

Conceptos y herramientas para la resolución de problemas de optimización multiobjetivo

Métodos exactos

Sergio Nasmachnow, Diego Rossit

sergion@fing.edu.uy – diego.rossit@uns.edu.ar

Universidad de la República, Uruguay

Universidad Nacional del Sur-CONICET, Argentina



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY



Métodos exactos para problemas MO

Se basan en la optimización mono objetivo clásica de la programación matemática.

Resuelven varias veces el mismo problema, tratando de obtener soluciones con distinto trade-off entre los objetivos en cada iteración.

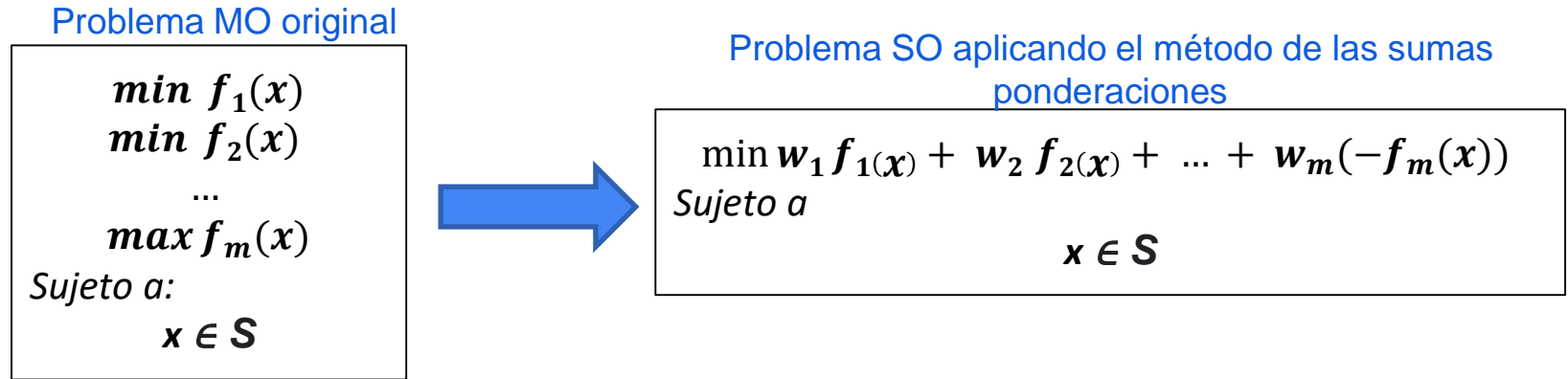
Durante el curso veremos:

- *método de las sumas ponderadas.*
- *método de las epsilon-restricciones.*
- *programación por metas.*
- *ordenamiento lexicográfico.*
- *métodos exactos y complejidad computacional.*
- *métodos exactos para problemas con aleatoriedad.*

Técnica de agregación - Métodos de las sumas ponderadas – Programación por compromiso

Métodos de sumas ponderadas

Se basa en transformar la función del problema MO (vectorial) en un problema mono-objetivo (escalar) mediante una agregación de los objetivos en una suma ponderada (Zadeh, 1963).



Para distintos valores del vector de pesos $\vec{w} = [w_1, w_2, \dots, w_m]$ se obtienen distintas soluciones del problema MO.

Notar que NO se modifican las restricciones del problema.

Ejemplo

$$\max f_1 = 0,4x_1 + 0,3x_2$$

$$\max f_2 = x_1$$

$$\max w_1 f_1 + w_2 f_2$$

sujeto a:

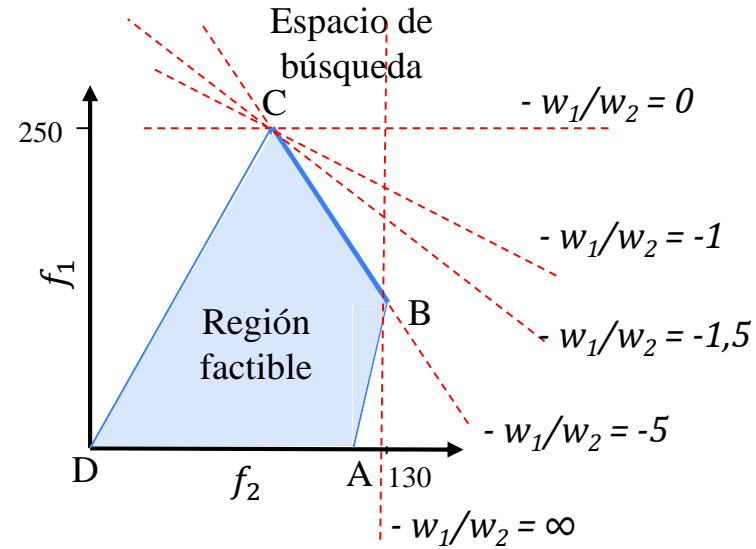
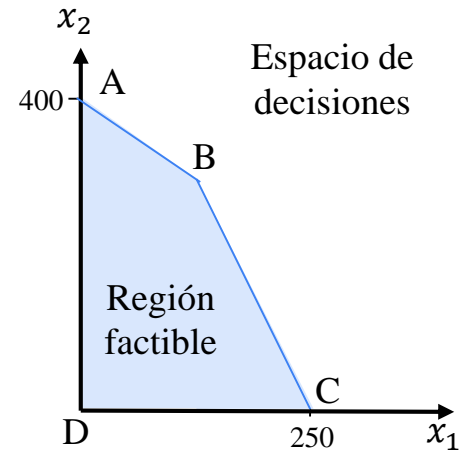
$$0,4x_1 + 0,3x_2 \leq 400$$

$$2x_1 + x_2 \leq 500$$

$$x \geq 0$$

Se resuelve el problema para $w_1 = 0,6$ y $w_2 = 0,4$ ($-w_1/w_2 = -1,5$):

