

Łańcuchy Markowa Rewolucja Monte Carlo

Kacper Wnęk

March 2023

Plan Prezentacji

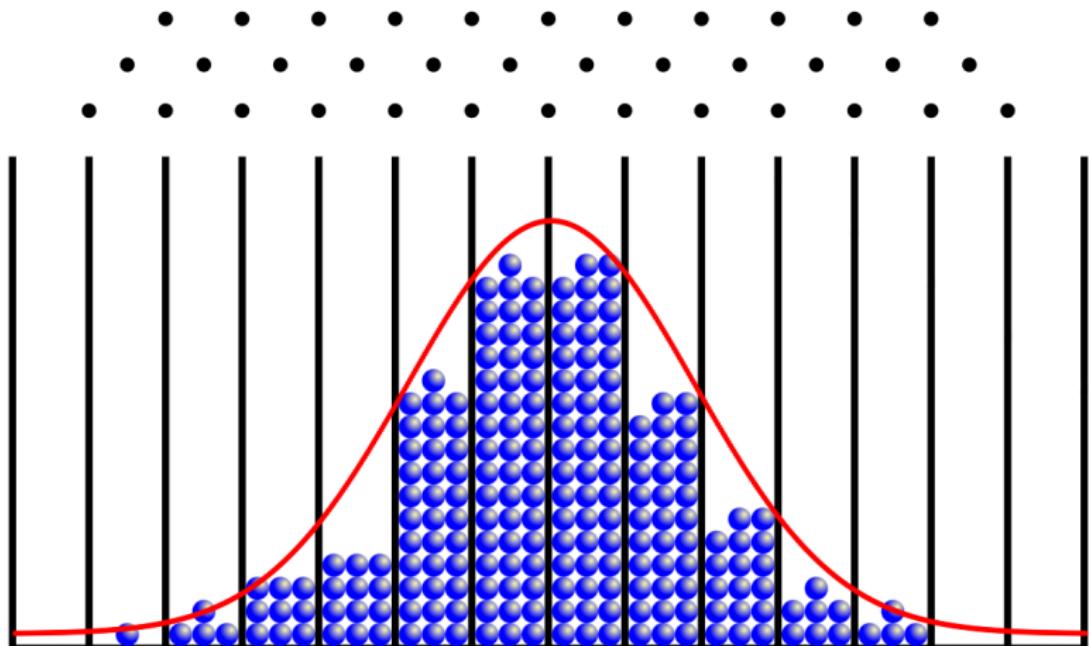
- Przypomnienie RP
- Łańcuchy Markowa
- Metoda Monte Carlo
- MCMC-Łańcuchy Markowa Monte Carlo

Koncepcja wartości oczekiwanej Bernouliego



Wniosek: Jeśli obserwacje zdarzeń będą kontynuowane w nieskończoność to okaże się, że wszystkim na świecie rządzą precyzyjne stosunki i zmiany podlegające stałym prawom

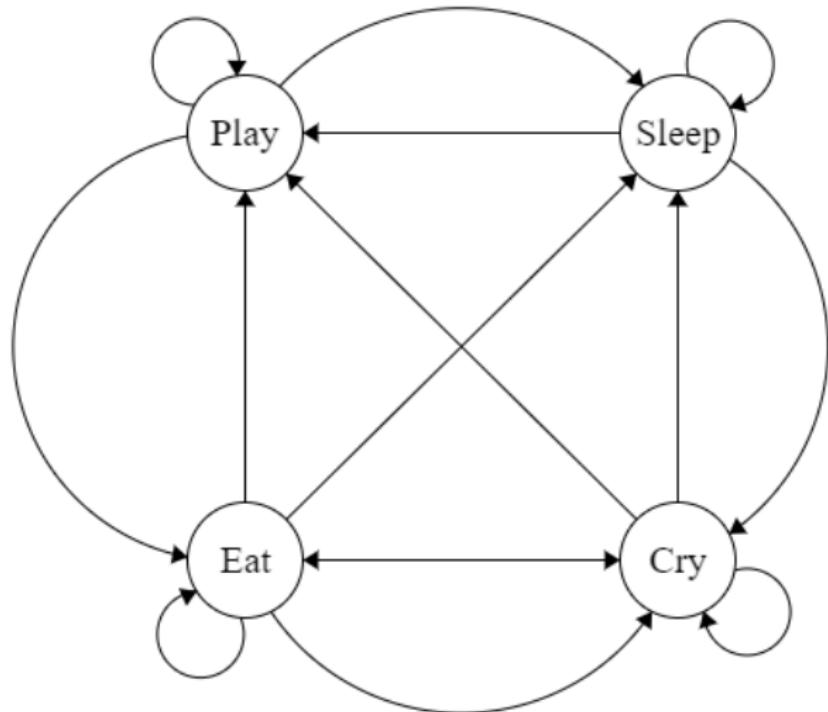
Deska Galtona



Nekrasov vs Markov



Łańcuch Markowa



Łańcuch Markowa

Łańcuch Markowa to rodzaj procesu stochastycznego, w którym przyszłe stany zależą tylko od obecnego stanu i są niezależne od przeszłych stanów. Formalnie, łańcuch Markowa to ciąg zmiennych losowych X_1, X_2, \dots , z wartościami w skończonym zbiorze stanów $S = \{s_1, s_2, \dots, s_n\}$, który spełnia warunek Markowa:

$$P(X_{n+1} = s_j \mid X_1 = s_{i_1}, X_2 = s_{i_2}, \dots, X_n = s_{i_n}) = P(X_{n+1} = s_j \mid X_n = s_{i_n})$$

dla $1 \leq i_1, i_2, \dots, i_n, j \leq n$.

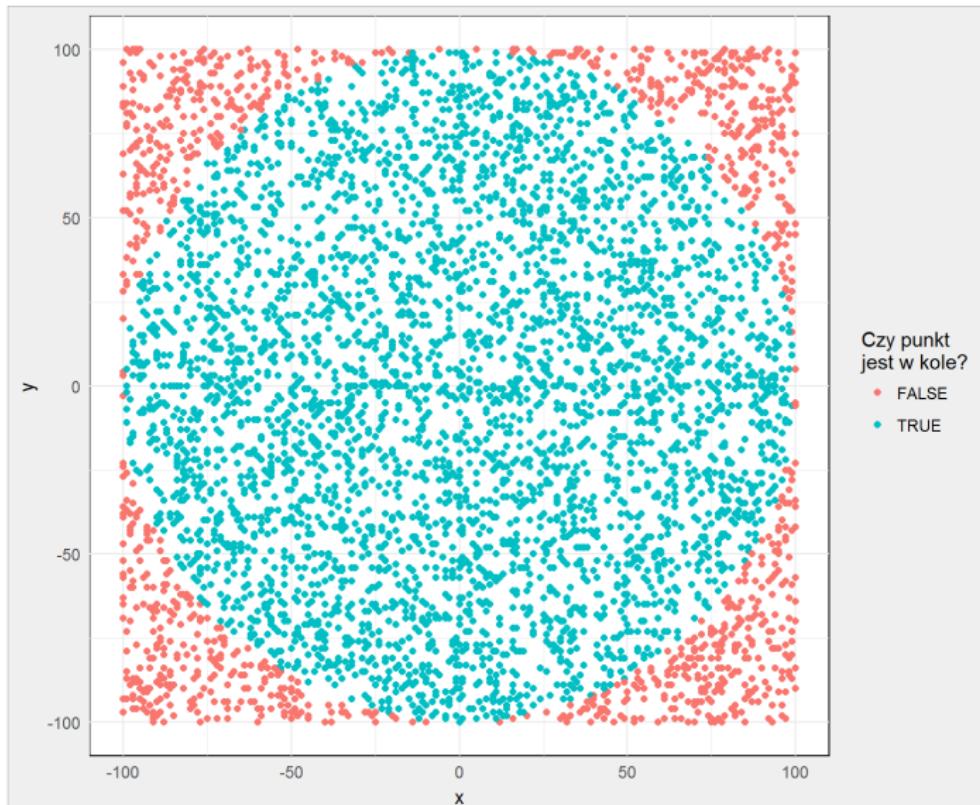
Łańcuch Markowa

Łańcuch Markowa posiada własność memoryless, co oznacza, że zależność przyszłych stanów od obecnego stanu nie zależy od ilości przeszłych stanów, które przeszły się pomiędzy nimi. Innymi słowy, przyszłe stany zależą tylko od ostatniego stanu.

