approx/bsplbasic.pdf`