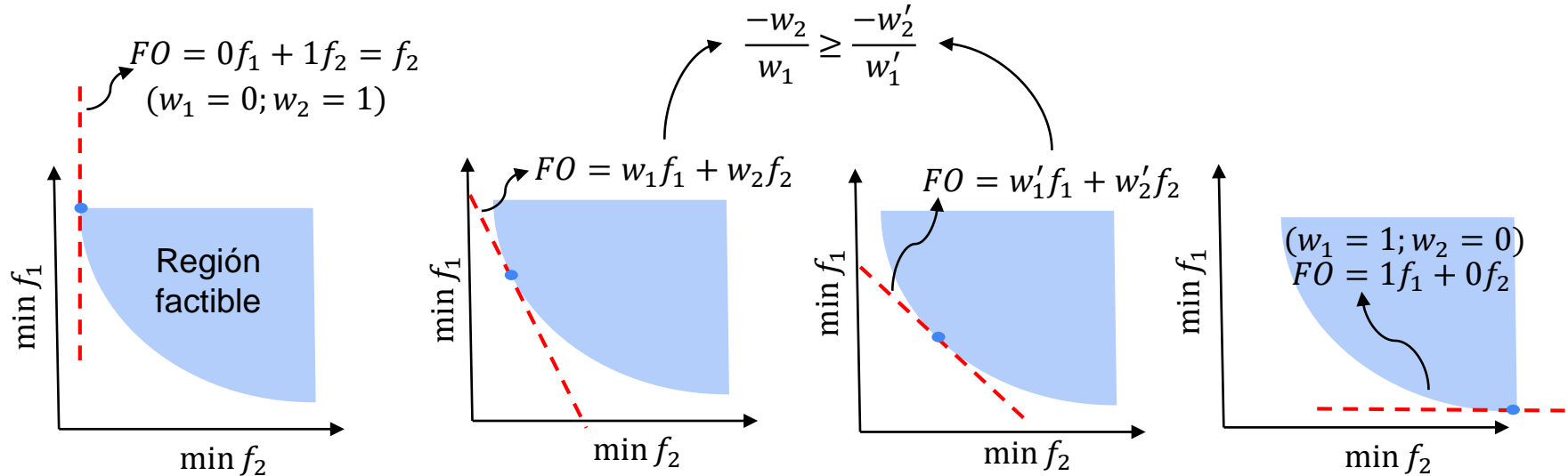
- $FO = 0,6 f_1 + 0,4 f_2 = 0,64 x_1 + 0,18 x_2$
- y la solución es $x_1 = 250$ y $x_2 = 0$ con $f_1 = 100$ y $f_2 = 250$ (punto C).



Métodos de suma ponderada según el enfoque de resolución

Enfoque de resolución a priori: se fija un vector de pesos de acuerdo a las preferencias del AD y se resuelve un único problema mono-objetivo.

Enfoque de resolución a posteriori: se resuelven varios problemas mono-objetivo con la utilización de distintos vectores de pesos para obtener el frente de Pareto (o una muestra representativa). Luego el AD elige una solución.



Obs: la suma de las componentes del vector de pesos no tienen que necesariamente sumar uno (aunque es algo usual).

Escalarización o normalización de objetivos

Los objetivos en general están en **unidades de medida** y **magnitudes** distintas. Por ende, para que los pesos manifiesten efectivamente la preferencia de los usuarios debe aplicarse una normalización o escalarización (reducir a una unidad y escala común).

En vez de usar directamente las funciones se mida la “cercanía” al punto ideal de cada objetivo (Zelany, 1974):

$$d_m(x) = \frac{f_m(x)}{f_m^*}$$

f_m^ : valor del objetivo m en el vector ideal.*

$$d_n(x) = \frac{|f_m(x) - f_m^*(x)|}{|f_{m^*}(x) - f_m^*(x)|}$$

f_{m^} : valor del objetivo m en el vector anti-ideal (debe tenerse en cuenta si estoy en un problema de max/min).*

En vez de usar como función objetivo vectorial $F(x) = [f_1(x), f_2(x), \dots, f_m(x)]$ se usa $D(x) = [d_1(x), d_2(x), \dots, d_m(x)]$.

Limitaciones del método de las ponderaciones

El método de las ponderaciones garantiza que resolviendo un problema mono-objetivo con pesos $w_i > 0$ se obtendrá un óptimo de Pareto del problema MO como solución (Emmerich y Deutz, 2018). Sin embargo, tiene algunos inconvenientes:

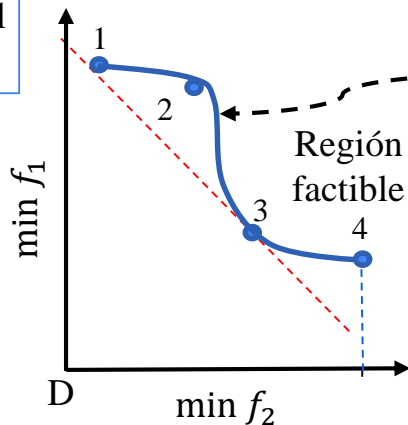
1. Se necesitan normalizar los objetivos antes de agregarlos.
2. Una distribución uniforme de los w_i no garantiza una distribución uniforme de las soluciones halladas en el frente de Pareto.
3. Distintos w_i pueden obtener la misma solución óptima -*Regiones de Indiferencia*- (gasto de recursos computacionales).
4. En problemas no lineales el método no puede hallar soluciones no respaldadas (*unsupported solutions*) -en regiones no convexas del frente de Pareto-.

Limitaciones del método de las ponderaciones

Una solución es no respaldada si existe una combinación lineal de soluciones que la domina. En los ejemplos:

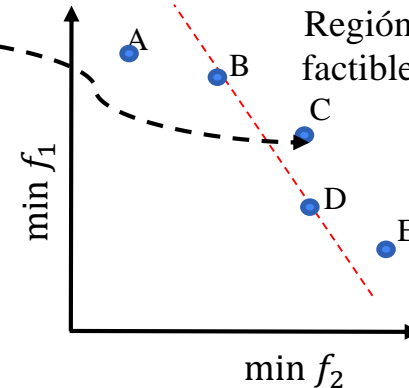
- la solución C es dominada por una combinación lineal de las soluciones B y E,
- y las soluciones de la porción de la curva 2-3 son dominadas por una combinación lineal de las soluciones 2 y 3.

Problema no lineal
 $FP = \text{curva } 1-4$



Soluciones óptimas
(de Pareto) no
respaldadas

Problema no lineal
entero $FP = \{A, B, C, D, E\}$



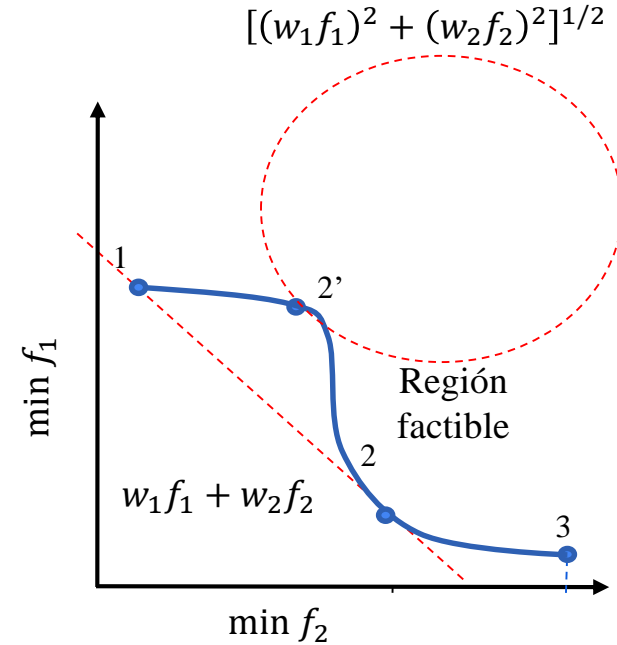
Técnicas de agregación avanzadas

Para hallar soluciones en zonas no respaldadas puede utilizarse una agregación no lineal

$$FO = \left\{ \sum_{i=1}^m (w_i f_i)^p \right\}^{1/p}, p > 1$$

Dificultades:

- determinar el grado p que permite hallar un punto de la región no convexa. Ryu y Min (2019) resuelven este problema en un proceso iterativo.
- se incorpora no linealidad al problema.