Łańcuch Markowa ma wiele zastosowań, w tym:

- Modelowanie procesów losowych, takich jak ruch uliczny, prognozowanie pogody, procesy ekonomiczne, itp.
- Analiza i modelowanie danych, takich jak sekwencje DNA, sekwencje językowe, itp.
- Zastosowania w teorii gier i sztucznej inteligencji, takie jak proces decyzyjny Markowa (MDP)

Metoda Monte Carlo



Łańcuchy Markowa Monte Carlo

W przypadku, gdy S jest przestrzenią skończoną stanów dyskretnych

$$S = \{x_1, x_2, \dots, x_n\}$$

możemy prawdopodobieństwo przejścia sformułować w postaci macierzy losowej P :

$$\begin{bmatrix} T(x_1, x_1) & T(x_1, x_2) & \dots & T(x_1, x_n) \\ T(x_2, x_1) & T(x_2, x_2) & \dots & T(x_2, x_n) \\ \dots & \dots & \dots & \dots \\ T(x_n, x_1) & T(x_n, x_2) & \dots & T(x_n, x_n) \end{bmatrix}$$

warunek normalizacji

$$\sum_{j=1}^n T(x_i, x_j) = 1$$

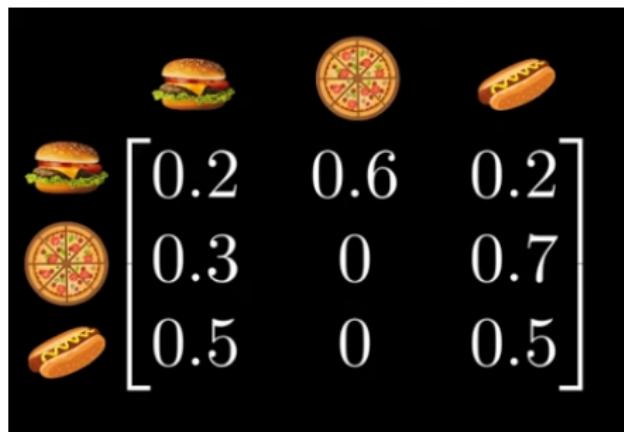
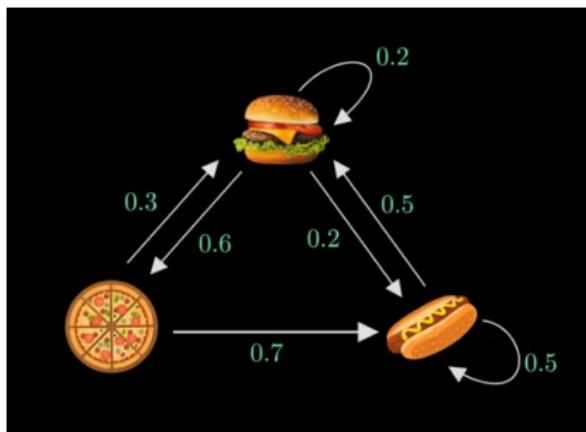
Łańcuchy Markowa Monte Carlo

Własności macierzy losowych:

- suma elementów w wierszu jest równa 1 (warunek normalizacji)
- iloczyn macierzy losowych też jest macierzą losową
- przynajmniej jedna wartość własna macierzy równa jest 1

Łańcuchy Markowa Monte Carlo

Jak to działa?



Łańcuchy Markowa Monte Carlo

Niech P będzie macierzą prawdopodobieństwa z poprzedniego slajdu

$$P = \begin{bmatrix} 0.2 & 0.6 & 0.2 \\ 0.3 & 0 & 0.7 \\ 0.5 & 0 & 0.5 \end{bmatrix}$$

To co chcemy obliczyć to rozkład, czyli prawdopodobieństwo każdego ze stanów, nazwijmy je π . Na początku wybierzmy dowolny ze stanów, od którego zaczniemy. Niech to będzie pizza. Oznaczmy zatem $\pi_0 = [0 \ 1 \ 0]$

Łańcuchy Markowa Monte Carlo

Zobaczmy co się stanie jeśli pomnożymy nasz wektor przez macierz prawdopodobieństwa

$$\pi_0 P = [\begin{array}{ccc} 0 & 1 & 0 \end{array}] \left[\begin{array}{ccc} 0.2 & 0.6 & 0.2 \\ 0.3 & 0 & 0.7 \\ 0.5 & 0 & 0.5 \end{array} \right] = [\begin{array}{ccc} 0.3 & 0 & 0.7 \end{array}]$$

Otrzymaliśmy drugi wiersz macierzy, czyli inaczej mówiąc prawdopodobieństwo pozostałych stanów jeśli pierwszym stanem była pizza. Weźmy ten wynik i wstawmy go w miejsce π_0 i wykonajmy działanie ponownie

Łańcuchy Markowa Monte Carlo

$$\pi_1 P = [\begin{array}{ccc} 0.3 & 0 & 0.7 \end{array}] \begin{bmatrix} 0.2 & 0.6 & 0.2 \\ 0.3 & 0 & 0.7 \\ 0.5 & 0 & 0.5 \end{bmatrix} = [\begin{array}{ccc} 0.41 & 0.18 & 0.41 \end{array}]$$

$$\pi_2 P = [\begin{array}{ccc} 0.41 & 0.18 & 0.41 \end{array}] \begin{bmatrix} 0.2 & 0.6 & 0.2 \\ 0.3 & 0 & 0.7 \\ 0.5 & 0 & 0.5 \end{bmatrix} = [\begin{array}{ccc} 0.34 & 0.25 & 0.41 \end{array}]$$

Łańcuchy Markowa Monte Carlo

Wykonując takie działania wielokrotnie finalnie otrzymamy $\pi P = \pi$
Jeśli to równanie przypomina wam to wektor własny to się nie mylicie :)) jednak rozczarowując miłośników algebry liniowej nie będę się zagłębiał w szczegóły tego podobieństwa. Najbardziej istotny jest fakt, że dane π jest szukanym rozkładem.

$$\pi = [0.35211 \quad 0.21127 \quad 0.43662]$$

Łańcuchy Markowa Monte Carlo

Gdzie w tym wszystkim metoda Monte Carlo?
Łańcuchy Markowa działają świetnie gdy mówimy o niezbyt wielkich macierzach. Co jeśli nasza macierz jest rozmiarów $10! \times 10!?$ Naturalnie nie będziemy takich macierzy liczyć, więc tu z pomocą przychodzi metoda Monte Carlo. Zamiast całej macierzy weźmiemy tylko próbkę.

Łańcuchy Markowa Monte Carlo

Ideą Markov Chain Monte Carlo (MCMC) jest wykorzystanie Łańcuchów Markowa do próbkowania zgodnie z niejednolitymi rozkładami. Metoda ta, choć obliczeniowo ekspansywna w niższych wymiarach, jest niezwykle wydajna w wyższych wymiarach, ponieważ jej stopień zbieżności jest niezależny od wymiaru przestrzeni.[2]

Łańcuchy Markowa Monte Carlo w zastosowaniu

Fantastycznym przykładem działania MCMC i wykorzystania w kryptografii jest ten czerpany z pracy studentów Stanfordu Marc Coram oraz Phil Beineke.[1]

W jednym z więzień udało się przechwycić zaszyfrowaną wiadomość

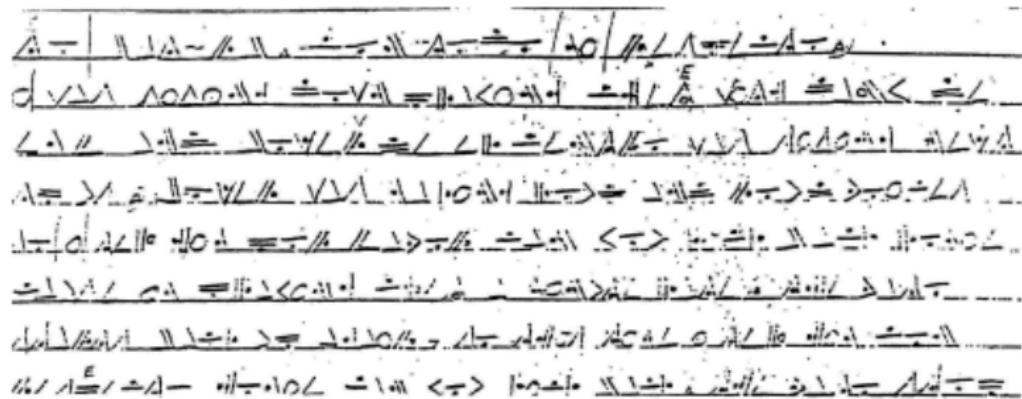


Figure: Obrazek pochodzi z [1]

Łańcuchy Markowa Monte Carlo w zastosowaniu

Studenci szybko ustalili, że szyfr polegał na zwykłym zastąpieniu każdej litery pewnym symbolem, zatem istniał jakiś nieznany schemat, nazwijmy go funkcją, która działała w następujący sposób:

$$f : \{ \text{zaszyfrowana wiadomość} \} \rightarrow \{ \text{zwykły alfabet} \}$$

Łańcuchy Markowa Monte Carlo w zastosowaniu

W celu złamania kodu studenci wykorzystali standardowy text i utworzyli macierz prawdopodobienstwa każdego z symboli oraz opracowali wzór wiarygodności funkcji

$$PI(f) = \prod_i M(f(s_i), f(s_{i+1}))$$

Następnie algorytm działał w następujący sposób:

- Wybranie losowego startu, nazwijmy je f .
- Obliczanie $PI(f)$
- Zmiana f na f_* dokonując losowej transpozycji 2 wartości przypisanych do f
- Obliczenie $PI(f_*)$ i jeśli jest większe od $PI(f)$ to zaakceptowanie f_*
- W przeciwnym wypadku "rzucenie monetą" o prawdopodobieństwie sukcesu $PI(f_*)/PI(f)$ i jeśli wypadnie orzeł to zaakceptowanie f_* , jeśli nie to zostanie przy f

