

INSTRUCCIONES PARA INSTALAR LOS SOFTWARES NECESARIOS PARA LAS TAREAS

Vamos a implementar las tareas computacionales utilizando python 3.7, y el paquete PuLP que permite modelar problemas lineales y resolverlos usando diferentes *solvers*. Además utilizaremos la herramienta Jupyter Notebook para crear documentos que contienen texto, código ejecutable, ecuaciones, visualizaciones, etc. Es muy útil para hacer ciencia con python (o con cualquier otro lenguaje). Así para la tarea es obligatorio utilizar python 3.7, PuLP y Jupyter Notebook.

Para la tarea se puede trabajar en cualquier plataforma, Linux, Mac OS X o Microsoft Windows, pero es parte de la tarea conseguir instalar lo necesario en su computador de trabajo. De todas formas vamos a dar algunas instrucciones que debieran ser suficientes. Lo primero que recomendamos es instalar Anaconda Python 3.7 (www.anaconda.com/download/) que es una distribución de python especialmente pensada para hacer ciencia de los datos. Esta distribución se puede instalar incluso si python está ya instalado a partir de otra fuente. No es necesario desinstalar python ni ningún paquete de éste para instalar Anaconda.

Al instalar Anaconda también se instala Jupyter Notebook. Para correr Jupyter Notebook simplemente abrir una terminal y ejecutar la línea de comando

```
jupyter notebook
```

desde el directorio de trabajo. Esto abrirá una ventana del navegador que tengamos por defecto y mostrará los archivos en ese directorio. Los notebooks son archivos de extensión `.ipynb`. En clases mostraremos cómo utilizar esta herramienta.

Anaconda también incluye Conda, que es un gestor de paquetes y dependencia que permite instalar fácilmente nuevos paquetes de python (o de otros lenguajes). Para instalar el paquete PuLP podemos abrir una terminal y ejecutar

```
conda install -c conda-forge pulp
```

Para aprender lo básico para usar este paquete, le recomendamos estudiar los notebooks de ejemplo de problemas de programación lineal que incluimos para la tarea (`ejemplo2.ipynb`, `ejemplo3.ipynb` y `ejemplo4.ipynb`). En clases también veremos ejemplos de modelamiento y solución de problemas en este lenguaje. No es necesario conocer en detalle las instrucciones, sino entender a partir de los ejemplos como se modelan las variables, la función objetivo y las restricciones, y cómo se obtiene la solución óptima, si existe.