Método de las ponderaciones

Es un método ampliamente utilizado por su sencillez de aplicación.

No modifica las restricciones del problema.

Permite incorporar las preferencias del AD con relativa simplicidad (previa aplicación de una normalización de los objetivos).

Requiere normalización de los objetivos.

Usado como enfoque de resolución a posteriori puede generar soluciones repetidas.

Una distribución uniforme de vectores de pesos no implica una distribución uniforme de soluciones en el frente de Pareto.

En su versión básica no puede hallar soluciones en áreas no convexas del frente de Pareto.

Método de las epsilon-restricciones

Método de las epsilon-restricciones

Se optimiza una de las funciones objetivo y se considera el resto como restricciones acotadas por ciertos niveles permisibles a los que se denomina ϵ_i .

Problema MO original

$$\begin{array}{l} \min f_1(x) \\ \min f_2(x) \\ \dots \\ \max f_m(x) \\ \text{Sujeto a:} \\ \\ x \in S \end{array}$$



Problema SO aplicando
el método de las
 ϵ -restricciones

$$\begin{array}{l} \min f_1(x) \\ \text{Sujeto a:} \\ f_2(x) \leq \epsilon_2 \\ \dots \\ f_m(x) \geq \epsilon_m \\ x \in S \end{array}$$

Para distintos valores del vector de pesos ϵ_i se obtienen distintas soluciones.

Método de las ε -restricciones según enfoque de resolución

Enfoque de resolución a priori: El AD establece el objetivo de primer orden y cotas para los objetivos de segundo orden y se resuelve un único problema SO para devolver una única solución.

Enfoque de resolución a posteriori:

- Se selecciona el objetivo de primer nivel que constituirá la función objetivo.
- Se hallan los rangos de variación en el frente de Pareto de los otros $m-1$ objetivos (hallar el vector ideal y el vector anti-ideal)
- Se dividen los rangos de los $m-1$ objetivos en n intervalos.
- Se resuelven los $(m-1) * (n+1)$ problemas SO para encontrar $(m-1) * (n+1)$ soluciones.

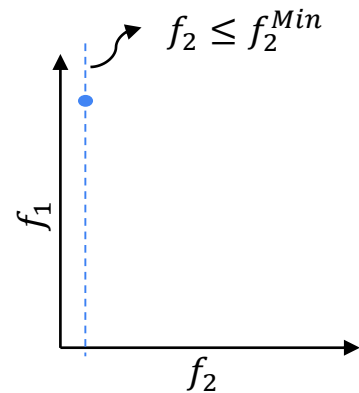
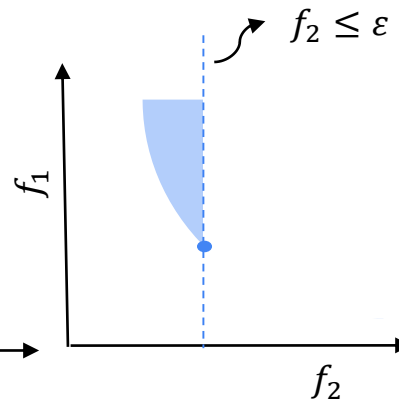
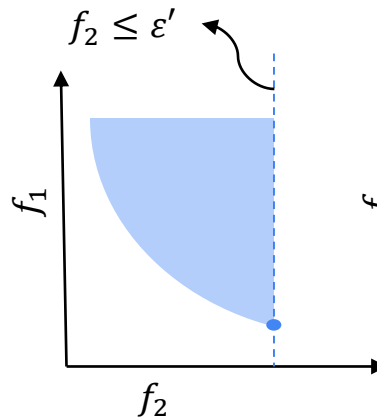
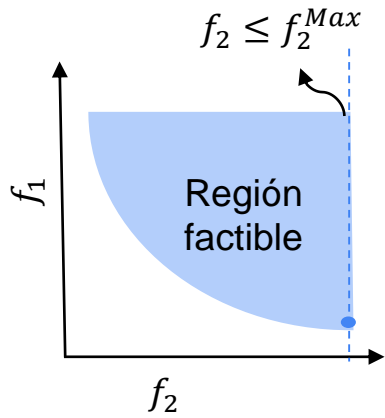
Método de las ε -restricciones

Se mantiene una de las funciones objetivo. El resto de las funciones objetivo pasan a ser restricciones.

Para distintos valores de *epsilon* se obtienen distintas soluciones

$$\begin{array}{l} \min f_1(x) \\ \text{Sujeto a:} \\ f_2(x) \leq \varepsilon_2 \\ x \in S \end{array}$$

(f_2 originalmente era objetivo de min)



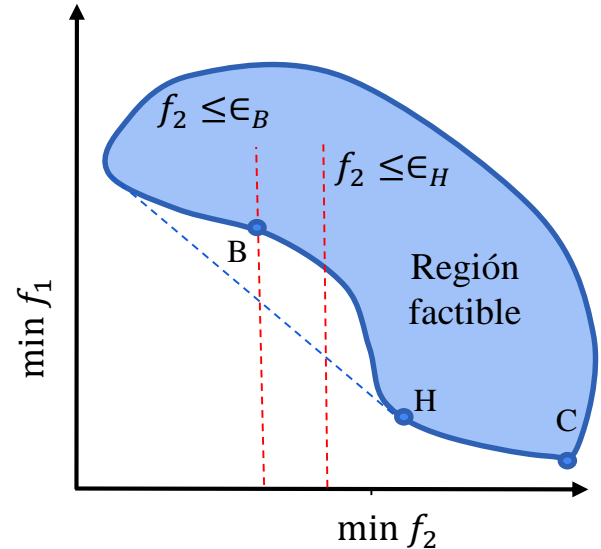
Ventajas/Desventajas método de las epsilon-restricciones

Ventajas:

- Permite obtener soluciones no respaldadas -en regiones no convexas del Frente de Pareto-.
- No es necesario normalizar las funciones objetivos (aunque es deseable conocer los vectores ideales y antideales para proponer extremos).
- Es más fácil controlar la distribución en el Frente de Pareto de las soluciones halladas.