Łańcuchy Markowa Monte Carlo w zastosowaniu

Algorytm miał działać do momentu aż zwrócony tekst będzie miał jakiś sens. Okazało się, że już po około 2 tysiącach kroków rozszywrowana wiadomość jest zrozumiała i w nie ulega zmianie w wyniku dalszego działania algorytmu. Algorytm działa na tyle dobrze, że można rozszyfrować wiadomość zaledwie ze skrawka tekstu a co dopiero z całej notatki jaką udało się przejąć w więzieniu.

to bat-rb. con todo mi respeto. i was sitting down playing chess with danny de emf and boxer de el centro was sitting next to us. boxer was making loud and loud voices so i tell him por favor can you kick back homie cause im playing chess a minute later the vato starts back up again so this time i tell him con respecto homie can you kick back. the vato stop for a minute and he starts up again so i tell him check this out shut the f**k up cause im tired of your voice and if you got a problem with it we can go to celda and handle it. i really felt disrespected thats why i told him. anyways after i tell him that the next thing I know that vato slashes me and leaves. dy the time i figure im hit i try to get away but the c.o. is walking in my direction and he gets me right dy a celda. so i go to the hole. when im in the hole my home boys hit doxer so now "b" is also in the hole. while im in the hole im getting schoold wrong and

Figure: Obrazek pochodzi z [1]

Łańcuchy Markowa Monte Carlo w zastosowaniu

Zastosowania symulacyjnych metod MCMC obejmują bardzo szeroki zakres dziedzin i aplikacji praktycznych.

Przykłady zastosowania MCMC

- Obliczanie całek wielowymiarowych
- Problemy typu "problem plecaka"
- Fizyka komputerowa
- Statystyka Bayesowska
- Biologii obliczeniowej
- Lingwistyce komputerowej

Dziękuję za uwagę

Bibliografia

-  Persi Diaconis.
The markov chain monte carlo revolution.
Bulletin of the American Mathematical Society, 46(2):179–205, 2009.
-  Matthew Richey.
The evolution of markov chain monte carlo methods.
The American Mathematical Monthly, 117(5):383–413, 2010.

Modyfikacje Symulowanego Wyżarzania

Wojciech Grabias

March 2023

Problem optymalizacyjny

Przedstawiony problem optymalizacyjny polegać będzie na
znalezieniu takiego $x \in \mathcal{X}$, dla którego funkcja celu f przyjmuje
wartość minimalną.

Simulated Annealing

T – Temperatura

$f : \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ – Funkcja celu, $\mathcal{X} \subset \mathbb{R}^d$

Rozkład Boltzmanna

$$\pi_T(x) = \frac{1}{Z_T} \exp\left(-\frac{f(x)}{T}\right)$$

Intuicja:

Dla małych T , z dużym prawdopodobieństwem x przyjmuje tylko takie stany, że wartość $f(x)$ jest mała.

Potrzebne definicje

Dla funkcji celu f :

$$S_* = \{x \in \mathcal{X} : f(x) = 0\}$$

$$S_\epsilon = \{x \in \mathcal{X} : f(x) \leq \epsilon\}, \epsilon > 0$$

$\mathcal{M}(\mathcal{X})$ – zbiór prawdopodobieństw na $(\mathcal{X}, \mathfrak{B}(\mathcal{X}))$

$G(x, dy)$ – Będziemy je traktowali jako rozkład normalny o średniej x

Algorithm 1: SA

Initialization with $x_0 \sim \mu_0$, $\mu_0 \in \mathcal{M}(\mathcal{X})$

for $k = 0, \dots$ **do**

 Generate a candidate $y_k \sim G(x_k, dy)$

 Compute the acceptance probability

$$p_k = \exp \left(- \left(\frac{f(y_k) - f(x_k)}{T_k} \right)_+ \right)$$

 Set $x_{k+1} = \begin{cases} y_k & \text{with probability } p_k \\ x_k & \text{with probability } 1 - p_k \end{cases}$

end

Algorytm 1: SA, [2]

Zbieżność algorytmu SA

Metropolis-Hastings (MH) kernel

$$P_k(x, dy) = p_k(y, x)G(x, dy) + (1 - r(x))\delta_x(dy), \text{ gdzie}$$

$$r(x) = \int_{\mathcal{X}} p_k(y, x)G(x, dy)$$

Wówczas zdefiniować możemy:

$$\mu_k(dx) = \mathbb{P}(x_k \in dx) = \mu_{k-1}P_k, \quad k > 1$$

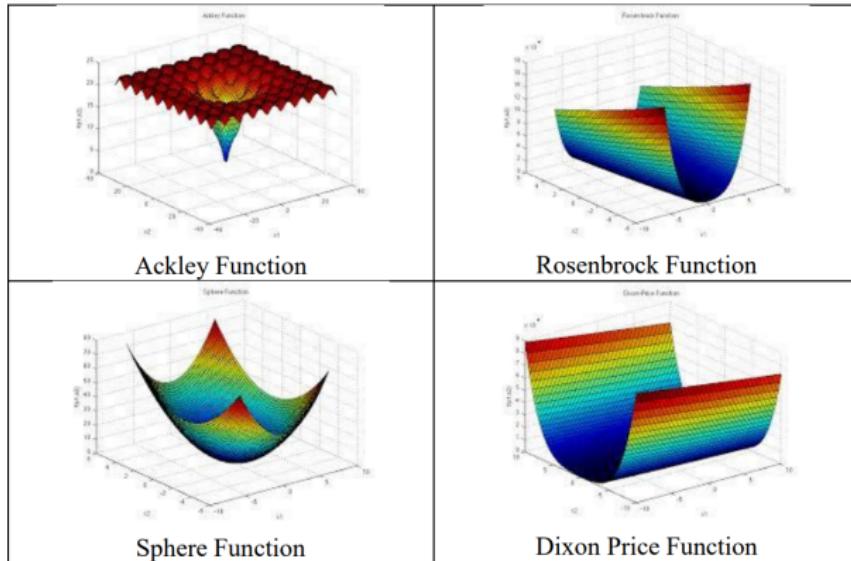
Zbieżność algorytmu SA

Twierdzenie

Jeżeli T maleje logarytmicznie z dokładnością do pewnej stałej, to przy odpowiednio dużym T_0 :

$$\lim_{k \rightarrow \infty} \mathbb{P}(x_k \in S_\epsilon) = 1$$

Efektywność modyfikacji - przykład I-MSAA



Obrazek 1: Optymalizowane funkcje, [3]

Efektywność modyfikacji - przykład I-MSAA

Function	D	Optimal	SA			I-MSAA		
			Best	Worse	SD	Best	Worse	SD
Dixon Price	10	0.0	100.3	891.4	1.9×10^2	0.0	0.0	8.9×10^{-8}
Ackley	50	0.0	17.9	19.0	0.2	0.0	0.0	2.3×10^{-5}
Neumaier	10	-210.0	764.8	3.6×10^3	6.4×10^2	-210.0	-210.0	3.2×10^{-9}
Rosenbrock	50	0.0	5.7×10^5	9.4×10^5	8.3×10^4	0.0	4.0	1.2

Tabela 1: Porównanie SA i I-MSAA,[3]

FSA- Fast Simulated Annealing

FSA - zamysł

FSA bierze swoją nazwę od przyspieszonego schematu chłodzenia - chcemy szybciej osiągać niskie temperatury.

Aby kontrolować zachowanie algorytmu stosować będziemy funkcje malejące wolniej od dotychczasowo przyjętej e^{-x}

Generalizacja SA o dowolność funkcji q określającej prawdopodobieństwo przyjęcia nowego punktu y_k (ang. acceptance function):

$$p_k = q(\rho_k), \text{ gdzie}$$

$$\rho_k = \left(\frac{f(y_k) - f(x_k)}{T_k} \right)_+$$

Dozwolone są jednak funkcje malejące wolniej od e^{-x} .

Zwykle przyjmuje się, że $\rho \mapsto q(\rho) = \frac{1}{1+\rho}$

Tak zmodyfikowane jądra przejścia oznaczać będziemy $P_k^{(F)}$

Twierdzenie*

Jeżeli dla pewnego $\gamma \in (0, 1]$:

$$T_k = \frac{1}{(k+1)^\gamma \log((k+1)^\gamma)}$$

to istnieje $C_\epsilon > 0$ spełniające:

$$\mathbb{P}(x_k \in S_\epsilon) \geq 1 - \frac{C_\epsilon}{(k+1)^\gamma}, \forall k \in \mathbb{N}$$

SMC-SA - zamysł

Zamiast szukać minimum pojedynczym punktem, szukajmy go kilkoma punktami jednocześnie, moderując ich zachowanie w każdej iteracji.

