UNIVERSIDAD DE LOS ANDES

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

Modelado, Simulación y Optimización

Profesor

Germán Montoya O.
ga.montoya44@uniandes.edu.co

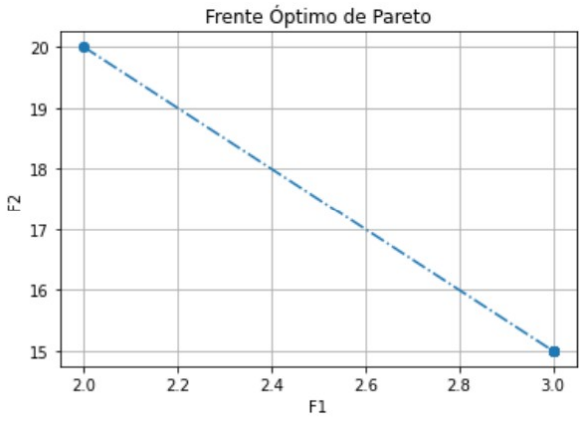
EXAMEN 2

NOTA: realizar todos los ejercicios valiéndose de los ejemplos y conceptos vistos en clase. Adicionalmente, apoyarse en la presentación "introducción a MATLAB.pptx" para poder hacer las implementaciones donde se requiera dicho lenguaje.

EJERCICIO 1 (20%): Método eConstraint en Pyomo

Resuelva el mismo caso presentado en "multiobjetivoHopsCosts_sumasPonderadas.py", pero **implementando el método de eConstraint en Pyomo**.

Tener en cuenta:
-Considerar el modelo "multiobjetivoHopsCosts_sumasPonderadas.py", el cual itera para cambiar los pesos del método de sumas ponderadas. Usted puede inspirarse en este modelo para realizar iteraciones que cambien el Epsilon del método de eConstraint.
-Se debe proponer un modelo multiobjetivo de tal forma que, al ejecutarlo **UNA ÚNICA VEZ**, este solucione el modelo para varios valores de Epsilon para finalmente arrojar la gráfica del frente óptimo de Pareto.
-El código fuente debería arrojar los mismos dos puntos (con iguales coordenadas) del frente óptimo de Pareto obtenido en "multiobjetivoHopsCosts_sumasPonderadas.py".



ENTREGABLE: El código fuente *.py con la gráfica del frente óptimo de Pareto.

EJERCICIO 2 (50%): Implementación de Newton Raphson para 3 dimensiones

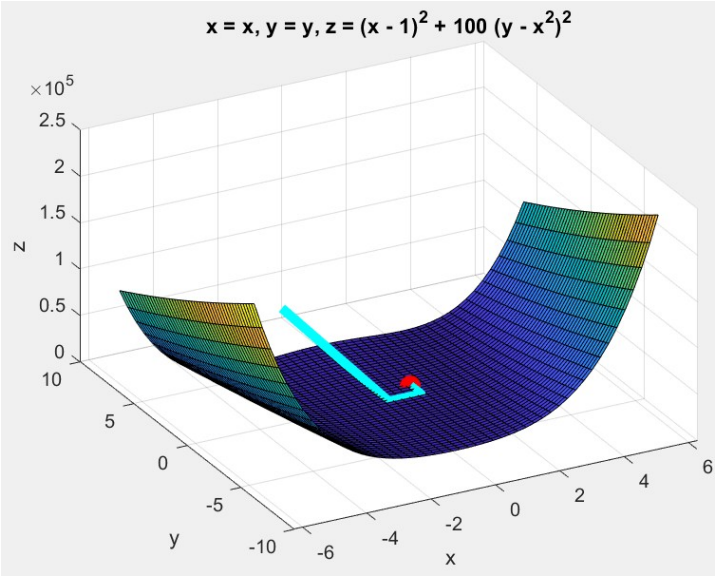
- Implemente en MATLAB los siguientes pasos para encontrar el mínimo de una función usando el método de Newton Raphson para tres dimensiones.
    - Teóricamente, defina y grafique la superficie $z = ¿$.
    - *Ayuda:* use la toolbox simbólica para definir la función. Además, use la función 'ezsurf()' para graficar la superficie.
    - Implemente el método de Newton Raphson para 3 dimensiones de acuerdo al siguiente pseudocódigo:

Algorithm Pseudocodigo de Newton Raphson para 3 dimensiones

$i \leftarrow 1$
$Inicializar\ x_i$
$\alpha \leftarrow 1$
$convergencia \leftarrow 0.001$
while do $|| \nabla f(x_i) || > convergencia$
    $x_{i+1} \leftarrow x_i - \alpha(H(f(x_i))^{-1} \nabla f(x_i)$
    $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$

    - Sintonice el paso (α) para que el mínimo se encuentre rápidamente.
    - Grafique el mínimo encontrado sobre la gráfica de la función teórica realizada anteriormente.
    - Grafique sobre la gráfica de la función teórica los puntos encontrados de cada iteración.
    - Se recomienda un punto de arranque ubicado en x=0, y=10.

El resultado debería lucir como la siguiente gráfica (la línea azul no necesariamente debe coincidir con la de uds, pero la roja si debe coincidir):



ENTREGABLE: un archivo de Matlab *.m.

EJERCICIO 3: Implementación del Algoritmo Simplex (30%)

Implemente el Algoritmo Simplex para solucionar el problema de Woodcarving:

$$max(3x_1 + 2x_2)$$
$$s.a:$$
$$2x_1 + x_2 \leq 100$$
$$x_1 + x_2 \leq 80$$
$$x_1 \leq 40$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Para la implementación tenga en cuenta lo siguiente:

- Los parámetros de entrada del algoritmo deberían ser las coordenadas de cada uno de los vértices del espacio de soluciones factibles. Estas coordenadas se pueden calcular manualmente para luego ser ingresadas como parámetros de entrada en la solución.
- Para realizar la prueba de optimalidad, asuma que el FEV actual inicial podría ser cualquier vértice, es decir, que se asigne aleatoriamente.
- El algoritmo debería verificar la prueba de optimalidad a partir del FEV actual inicial hasta cumplir la prueba de optimalidad, y así, ofrecer la solución del problema. En otras, palabras el algoritmo debería arrojar el valor óptimo de Z, y los valores de X1 y X2.
- Cada vez que se ejecute el algoritmo, independientemente del FEV actual inicial aleatorio, la solución arrojada **SIEMPRE** debería ser la misma.

ENTREGABLE: El código fuente en Matlab o en Python.

ENTREGABLES

Las actividades solicitadas deben ser entregadas por el estudiante teniendo en cuenta las siguientes consideraciones:

- El informe a entregar consiste en lo indicado en los entregables de cada ejercicio.
- Se puede entregar en parejas.
- Plazo de entrega: 1 semana después de la publicación de la actividad. **Tenga en cuenta que esta actividad es un examen (no un laboratorio) y además es en parejas, por lo cual no habrá extensiones.**
- **Se utilizará la herramienta de plagio TURNITIN para verificar la originalidad de los códigos.** Por esta razón, se recomienda trabajar a conciencia y cualquier duda, consultar con el docente.
- **Orden recomendado de solución del examen:** resolver los puntos mas fáciles primero, es decir, el 1, el 3, y por último, el 2.