Desventaja:

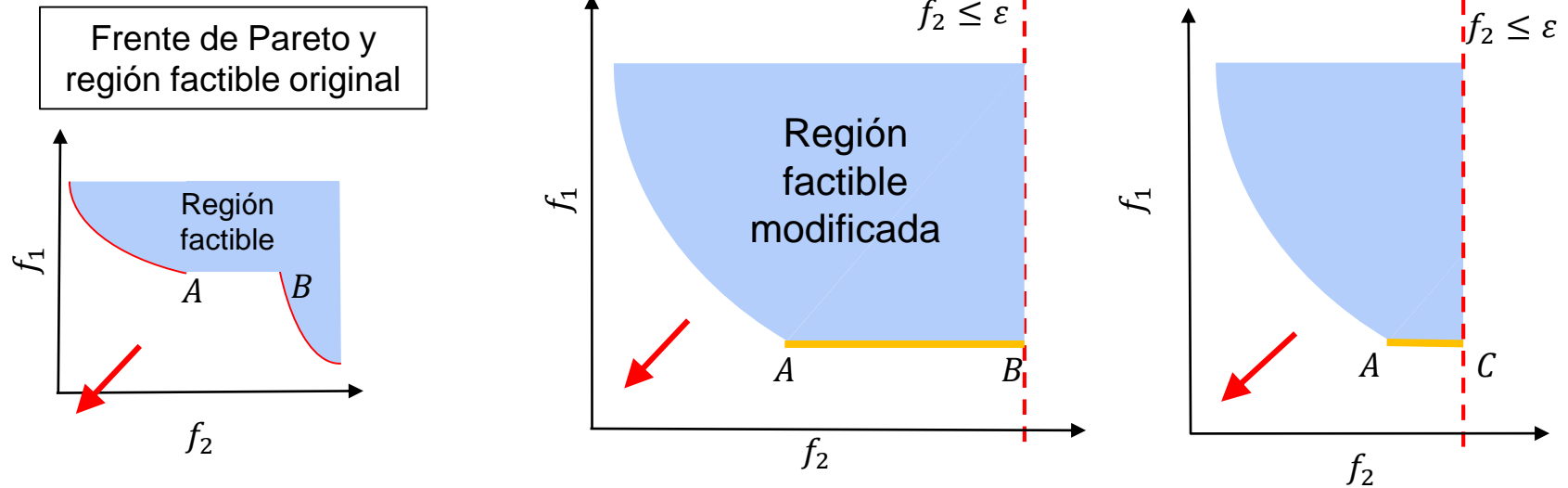
- Agrega restricciones al problema original lo cual puede volverlo más complejo de resolver.
- Puede hallar soluciones repetidas o dominadas cuando el problema SO tiene óptimos alternativos.



Soluciones dominadas y/o repetidas

Puede hallar soluciones repetidas o dominadas cuando el problema SO tiene óptimos alternativos dependiendo de la forma del FP. El segmento AB contiene todas las soluciones óptimas (alternativas) del problema SO. El método puede devolver cualquier solución sobre este segmento.

Sin embargo, el único óptimo de Pareto del segmento AB en el problema MOP, es el punto A.



Método de las ε -restricciones aumentado

Abordan el problema de obtener soluciones repetidas y/o no dominadas incluyendo una optimización lexicográfica en la función objetivo con los objetivos restringidos.

$$\text{Min } f_1(x) + \text{eps} \left(\frac{s_2}{r_2} + \frac{s_3}{r_3} + \dots + \frac{s_m}{r_m} \right)$$

sujeto a

$$f_n(x) - s_n = \varepsilon_n, \quad n = 2, \dots, m$$

$$x \in X, s \geq 0$$

$$\text{Min } f_1(x) + \text{eps} \left(\frac{s_2}{r_2} + 10^{-1} \frac{s_3}{r_3} + \dots + 10^{-(m-2)} \frac{s_m}{r_m} \right)$$

sujeto a

$$f_n(x) - s_n = \varepsilon_n, \quad n = 2, \dots, m$$

$$x \in X, s \geq 0$$

r_n es el rango del objetivo (idealmente en el Frente de Pareto). s_n es la variable de holgura de la restricción (convierte desigualdad en igualdad).

Método de las ε -restricciones aumentado

Incorpora optimización lexicográfica para obtener rango de los objetivos.

La información de las variables de holgura (s_n) permite conocer si en la siguiente iteración se encontrará una solución repetida comparándosela contra el tamaño del paso.

Existen otras variantes del método de las epsilon-restricciones:

- SAUGMECON (Zhang y Reimann, 2014)
- AUGMECON-R (Nikas et al., 2020)

Consideraciones del Método de las ε -restricciones

Resuelve el problema del método de la suma ponderada que no puede encontrar soluciones en áreas no convexas.

No se requiere normalizar las funciones objetivo.

Puede controlar mejor la distribución de soluciones en el FP.

Incorpora restricciones al problema original lo cual puede aumentar la dificultad computacional.

En su versión básica, dependiendo de la forma del FP o si el vector nadir no está adecuadamente calculado puede encontrar soluciones no dominadas o repetidas.

Programación por metas

Programación por metas

El concepto no es encontrar múltiples soluciones (con distinto trade-off) sino encontrar una (única) solución que se aproxime a las metas (valores deseados) de los objetivos definidos previamente por el AD (enfoque de resolución a priori).

Problema MO original

$$\begin{array}{l} \min f_1(x) \\ \min f_2(x) \\ \dots \\ \max f_m(x) \\ \\ \text{Sujeto a:} \\ \\ x \in S \end{array}$$

Metas del AD (ψ_n)



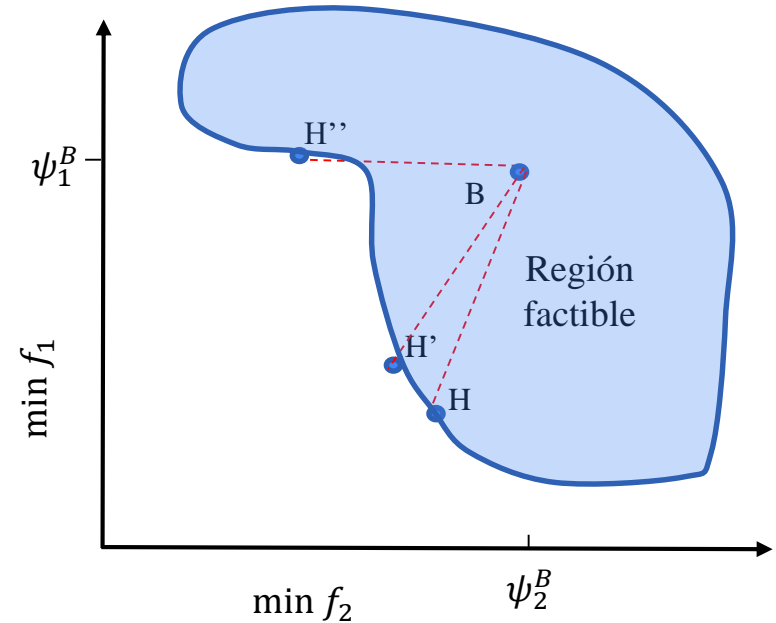
Problema SO aplicando programación por metas

$$\begin{array}{l} \min \sum_{n=1}^m (w_n^- d_n^- + w_n^+ d_n^+) \\ \\ \text{sujeto a} \\ \\ f_n(x) + d_n^- - d_n^+ = \psi_n, \quad n = 1, \dots, m \\ \\ x \in S \\ d \geq 0 \end{array}$$

d_n^- y d_n^+ son las desviaciones negativas y positivas de la meta deseada para el objetivo n . w_n^- y w_n^+ son los pesos asignados a esas desviaciones.