Aktualny zbiór punktów

Do dyspozycji mamy n punktów x_1, \dots, x_n

Przypisanie wag każdemu z punktów

Każdy z punktów otrzymuje swoją wagę w oparciu o stosunek rozkładów Boltzmann'a

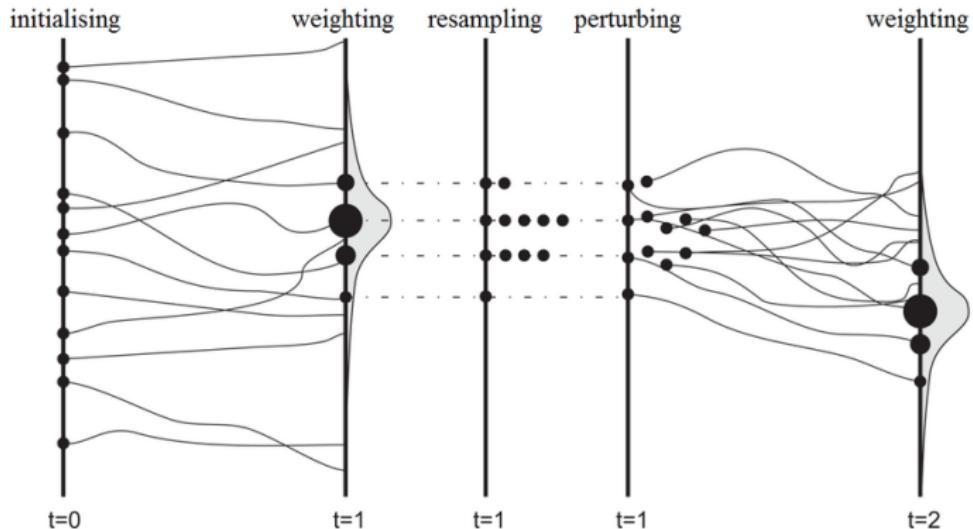
Resampling

Na podstawie wag z odpowiednim prawdopodobieństwem wybieramy odpowiednie punkty (mogą się powtarzać), nowy zbiór punktów może być większy od poprzedniego

Zmiana położenia punktów

Każdy z nowych punktów przepuszczamy przez kernel MH - P_k

SMC-SA: Schemat działania



Obrazek 2: Wizualizacja resamplingu na podstawie rozkładu normalnego,
[1]

Algorithm 2: SMC-SA

Initialize the algorithm $x_k^{(n)} \sim \mu_0$ for $1 \leq n \leq N_0$;
for $k = 1, \dots$ **do**

- Compute the self-normalized weights
 $w_k^{(n)} \propto \frac{\pi_k}{\pi_{k-1}}(x_{k-1}^{(n)})$
- Resample $\{\tilde{x}_k^{(n)}\}_{n=1}^{N_k}$ from $\{x_{k-1}^{(n)}, w_k^{(n)}\}_{n=1}^{N_{k-1}}$
- Generate $\{x_k^{(n)}\}_{n=1}^{N_k}$ propagating the points
 $\{\tilde{x}_k^{(n)}\}_{n=1}^{N_k}$ with the MH kernel $P_k(x, dy)$

end

Algorytm 2: SMC-SA, [2]

Podsumowanie 3 modyfikacji SA

SA

Podstawowa wersja: p_k ścisłe związane z π_k , wolne tempo chłodzenia

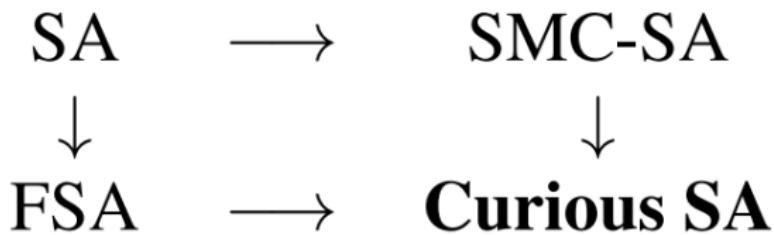
FSA

Szybsze tempo chłodzenia, wolniej malejąca funkcja wyznaczająca prawdopodobieństwa zmiany stanu

SMC-SA

Rozszerzenie SA o wielopunktowość, kontrola punktów poprzez narzucenie rozkładu

Curious Simulated Annealing



Curious Simulated Annealing

Algorithm 3: CSA

Initialize the algorithm $x_k^{(n)} \sim \mu_0$ for $1 \leq n \leq N$;

for $k = 1, \dots$ **do**

 Compute the self-normalized weights

$$w_k^{(n)} \propto \frac{\pi_k}{\pi_{k-1}}(x_{k-1}^{(n)})$$

 Resample $\{\tilde{x}_k^{(n)}\}_{n=1}^N$ from $\{x_{k-1}^{(n)}, w_k^{(n)}\}_{n=1}^N$

 Generate $\{x_k^{(n)}\}_{n=1}^N$ propagating the points

$\{\tilde{x}_k^{(n)}\}_{n=1}^N$ with the MH kernel $P_k^{(F)}(x, dy)$

end

Algorytm 3: CSA, [2]

Problem 1

Minimalizacja funkcji Rosenbrock'a w \mathbb{R}^{10} :

$$f(x) = \sum_{i=1}^9 5(x_{i+1} - x_i^2)^2 + (1 - x_i)^2, \quad \forall x \in \mathbb{R}^{10}$$

		SA	FSA	SMC-SA	CSA
(P ₁)	$\langle f_{50}^* \rangle$	6.31	6.49	6.41	4.05
	σ_{50}^*	0.829	0.732	1.15	1.17
	$\langle f_{500}^* \rangle$	3.64	3.72	5.06	2.19
	σ_{500}^*	0.761	0.778	1.26	0.447

Tabela 2: Porównanie testów numerycznych, [2]

Problem 2

Minimalizacja funkcji Rastrigin'a z równomiernie rozłożonymi minimami:

$$f(x) = \sum_{i=1}^{10} x_i^2 - \cos(2\pi x_i), \quad \forall x \in \mathbb{R}^{10}$$

		SA	FSA	SMC-SA	CSA
(P ₂)	$\langle f_{50}^* \rangle$	3.29	3.36	3.26	3.23
	σ_{50}^*	0.425	0.453	0.521	0.484
	$\langle f_{500}^* \rangle$	2.52	2.64	2.62	2.47
	σ_{500}^*	0.320	0.304	0.413	0.502

Tabela 3: Porównanie testów numerycznych, [2]

Dziękuję za uwagę :)

Bibliografia

-  Danilo Alvares, Carmen Armero, Anabel Forte, and Nicolas Chopin.
Sequential monte carlo methods in bayesian joint models for longitudinal and time-to-event data.
Statistical Modelling, 21(1-2):161–181, 2021.
-  Emilie Chouzenoux, Víctor Elvira, and Thomas Guilmeau.
Simulated annealing: a review and a new scheme.
PGMO DAYS 2021, page 33, 2021.
-  Jesús Suarez, Carlos Millan, and Euriel Millan.
Improved modified simulated annealing algorithm for global optimization.
Contemporary Engineering Sciences, 11(96):4789–4795, 2018.

SMC-SA

Kamil Kisiel

Marzec 2023

Plan prezentacji

- SA
- SMC-SA
- Zakres błędu
- Eksperymenty numeryczne

SA - przypomnienie

Co to było SA?

Algorytm szukający rozwiązania danego problemu, którego działanie zaprezentuje na następującym przykładzie:

Problem optymalizacyjny

Znalezienie maksimum danej funkcji:

Założenia

\mathcal{X} - niepusty zbiór na \mathbb{R}^n

$H : \mathcal{X} \rightarrow \mathbb{R}$ (jest ograniczona i ciągła)

SA - Algorytm

- ① Losujemy startowy punkt x_k
- ② Generujemy y_k z rozkładu o gęstości $g_k(y|x_k)$
- ③ Obliczamy prawdopodobieństwo $\rho_k = \min\left\{\frac{H(y_k) - H(x_k)}{T_k}, 1\right\}$
- ④ Generujemy losowe $u \in [0, 1]$
- ⑤ Ustalamy

$$x_{k+1} = \begin{cases} y_k, & u \geq \rho_k \\ x_k, & u < \rho_k \end{cases} \quad (1)$$

- ⑥ Sprawdzamy kryterium stopu, ewentualnie zwiększamy k i zmniejszamy temperaturę

Co to SMC-SA?

W skrócie jest to połączenie SA oraz metody Monte Carlo, dzięki czemu jesteśmy w stanie pracować na wielu punktach na raz.

Start

- ① Dostarczamy ciąg wielkości próbek $\{N_k\}$ oraz ciąg temperatur $\{T_k\}$
- ② Inicjalizacja: generujemy $x_0^i \stackrel{\text{iid}}{\sim} \text{Unif}(X)$, $i = 1, 2, \dots, N_0$
- ③ Ustawiamy $k = 1$

SMC-SA - Algorytm

- ① Importance updating: generujemy w_k^i z rozkładu:

$$\begin{cases} \exp\left\{\frac{H(x_0^i)}{T_1}\right\}, & k = 1 \\ \exp\left\{H(x_{k-1}^i)\left(\frac{1}{T_k} - \frac{1}{T_{k-1}}\right)\right\}, & k > 1 \end{cases} \quad (2)$$

- ② Resampling: generujemy $\{\tilde{x}_k^i\}_{i=1}^{N_k}$ z $\{x_{k-1}^i, w_k^i\}_{i=1}^{N_{k-1}}$
- ③ SA Move: generujemy x_k^i z \tilde{x}_k^i dla $i = 1, \dots, N_k$, korzystając z SA
- ④ Sprawdzamy kryterium stopu oraz ewentualnie zwiększamy k

SMC-SA - zakres błędu

Przy naszych założeniach, rozkład Boltzmana słabo się zgłasza do równomiernego rozkładu na zbiorze optymalnych rozwiązań.

