UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

EDA: *Estimation Distribution Algorithms*

Alan Reyes-Figueroa

Inteligencia Artificial (Aula 15) 08.marzo.2022

EDA

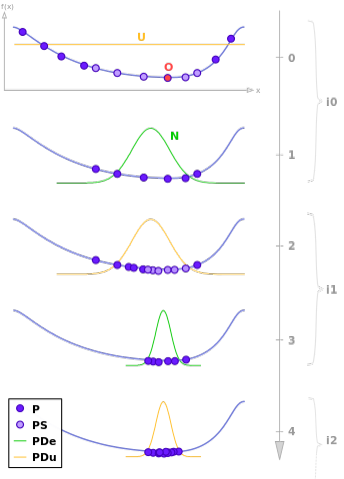
Los **algoritmos de estimación de distribución** (EDA), son métodos de optimización estocástica que guían la búsqueda del óptimo mediante la construcción y muestreo de modelos probabilísticos.
Estos modelos de probabilidad explícitos, buscan regiones de soluciones candidatas prometedoras.

La optimización se ve como una serie de actualizaciones incrementales de un modelo probabilístico, comenzando con el modelo que codifica una priori no informativa sobre soluciones admisibles y terminando con el modelo que genera solo los óptimos globales.

Los EDA pertenecen a la clase de algoritmos evolutivos. La principal diferencia entre los EDA y la mayoría de los algoritmos evolutivos convencionales es que los algoritmos evolutivos generan nuevas soluciones candidatas utilizando una distribución implícita.

El procedimiento general de una EDA se describe a continuación:

EDA



1. Para cada iteración *t*, se construye una muestra de la población $P(t)$ (en el espacio de puntos), bajo cierta distribución $p(\theta)$.
2. De dicha muestra, se priorizan u ordenan (*ranking*) los datos en la muestra, atendiendo a cierta función objetivo, y se seleccionan los mejores.
3. Los parámetros de la distribución $p(\theta)$ se estiman de nuevo, en función de la submuestra selecta.
4. En la siguiente iteración, se vuelve a muestrear datos con esta nueva distribución actualizada $p(\theta)$. Este proceso se repite hasta algún criterio de paro.

EDA

Algoritmo: (EDA General)
Inputs: Función objetivo f, y M un modelo de distribución sobre el espacio de soluciones admisibles.
Outputs: Solución subóptima **x**.

Initialize model $M(0)$
$t = 0$
Repeat until stop criteria holds:
    Define $P(t)$: generate $N > 0$ candidate solutions by sampling $M(t)$,
    Evaluate all candidate solutions in $P(t)$ **under the function** f, and rank,
    Select subsample $S(P(t))$ of best elements,
    Define $M(t+1)$ by adjust the model $(P, f, M(t))$,
    $t = t + 1$.

EDA

Ventajas:

El uso de EDAs

- permite resolver de manera factible problemas de optimización que eran notoriamente difíciles para la mayoría de los algoritmos evolutivos convencionales y las técnicas de optimización tradicionales.
- proporciona una serie de modelos probabilísticos que revelan información estructural sobre el problema que se está resolviendo.
  Esta información, a su vez, se puede utilizar para diseñar operadores de vecindad específicos del problema para la búsqueda local, para sesgar las ejecuciones futuras de EDA en un problema similar o para crear un modelo computacional eficiente del problema.

EDA

Tipos de EDA:

- Modelos Univariados.
  - UMDA (*Univariate Marginal Distribution Algorithm*)
  - PBIL (*Population-based Incremental Learning*)
  - cGA (*Compact Genetic Algorithm*)
- Modelos Bivariados.
  - MIMIC (*Mutual Information Maximization Input Clustering*)
  - BMDA (*Bivariate Marginal Distribution Algorithm*)
- Modelos Univariados.
  - eCGA (*Extended Compact Genetic Algorithm*)
  - BOA (*Bayesian Optimization Algorithm*)
  - LTGA (*Linkage-tree Genetic Algorithm*)
  - Graphical Models

EDA

Describimos ahora algunos de estos modelos. Se asume siempre una población $P(t)$ en la generación t, un operador de selección α, y un operador para generar un modelo β.