Desventajas del método de programación por metas

1. Necesidad de normalizar los objetivos/desviaciones antes de agregarlas.
2. Requiere conocimiento previo del problema por parte del AD para proponer las metas.
3. Si las metas propuestas por el AD no son lo suficientemente ambiciosas puede obtenerse una solución subóptima (en términos de optimalidad de Pareto).



Mejoras al método de programación por metas

Se incorpora un control a la máxima desviación (Romero, 1991):

ϕ controla la máxima desviación entre todas las metas. k_n es un parámetro de normalización y λ es un parámetro de control tal que $\lambda \in [0,1]$.

$$\text{Min } (1 - \lambda)\phi + \lambda \left[\sum_{n=1}^m w_n \left(\frac{d_n^- - d_n^+}{k_n} \right) \right]$$

sujeto a

$$f_n(x) + d_n^- - d_n^+ = \psi_n, \quad n = 1, \dots, m$$

$$\frac{d_n^- - d_n^+}{k_n} \leq \phi, \quad n = 1, \dots, m$$

$$x \in S \\ d, \phi \geq 0$$

- $\lambda=0$: solución balanceada que minimiza la desviación máxima.
- $\lambda=1$: solución priorizando la ponderación del AD.
- $\lambda \in (0,1)$: soluciones intermedias a los dos casos extremos.

Ordenamiento lexicográfico

Ordenamiento lexicográfico

El concepto no es encontrar múltiples soluciones (con distinto trade-off) sino encontrar una (única) solución basado en que el AD pueda establecer algún orden de importancia/prioridad entre los objetivos.

Las funciones objetivo se jerarquizan en orden de importancia.

Se resuelven tantos problemas de optimización SO como funciones objetivos halla.

Los problemas SO se resuelven en el orden de la jerarquía, agregando restricciones ficticias para las funciones objetivo resueltas previamente no se deterioren.

Ordenamiento lexicográfico

Secuencias de problemas SO (para hallar una única solución) con el método lexicográfico

Problema MO original

$$\begin{array}{l} \min f_1(x) \\ \min f_2(x) \\ \dots \\ \max f_m(x) \\ \text{sujeto a:} \\ x \in S \end{array}$$

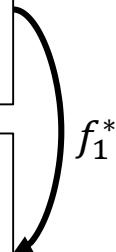
Orden dado a los objetivos por el AD
 $f_1 > f_2 > \dots > f_m$



$$\begin{array}{l} \min f_1 \\ \text{sujeto a:} \\ x \in S \end{array}$$

$$\begin{array}{l} \min f_2 \\ \text{sujeto a:} \\ f_1 = f_1^* \\ x \in S \end{array}$$

$$\begin{array}{l} \dots \\ \max f_m \\ \text{sujeto a:} \\ f_1 = f_1^* \\ f_2 = f_2^* \\ \dots \\ f_{m-1} = f_{m-1}^* \\ x \in S \end{array}$$



Consideraciones del ordenamiento lexicográfico

- Permite obtener una única solución del frente de Pareto.
- El esfuerzo computacional puede variar dependiendo del orden elegido en los objetivos. La solución también varía según este orden.
- Es útil cuando el AD puede determinar un orden jerárquico de los objetivos que es coherente con sus preferencias y no está interesado en explorar el trade-off entre los objetivos.
- Los problemas SO se van volviendo cada vez más restringidos lo cual puede volverlo más complejo de resolver. Alternativas:
 - Utilizar la estructura de la solución del problema inmediato anterior para inicializar la siguiente etapa -*warm start*-.
 - Permitir cierto relajamiento de los objetivos restringidos.

Formas de hallar/aproximar el vector nadir

¿Por qué es deseable conocer (o tener una buena aproximación) del vector nadir?

- Brinda información al AD sobre el rango de los objetivos dentro del frente de Pareto para
 - en enfoques de resolución a priori:
 - definir cotas realistas en método de las ε -restricciones.
 - definir metas realistas en método de programación por metas.
 - evaluar soluciones.
- También son necesarios para el enfoque de resolución a posteriori:
 - en el método de la suma ponderada para una mejor normalización de los objetivos.
 - en el método de las ε -restricciones para evitar soluciones no dominadas.

¿Cómo hallar el vector nadir?

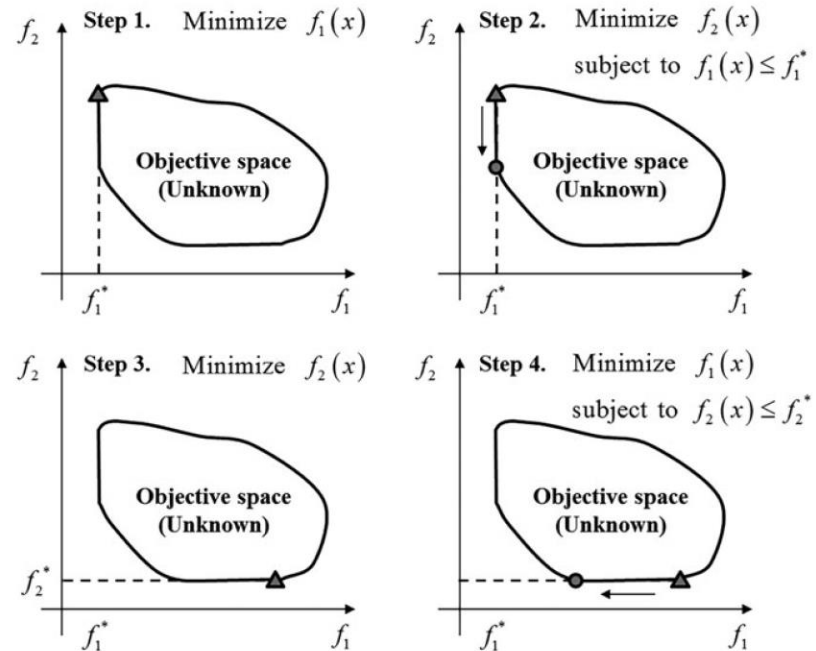
Aplico optimización lexicográfica para todos los órdenes de jerarquía posibles de los objetivos. Con dos objetivos:

1. optimizo uno de los objetivos y obtengo su valor libre.
2. incluyo una restricción al modelo para que dicho objetivo no empeore.
3. optimizo el segundo objetivo.
4. Realizo lo mismo cambiando el orden de jerarquía de los objetivos.