Propozycja 1

Dla każdego $\xi > 0$:

$$\lim_{T_k \rightarrow 0} \pi_k(\mathcal{X}_\xi) = 1,$$

gdzie $\mathcal{X}_\xi = \{x \rightarrow \mathcal{X} : H(x) > H^* - \xi\}$

H^* - optymalna wartość funkcji H

Oznaczenia pomocnicze

\mathcal{F} - σ -ciało na (\mathcal{X})

$\mathcal{B}(\mathcal{X})$ - zbiór mieralnych i ograniczonych funkcji $\phi : \mathcal{X} \rightarrow \mathbb{R}$

$\mathcal{B}_+(\mathcal{X})$ - zbiór mieralnych i ograniczonych funkcji $\phi : \mathcal{X} \rightarrow \mathbb{R}_+$

$$\langle v, \phi \rangle = \int \phi(x) v(dx), \quad \forall \phi(x) \in \mathcal{B}(\mathcal{X})$$

Definicje pomocnicze pt.1

Supremum normy

$$\|\phi\| = \sup_{x \in \mathcal{X}} |\phi(x)|$$

Całkowity dystans zmienności

v_1, v_2 - miary probabilistyczne na $(\mathcal{X}, \mathcal{F})$

$$\|v_1 - v_2\|_{TV} = \sup_{A \in \mathcal{F}} \|v_1(A) - v_2(A)\|$$

Definicje pomocnicze pt.2

Rozkłady prawdopodobieństwa w k-tej iteracji SMC-SA

$$\pi_k^d = \frac{\exp(H(x)/T_k)}{\int \exp(H(x)/T_k) dx}$$

$$\tilde{\mu}_k = \sum_{i=1}^{N_{k-1}} \omega_k^i \delta_{x_{k-1}^i}$$

$$\tilde{\mu}_k^{N_k} = \frac{1}{N_k} \sum_{i=1}^{N_k} \delta_{\tilde{x}_k^i}$$

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \delta_{\tilde{x}_k^i}$$

Definicje pomocnicze pt.3

$$\Psi_k = \frac{\pi_k^d}{\pi_{k-1}^d}$$

Zwiazki miedzy rozkladami

$$\mu_{k-1} \rightarrow \tilde{\mu}_k = \frac{\mu_{k-1} \Psi_k}{\langle \mu_{k-1}, \Psi_k \rangle} \rightarrow \tilde{\mu}_k^{N_k} \rightarrow \mu_k = \tilde{\mu}_k^{N_k} P_k$$

Założenie 1

Gęstość zaproponowana w SA Move musi spełniać następujący warunek:

$$g_k(y|x) \geq \epsilon_k > 0, \quad \forall x, y \in \mathcal{X}$$

Jednostajna ergodyczność

Twierdzenie 1

Rozważmy Łańcuch Markowa o kernelu przejściowym $P(x, dy)$ dla $x, y \in \mathcal{X}$ i stacjonarny rozkład prawdopodobieństwa π . Wtedy przestrzeń \mathcal{X} nazywamy mała jeśli istnieja:

- $n_0 \in \mathbb{Z}_+$
- Stała $\epsilon \in (0, 1)$
- Miara probabilistyczna v na \mathcal{X}

takie, że spełniony jest warunek minoryzacji:

$$P^{n_0}(x, A) \leq \epsilon v(A), \quad \forall x \in \mathcal{X}, \forall A \in \mathcal{F}$$

Wtedy Łańcuch jest ergodyczny oraz:

$$\|P^n(x) - \pi\|_{TV} \leq (1 - \epsilon)^{\lfloor n/n_0 \rfloor}, \quad \forall x \in \mathcal{X}$$

Wnioski z Twierdzenia 1

Wniosek 1.1

Przy założeniu 1 Łańcuch Markowa zgodny z krokiem SA Move po każdej iteracji jest jednostajnie ergodyczny oraz:

$$\exists \epsilon_k \in (0, 1) \quad ||P_k^n(x) - \pi_k||_{TV} \leq (1 - \epsilon_k)^n,$$

$$\epsilon_k = \varepsilon_k \exp\left\{\frac{H_l - H_u}{T_k}\right\} \lambda(\mathcal{X})$$

Wnioski z Twierdzenia 1 c.d.

Wniosek 1.2

Rozważmy łańcuch Markowa z początkowym rozkładem μ , kernelem przejść P oraz stacjonarnym rozkładem prawdopodobieństwa π . Załóżmy, że

$\forall \phi \in \mathcal{B}(\mathcal{X}) \quad |\langle \mu - \pi, \phi \rangle| \leq c\|\phi\|$, gdzie c jest dodatnia stała.

Wtedy jeśli łańcuch jest jednostajnie ergodyczny, to:

$$|\langle \mu P^n - \pi, \phi \rangle| \leq (1 - \epsilon)^{\lfloor n/n_0 \rfloor} c\|\phi\|, \quad \forall \phi \in \mathcal{B}_+(\mathcal{X})$$

Lematy

Lemat 1

Weźmy zmienne losowe x^1, \dots, x^N , które są i.i.d. i mają (warunkowy) rozkład v . Oznaczając $v^N = \frac{1}{N} \sum_{i=1}^N \delta_{x^i}$ mamy:

$$E[|\langle v - v^N, \phi \rangle| | \mathcal{F}] \leq \frac{\|\phi\|}{\sqrt{N}}, \quad \forall \phi \in \mathcal{B}(\mathcal{X})$$

Lemat 2

Założymy, że $\forall \phi \in \mathcal{B}(\mathcal{X}) |\langle \mu - v, \phi \rangle| \leq c \|\phi\|$ gdzie c to dodatnia stała oraz $\mu' = \frac{\mu \Psi}{\langle \mu, \Psi \rangle}$. Wtedy:

$$|\langle \mu' - v', \phi \rangle| \leq c \|\Psi\| \|\phi\|, \quad \forall \phi \in \mathcal{B}_+(\mathcal{X})$$

Twierdzenie 2

Bez straty ogólności zakładamy, że $\forall x \in \mathcal{X} H(x) > 0$. Ustalmy, że wstępny rozkład to v , a jego gęstość to v^d , to przy założeniu 1:

$$E[|\langle \mu_k - \pi_k, \phi \rangle| | \mathcal{F}] \leq c_k \|\phi\|, \quad \forall \phi \in \mathcal{B}_+(\mathcal{X})$$

, gdzie:

$$c_k = \begin{cases} \frac{\|\pi_0^d/v^d\|^2}{N_0}, & k = 0 \\ (1 - \epsilon_k)\left(\frac{1}{\sqrt{N_k}}\right) + \exp(H^* \Delta_k) c k_1, & k > 0 \end{cases} \quad (3)$$

Następstwo twierdzenia 2

Jeśli $T_k = T_0/\log(k+1)$, $\varepsilon_k \lambda(\mathcal{X}) = \varepsilon < 1$, gdzie,
 $\varepsilon > (1/2)^{1 - \frac{H_u - H_l}{T_0}}$ oraz $\frac{H_u - H_l}{T_0} < 1$ i $\{N_k\}$ wzrasta wystarczająca
szybko wraz z wzrostem k , to:

$$k \rightarrow \infty \implies \{c_k\} \rightarrow 0$$

Ekperymenty numeryczne

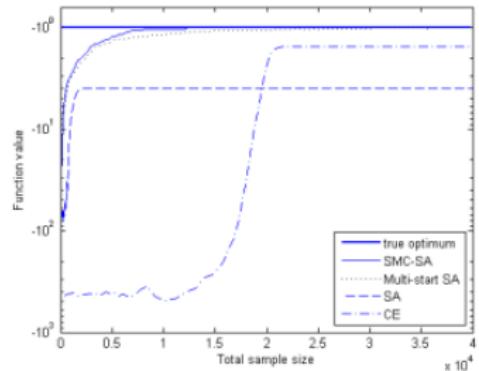
Problemy optymalizacyjne, których użyjemy do porównania

- 5. Funkcja Dejong'a (H_a)
- 20-wymiarowa funkcja Powel'a (H_b)

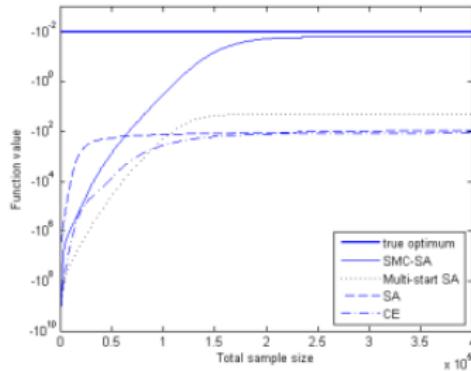
	H^*	SMC-SA		multi-start SA		standard SA	
		$\bar{H}^*(std_err)$	M_ε	$\bar{H}^*(std_err)$	M_ε	$\bar{H}^*(std_err)$	M_ε
H_a	-0.998	-0.998(1.34E-7)	100	-1.0024(0.0014)	19	-3.999(0.2117)	4
H_b	-0.01	-0.0164(4.95E-4)	81	-20.46(4.26)	0	-89.63(1.277)	0