UMDA (**Univariate Marginal Distribution Algorithm**) Las EDA más simples asumen que las variables de decisión son independientes, es decir, $p(X_1, X_2) = p(X_1)\,p(X_2)$. Por lo tanto, las EDA univariadas se basan solo en estadísticas univariadas y las distribuciones multivariadas deben factorizarse como producto de n distribuciones univariadas

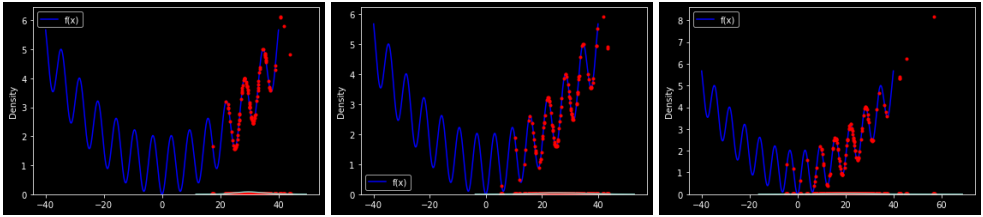
$$D_{\text{Univariate}} = p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i).$$

Estas factoraciones se utilizan en muchas EDA diferentes, a continuación describimos algunas de ellas.

El UMDA usa un operador α_{UMDA} para estimar probabilidades marginales de una población selecta $S(P(t))$ de k elementos es α_{UMDA} produce probabilidades:

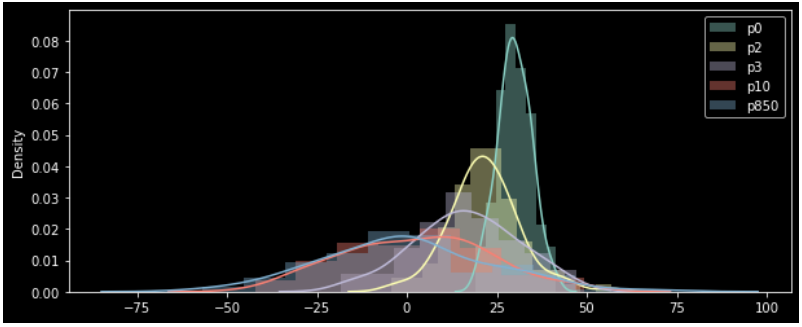
$$p_{t+1}(X_i) = \frac{1}{k} \sum_{x \in S(P(t))} x_i, \text{ para } i = 1, 2, \ldots, n.$$

Ejemplos UMDA



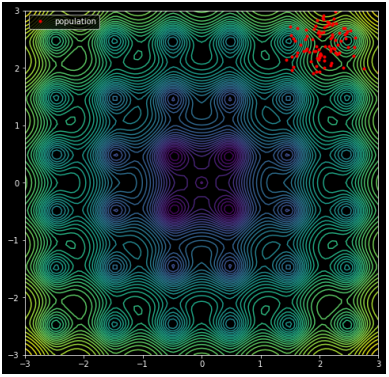
UMDA 1-dimensional, poblaciones 0, 1 y 2. Observe cómo en cada iteración, la población se "mueve" hacia donde está el mínimo de la función objetivo f.

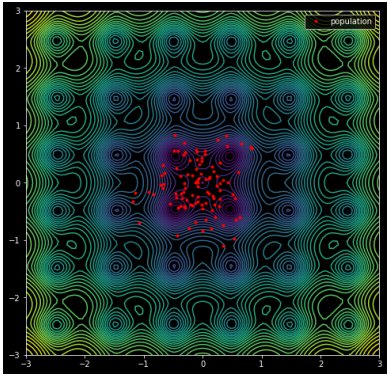
Ejemplos UMDA



UMDA 1-dimensional, distribución de algunas poblaciones iniciales. Observe cómo en cada iteración k, el centro o media de la distribución p_k se "mueve" hacia donde está el mínimo de la función objetivo f.

Ejemplos UMDA





UMDA 2-dimensional, poblaciones 0 y 880. De nuevo, en cada iteración la población se "mueve" hacia donde está el mínimo de la función objetivo f.

MIMIC: (*Mutual Information Maximization Input Clustering*)

El MIMIC factoriza la distribución de probabilidad conjunta en un modelo en forma de cadena que representa sucesivas dependencias entre variables. Encuentra una permutación de las variables de decisión $r : i \to j$, tal que $x_{r(1)}, x_{r(2)}, \ldots, x_{r(n)}$ minimiza la divergencia de Kullback-Leibler en relación con la distribución de probabilidad verdadera, $\pi_{r(i+1)} = \{X_{r(i)}\}$.

MIMIC modela una distribución de la forma

$$p_{t+1}(X_1, \ldots, X_n) = p_t(X_{r(n)}) \prod_{i=1}^{n-1} p_t(X_{r(i)} \mid X_{r(i+1)}).$$

Las nuevas soluciones se muestrean desde la variable más a la izquierda a la más a la derecha, la primera se genera de forma independiente y las otras según probabilidades condicionales. MIMIC utiliza poblaciones concretas de la siguiente manera $P(t+1) = \beta_\mu \circ \alpha_{\text{MIMIC}} \circ S(P(t))$.