Si converge tengo el vector nadir.

Posible problema: el paso 3 puede ser un problema muy restringido \rightarrow dificultad computacional.

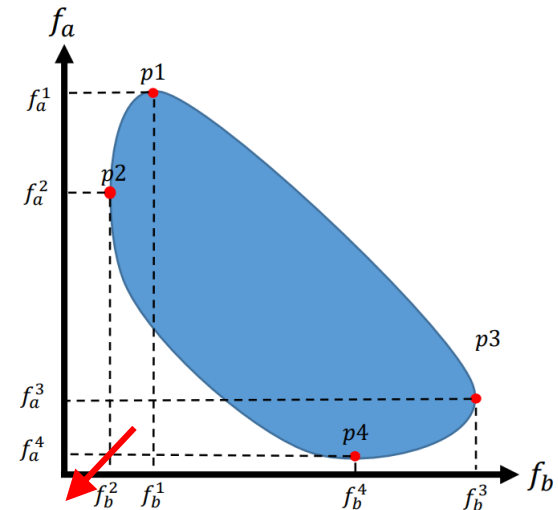
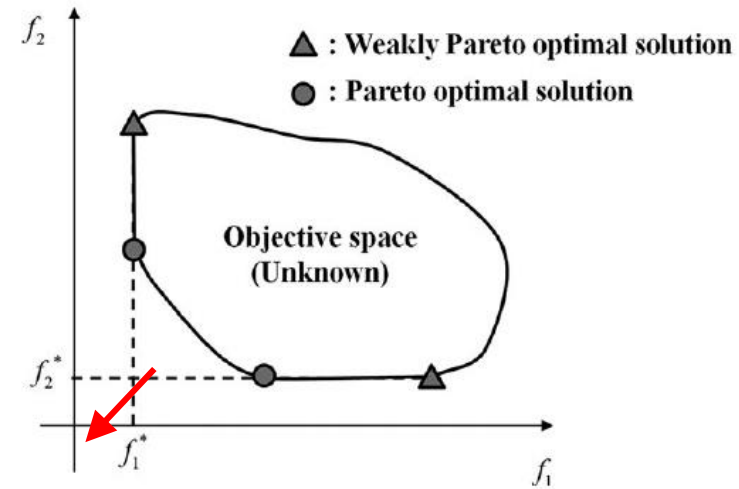
Ayuda: utilizar la estructura de la solución del paso 1 para inicializar el paso 3 (*warm start*).



Fuente: Ryu y Ming (2019).

¿Cómo aproximar el vector nadir?

1. Modifico la optimización lexicográfica e incorporo un relajamiento a los objetivos restringidos (la restricción del paso 2 slide anterior). *En vez de que el objetivo no empeore que no lo haga en más de $x\%$. Con esto ya estoy aproximando el vector nadir.*
2. Lo tomo de la matriz de pagos utilizada para hallar el vector ideal (optimización mono-objetivo de cada objetivo).
3. Optimizo el inverso de cada objetivo.



¿Cómo aproximar el vector nadir?

4. Usar en método de la suma ponderada con un vector de pesos desbalanceado:

- Hallar la matriz de pagos.
- Aplicar el método de la suma ponderada m veces dando en cada aplicación una mayor prioridad a un objetivo distinto de los m objetivos. Utilizo los valores de la matriz de la pagos para normalizar.

$$\min \left[w_{k'} \frac{f_{k'} - f_{k'}^*}{f_{k'^*} - f_{k'}^*} + \sum_{k \in K, k \neq k'} \left(w_k \frac{f_k - f_k^*}{f_{k^*} - f_k^*} \right) \right], w_{k'} \gg w_k$$

- Vuelvo a aplicar el método de la suma ponderada desbalanceado mejorando la aproximación al vector nadir con el resultado anterior.

Métodos exactos y complejidad computacional

Métodos exactos y complejidad computacional

Los métodos exactos para MOP están basados en programación matemática, por ende están afectados por las mismas reglas que los problemas mono-objetivo en cuanto a complejidad computacional. En general:

| | con variables de decisión continuas | con variables de decisión discontinuas (discretas o continuas a tramos) |
|------------------------------|--|---|
| Problemas lineales | Relativamente sencillo de resolver | Relativamente complejo de resolver |
| Problemas no lineales | Relativamente complejo de resolver | Relativamente muy complejo de resolver |

Hay excepciones para algunos problemas específicos. Por ejemplo, el problema del árbol de mínima expansión se puede modelar como un problema entero pero existe el algoritmo de Prim que lo resuelve eficientemente.

Para abordar los problemas complejos de resolver se utilizan herramientas heurísticas. En estos casos los métodos exactos siguen siendo útiles para validación en instancias pequeñas.

Extensión de los métodos exactos para MOP con aleatoriedad

Problemas con riesgo o incertidumbre

El proceso de toma de decisiones puede ocurrir en condiciones de:

- certeza. Los parámetros son conocidos y constantes. Se utilizan modelos determinísticos.
- riesgo. Hay parámetros aleatorios pero existe suficiente información para estimar su distribución probabilística. Se utilizan *modelos estocásticos* que encuentran soluciones que **optimizan el valor esperado** sobre el conjunto de escenarios posibles.
- incertidumbre. Hay parámetros aleatorios pero no se dispone de suficiente información para estimar su distribución probabilística. Se utilizan *modelos robustos* que encuentran soluciones relativamente “buenas” que **mantienen la factibilidad en el peor caso posible**.

Extensión de los métodos exactos para MO con riesgo o incertidumbre

Los métodos exactos para MO vistos anteriormente fueron presentados para modelos determinísticos.

Sin embargo, los modelos robustos y estocásticos se basan en transformar los parámetros con aleatoriedad en parámetros determinísticos.

Por ende, los métodos exactos para MO vistos pueden extenderse naturalmente a problemas robustos y estocásticos.

Métodos exactos en problemas con incertidumbre (modelos robustos)

Un modelo robusto es un modelo determinístico donde al problema original (o nominal) se agregan ecuaciones, parámetros y variables adicionales. Por ende, se pueden aplicar los métodos vistos anteriormente.