Wykresy

(a) 2-D DeJong's 5th function



(b) 20-D Powel function



Bibliografia

-  Enlu Zhou i Xi Chen (2011) *Sequential Monte Carlo Simulated Annealing*, Springer Sciensce+Business Media
-  H. E. Romeijn i R. L. Smith *Simulated annealing for constrained global optimization*, Journal of Global Optimization

Próbnik Gibbsa

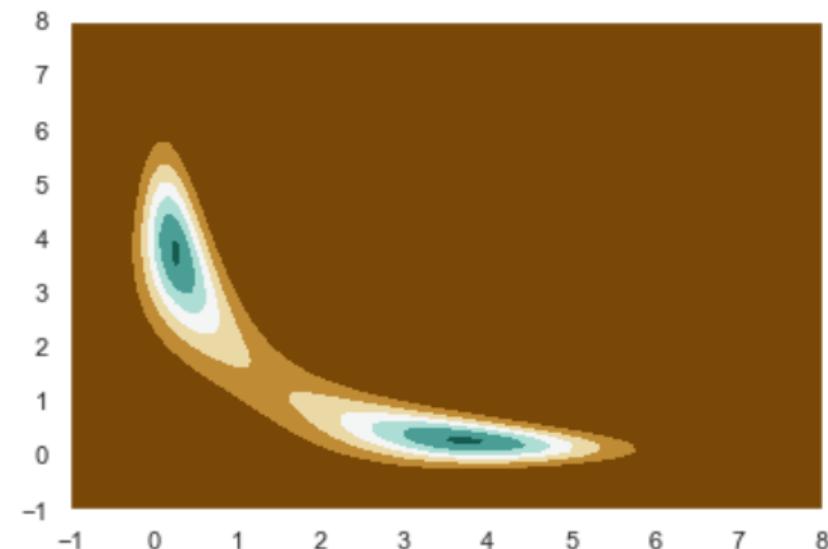
Natalia Safiejko

Plan prezentacji

- 1 Próbnik Gibbsa- co to i kiedy się używa?
- 2 Przykłady
- 3 Gibbs Sampling with People
- 4 Problemy

Co to?

Algorytm próbnika Gibbsa jest techniką numeryczną stosowaną do symulowania złożonych rozkładów prawdopodobieństwa. Algorytm ten polega na wykorzystaniu warunkowego rozkładu prawdopodobieństwa, aby wygenerować próbę z pełnego rozkładu. Działanie algorytmu opiera się na wykonaniu sekwencji kroków, w których każdy krok wykorzystuje jedną zmienną, aby wygenerować nową wartość próbki.



Zastosowania

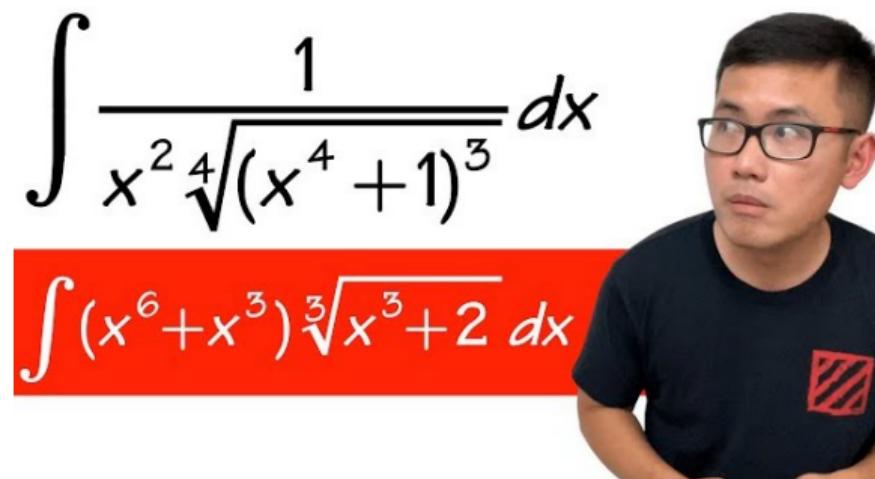
Algorytm próbnika Gibbsa jest wykorzystywany w wielu dziedzinach, takich jak analiza danych, sztuczna inteligencja, bioinformatyka, chemia i fizyka. Jego zastosowania obejmują symulacje molekularne, estymację parametrów modeli statystycznych, analizę sieci neuronowych, detekcję zmian w sygnałach, klasyfikację i rozpoznawanie obrazów oraz generowanie danych losowych.

Kiedy używamy?

Próbnik Gibbsa jest używany w sytuacjach, gdy nie jest możliwe uzyskanie analitycznej formuły rozkładu prawdopodobieństwa, a także wtedy, gdy rozkład ten jest skomplikowany i nie ma wystarczająco dużo zasobów obliczeniowych, aby go dokładnie zasymulować. W takich sytuacjach próbnik Gibbsa umożliwia generowanie próbek z rozkładu prawdopodobieństwa poprzez symulację warunkową, co pozwala na obliczenie różnych parametrów i statystyk.

$$\int \frac{1}{x^2 \sqrt[4]{(x^4 + 1)^3}} dx$$

$$\int (x^6 + x^3) \sqrt[3]{x^3 + 2} dx$$



NA PRZYKŁAD

Mamy dany wielowymiarowy rozkład prawdopodobieństwa. Zależy nam na otrzymaniu np. średniej czy wartości oczekiwanej z gęstości brzegowej.

$$f(x) = \int \cdots \int f(x, y_1, \dots, y_p) dy_1 \dots dy_p$$

Przykład na dwuwymiarowej zmiennej

Generowanie próbek

X, Y - zmienne losowe

Generowanie próbek z rozkładów warunkowych $f(x | y)$ i $f(y | x)$

"**Gibbs sequence**": $Y'_0, X'_0, Y'_1, X'_1, Y'_2, X'_2, \dots, Y'_k, X'_k$

Początkowa wartość $Y'_0 = y'_0$, następne elementy generowane według schematu: [2]

$$X'_j \sim f(x | Y'_j = y'_j)$$

$$Y'_{j+1} \sim f(y | X'_j = x'_j)$$

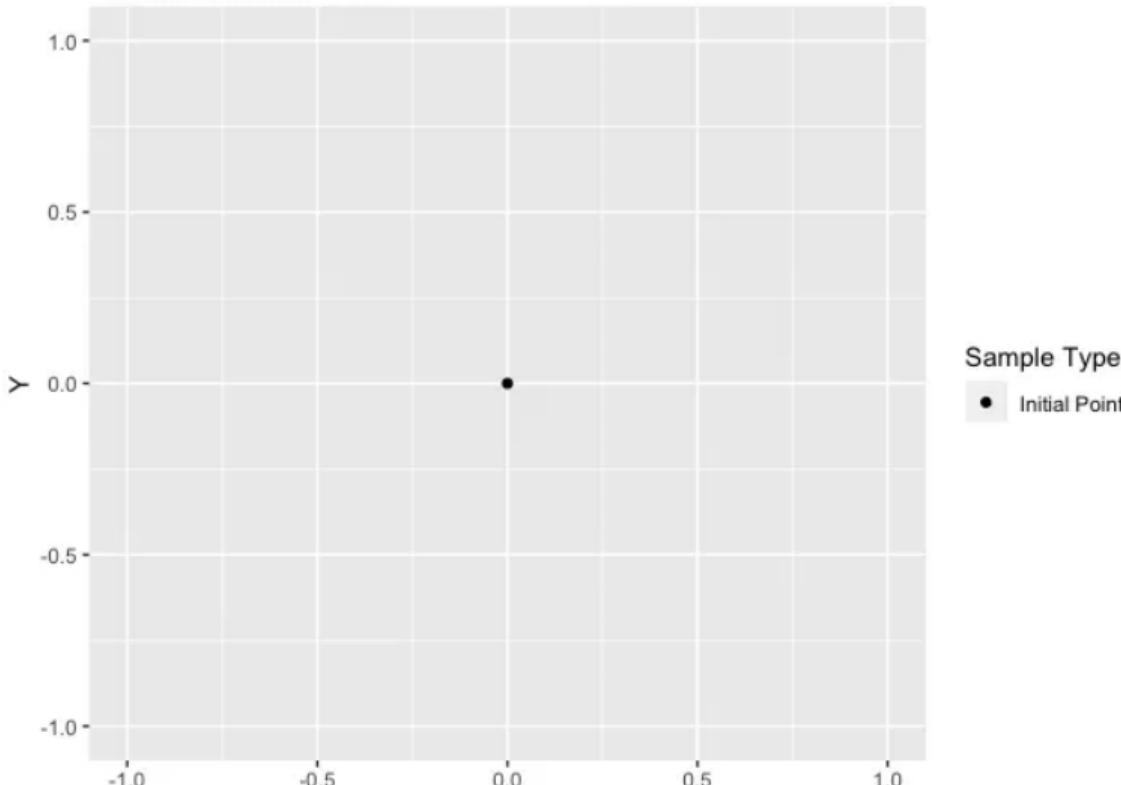
Konkretniej

$$X \mid (Y = y) \sim N(\rho y, 1 - \rho^2)$$

$$Y \mid (X = x) \sim N(\rho x, 1 - \rho^2)$$

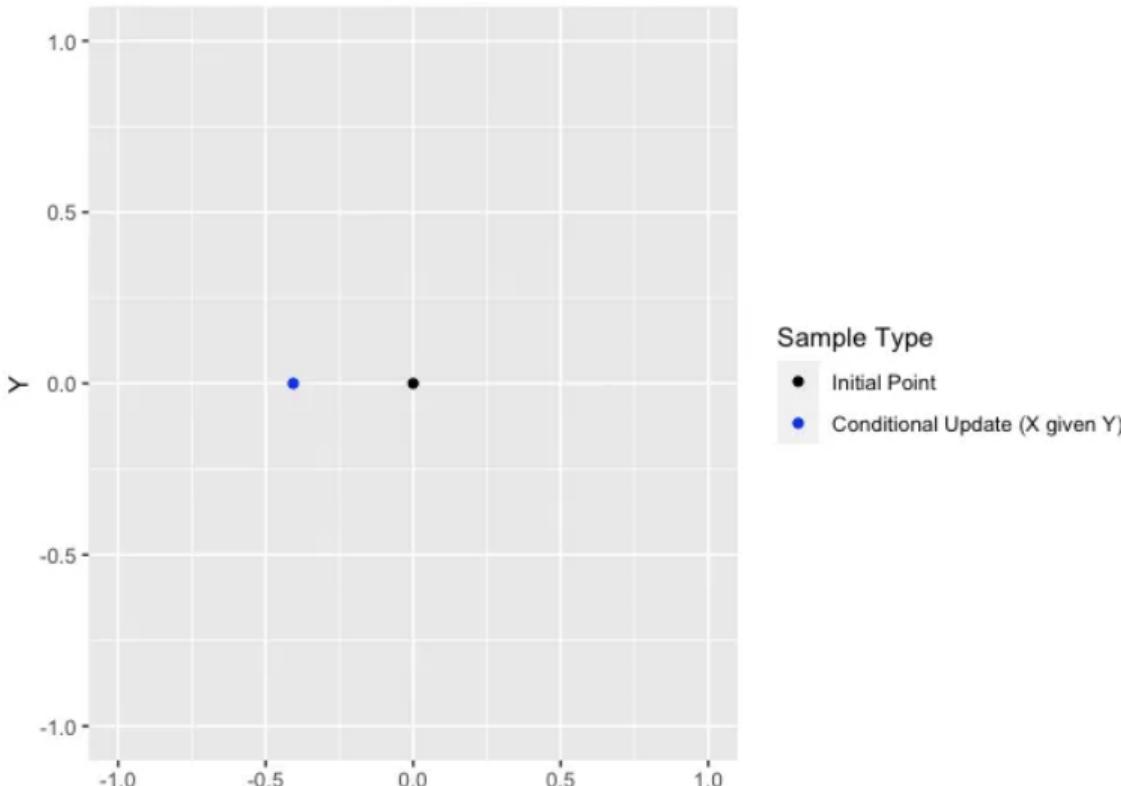
[1]

Gibbs Illustration, Step 1: Initialization



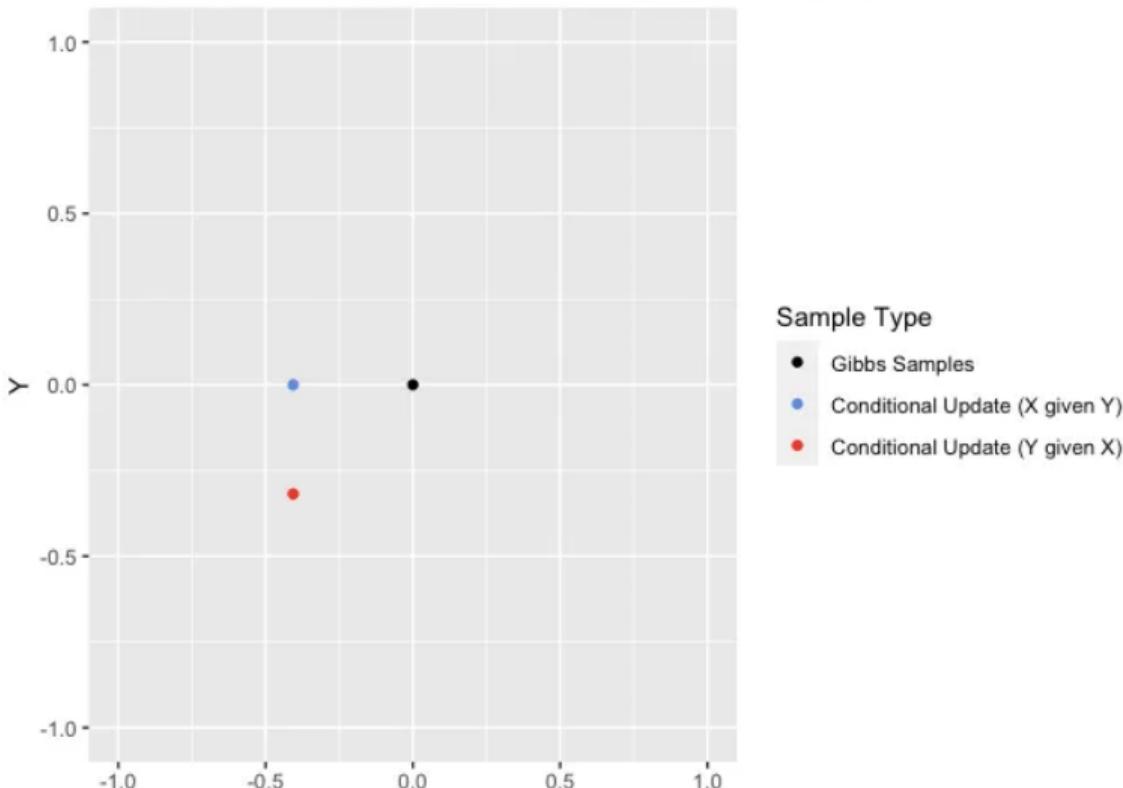
$$X_1 \mid (Y_0 = 0) \sim N(0 \cdot \rho, 1 - \rho^2)$$