Modelo en condiciones de certeza

$$\begin{aligned} & \text{Max } \sum_j c_j x_j \\ & \text{sujeto a} \\ & \sum_j a_{ij} x_j \leq b_i, \forall i \\ & x \geq 0 \end{aligned}$$

Se incluye incertidumbre

Considerar que a_{ij} es un parámetro aleatorio que varía (uniformemente) en el rango $[\bar{a}_{ij} - \hat{a}_{ij}; \bar{a}_{ij} + \hat{a}_{ij}]$.

Modelo en condiciones de incertidumbre según método de Bertsimas & Sim (2004) (es nuevamente un modelo determinístico)

$$\begin{aligned} & \text{Max } \sum_j c_j x_j \\ & \text{sujeto a} \\ & \sum_j \bar{a}_{ij} x_j + z_i \Gamma_i + \sum_{j \in J_i} p_{ij} \leq b_i, \forall i \\ & z_i + p_{ij} \geq \hat{a}_{ij} y_j, \quad \forall i, j \in J_i \\ & -y_j \leq x_j \leq y_j \\ & p, z, x, y \geq 0 \end{aligned}$$

Γ_i : nivel de conservadurismo $0 \leq \Gamma_i \leq |J_i|$

J_i : conjunto de parámetros que pueden variar en la ec. i .

Métodos exactos en problemas con riesgo (modelos estocásticos)

Los modelos estocásticos son más complejos que los robustos. En general se dividen en dos partes:

- variables que deben definirse en un primer paso (en el ejemplo las variables x).
- variables que dependen del valor de las primeras y que se definen en un segundo paso (en el ejemplo las variables y). Estas dependen del escenario.

$$\min c' y' + E[Q(x, \xi')]$$

sujeto a

$$Ax \leq b_i, \forall i \\ x \geq 0$$

donde $E[\cdot]$ es el operador de valor esperado y $Q(x, \xi)$ es el problema de optimización de segundo orden en el escenario ξ .

$$\min q'_\xi y'$$

sujeto a

$$Wy = h'_\xi - t'_\xi x \\ y \geq 0$$

Métodos exactos en problemas con riesgo (modelos estocásticos)

Hay distintos enfoques para resolver modelos estocásticos de programación matemática que dan lugar a modelos determinísticos donde pueden aplicarse las herramientas vistas. Algunos son:

- **Formulación extendida.** Se incorpora todos los problemas $Q(x, \xi)$ - asociados al escenario- al problema de la primera etapa. La resultante es un modelo determinístico (ampliado). Puede aumentar mucho la complejidad computacional si hay muchos escenarios.
- **Descomposición de Benders o L-shape.** Similar al anterior pero en vez de incorporar todos los $Q(x, \xi)$ al problema de la primera etapa se incorpora sólo cierta información necesaria a medida que se necesita. Resulta en modelos determinísticos.
- **Progressive Hedging.** Se basa en descomposición de Lagrange. También da lugar a modelos determinísticos.

Métodos exactos para MO en problemas con parámetros aleatorios

Lo importante para efectos del curso es recordar:

- los métodos exactos para MO vistos pueden extenderse para problemas donde exista aleatoriedad ya que tanto los modelos robustos –para condiciones de incertidumbre- como estocásticos – para condiciones de riesgo- se basan en problemas determinísticos.
- la aplicación de métodos exactos para MO en un problema con aleatoriedad puede dar lugar a un modelo con alta complejidad computacional.

Conceptos y herramientas para la resolución de problemas de optimización multiobjetivo

Métodos exactos

Sergio Nesmachnow, Diego Rossit

Universidad de la República, Uruguay

Universidad Nacional del Sur-CONICET, Argentina



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Bibliografía

- Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations research*, 52(1), 35-53.
- Birge, J. R., & Louveaux, F. (2011). Introduction to stochastic programming. Springer Science & Business Media.
- Caramia, M., & Dell'Olmo, P. (2008). Multi-objective management in freight logistics (pp. 11-36). London: Springer.
- Charnes, A., Cooper, W., & Ferguson, R. (1955). Optimal estimation of executive compensation by linear programming. *Management Science*, 1(2), 138-151.
- Emmerich, M., & Deutz, A. (2018). A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17(3), 585-609.
- Mausser, H. (2006, August). Normalization and other topics in multi-objective optimization. In Fields-MITACS Industrial Problems Workshop (p. 89).
- Mavrotas, G. (2009). Effective implementation of the ε -constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2), 455-465.
- Mavrotas, G., & Florios, K. (2013). An improved version of the augmented ε -constraint method (AUGMECON2) for finding the exact pareto set in multi-objective integer programming problems. *Applied Mathematics and Computation*, 219(18), 9652-9669.

Bibliografía

Nikas, A., Fountoulakis, A., Forouli, A., & Doukas, H. (2020). A robust augmented ε -constraint method (AUGMECON-R) for finding exact solutions of multi-objective linear programming problems. *Operational Research*, 1-42.

Romero, C. (2014). *Handbook of critical issues in goal programming*. Elsevier.

Rossit, D., Toutouh, J., & Nesmachnow, S. (2020). Exact and heuristic approaches for multi-objective garbage accumulation points location in real scenarios. *Waste Management*, 105, 467-481.

Ryu, N., & Min, S. (2019). Multiobjective optimization with an adaptive weight determination scheme using the concept of hyperplane. *International Journal for Numerical Methods in Engineering*, 118(6), 303-319.

Zadeh, L. (1963). Optimality and non-scalar-valued performance criteria. *IEEE transactions on Automatic Control*, 8(1), 59-60.

Zhang, W., & Reimann, M. (2014). A simple augmented ε -constraint method for multi-objective mathematical integer programming problems. *European Journal of Operational Research*, 234(1), 15-24.

Zelany, M. (1974). A concept of compromise solutions and the method of the displaced ideal. *Computers & Operations Research*, 1(3-4), 479-496.

Breve Tutorial para usar Pyomo
(no hay que preocuparse si hay inconvenientes
ya que lo repasaremos en la siguiente clase).

¿Cómo usar Pyomo?

En el curso utilizaremos Pyomo (basado en Python) como entorno de modelado y glpk como solver. Se puede usar:

- de forma colaborativo

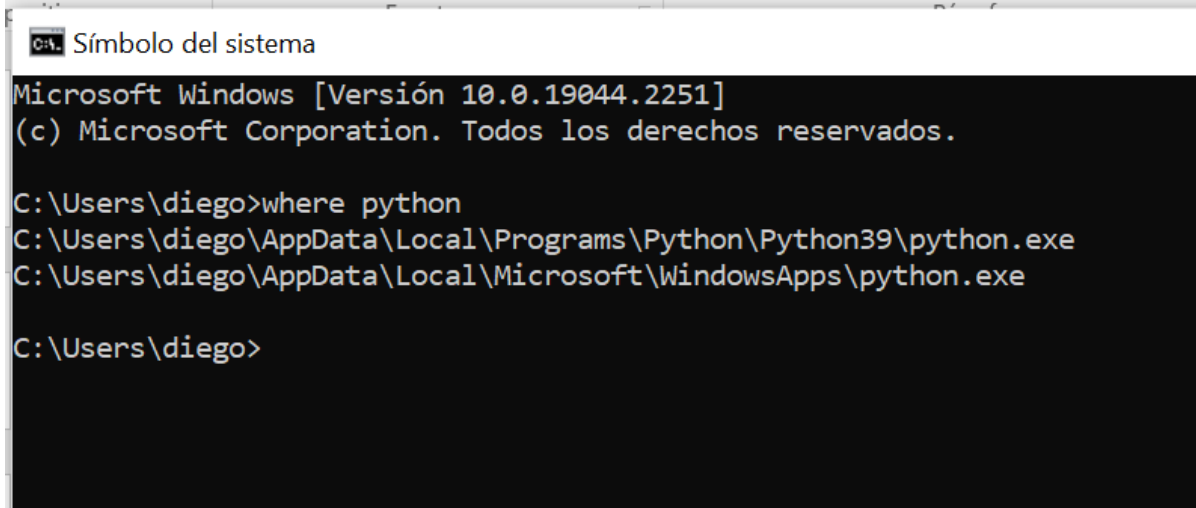
Para usar Google Colab seguir los pasos en

<https://colab.research.google.com/github/jckantor/ND-Pyomo-Cookbook/blob/master/notebooks/01.02-Running-Pyomo-on-Google-Colab.ipynb>

- en el ordenador local

A través de Visual Studio Code. A continuación se enumeran los pasos.

1. Averiguar si Python está instalado.



```
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.19044.2251]
(c) Microsoft Corporation. Todos los derechos reservados.

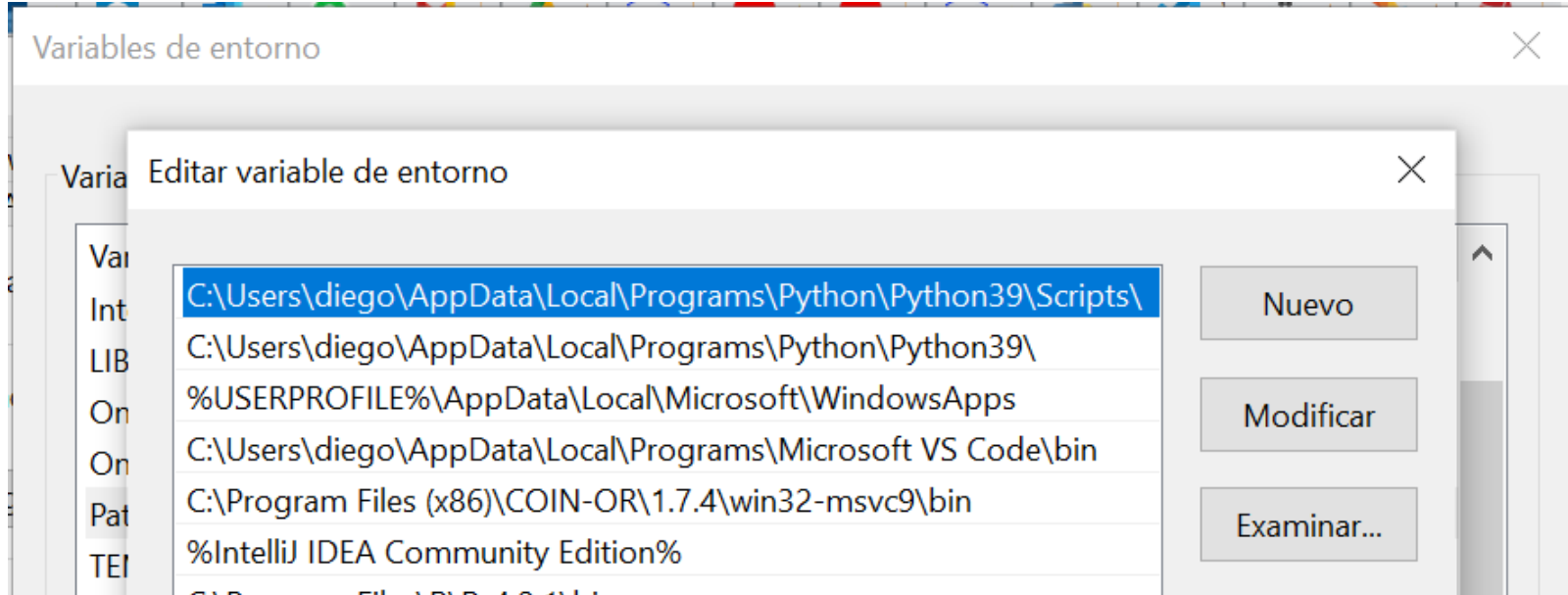
C:\Users\diego>where python
C:\Users\diego\AppData\Local\Programs\Python\Python39\python.exe
C:\Users\diego\AppData\Local\Microsoft\WindowsApps\python.exe

C:\Users\diego>
```

Sino está instalado, instalarlo siguiendo los pasos <https://www.python.org/downloads/>. Se recomienda no instalar ninguna versión en fase beta.

Agregar la variable de entorno a el interpretador de Python (python.exe) <https://realpython.com/add-python-to-path/> y a los Scripts

1. Averiguar si Python está instalado.



2. Instalar Visual Studio Code

Link de descarga: <https://code.visualstudio.com/>

3. Instalar *pyomo* y *matplotlib*

- Instrucciones pyomo: <https://pyomo.readthedocs.io/en/stable/installation.html>
 - comandos: “pip install pyomo” o “python -m pip install pyomo”
- Instrucciones matplotlib:
<https://matplotlib.org/stable/users/installing/index.html>
 - comandos: “pip install matplotlib” o “python -m pip install matplotlib”

4. Instalar glpk

Importante si ya posee otro *solver* instalado de forma independiente (*standalone*) puede utilizarlo mientras tenga vinculación con *pyomo* (https://pyomo.readthedocs.io/en/stable/solving_pyomo_models.html).

Tener en cuenta que tener GAMS o AMPL no implica tener una versión *standalone*.

- Para CPLEX seguir los pasos <https://www.ibm.com/docs/en/icos/20.1.0?topic=cplex-setting-up-python-api>

Para instalar *glpk* descargar el solver desde <https://winglpk.sourceforge.net/> y luego configurar la variable de entorno del sistema. Por ejemplo en Windows 64 bits Path = GLPK_HOME\winglpk-4.65\glpk-4.65\w64

Conceptos y herramientas para la resolución de problemas de optimización multiobjetivo

Métodos exactos

Sergio Nasmachnow, Diego Rossit

sergion@fing.edu.uy – diego.rossit@uns.edu.ar

Universidad de la República, Uruguay

Universidad Nacional del Sur-CONICET, Argentina



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