Gibbs Illustration, Step 2: Conditional Update for X given Y

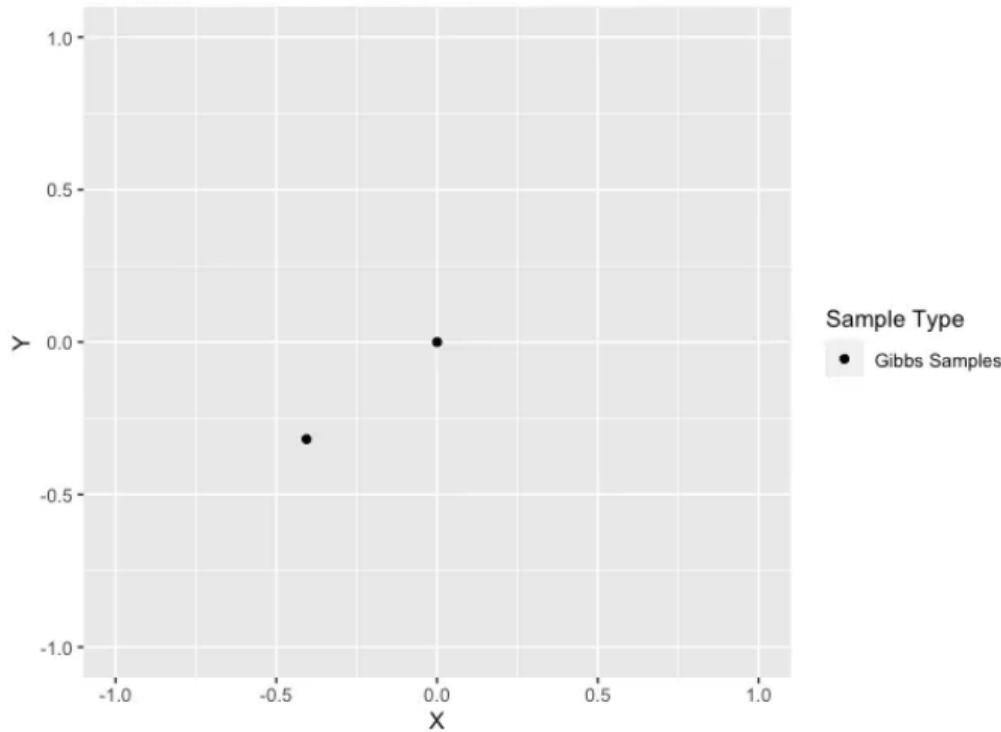


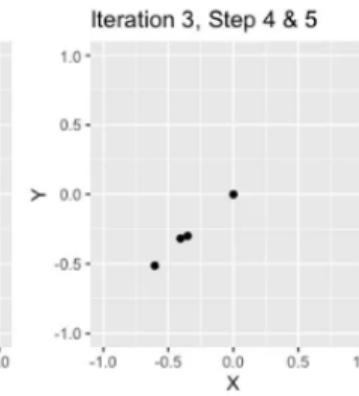
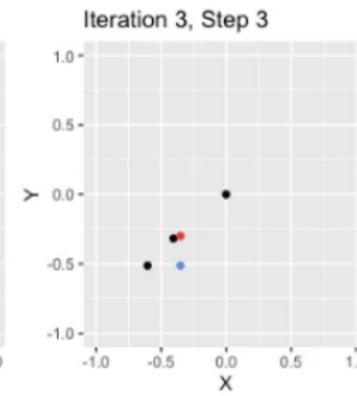
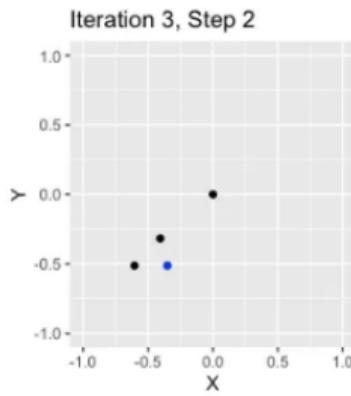
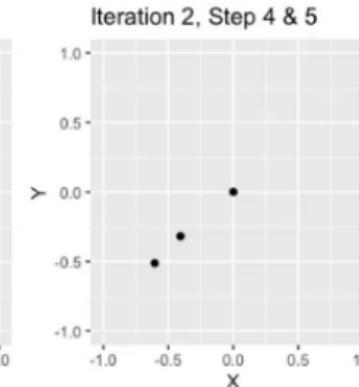
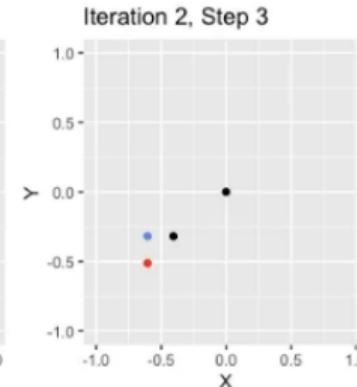
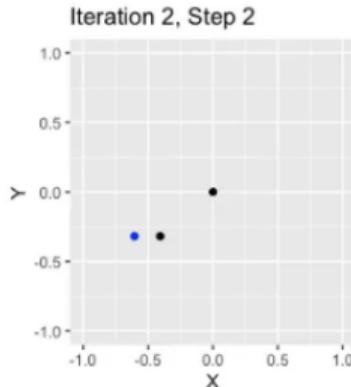
$$Y_1 \mid (X_1 = -0.4) \sim N(-0.4 \cdot \rho, 1 - \rho^2)$$

Gibbs Illustration, Step 3: Conditional Update for Y given X

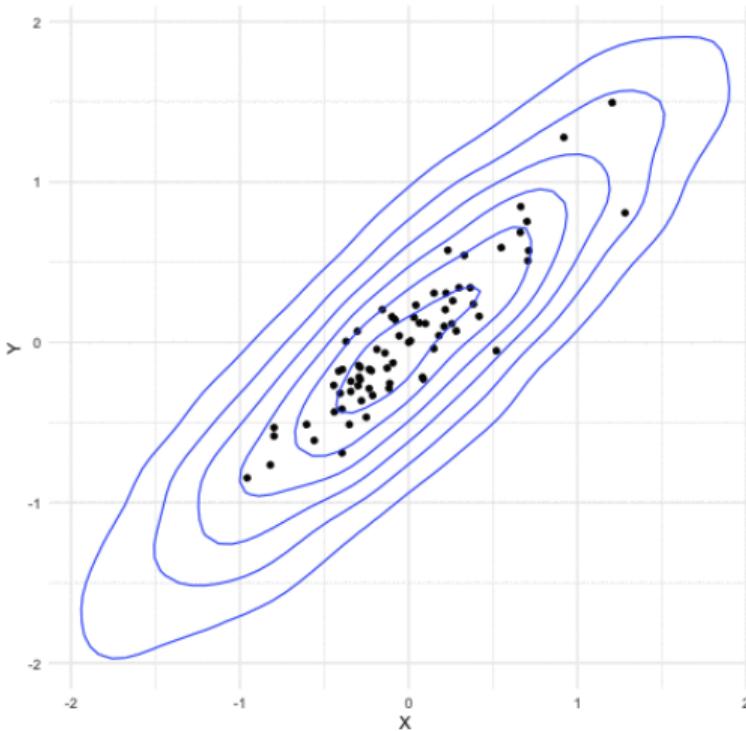


Gibbs Illustration, Steps 4 and 5: Increment and Return





Gibbs Sampling Illustration: Step 71



Wizualizacja algorytmu

<https://chi-feng.github.io/mcmc-demo/app.html?algorithm=GibbsSampling&target=banana>

Rzut monetą :)

Przeprowadzamy k eksperymentów. W każdym:

- Rzucamy monetą n razy (n jest niewiadomą)
- Prawdopodobieństwo wyrzucenia reszki oznaczmy jako θ (niewiadoma)
- Jako X_i oznaczmy liczbę wyrzuconych reszek

Otrzymamy dzięki temu wektor (X_1, X_2, \dots, X_k)

$$\bar{X} = (X_1, X_2, \dots, X_k)$$

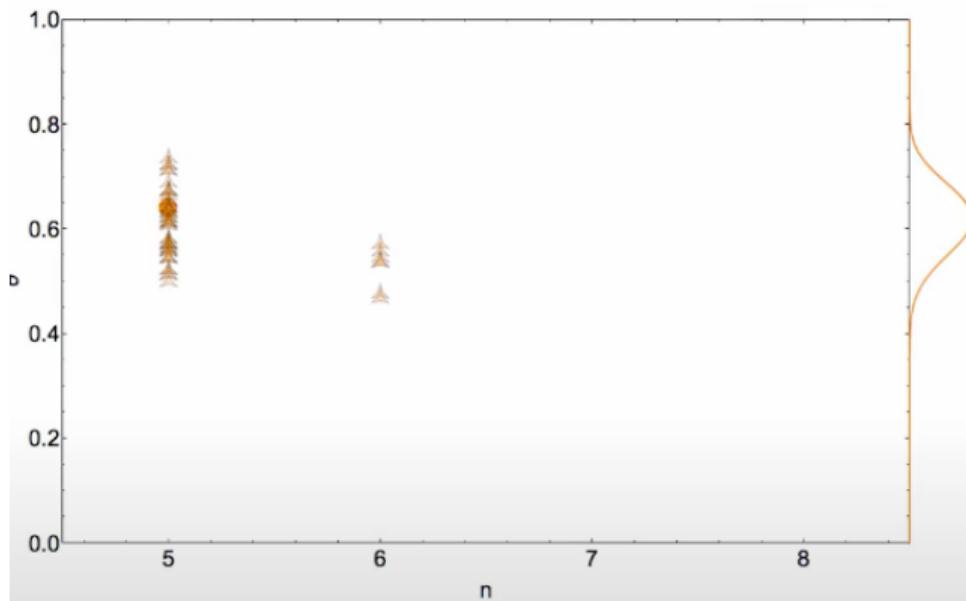
$$\theta \sim U(0, 1) \quad n \sim U(5, 8)$$

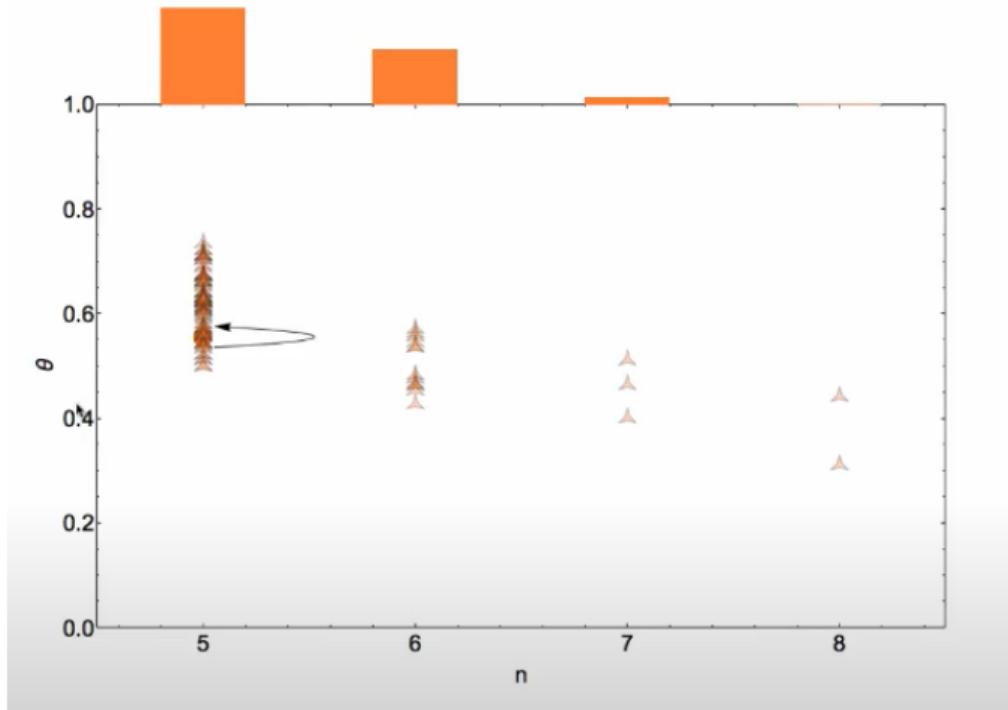
$$P(n, \theta | \bar{X}) \propto P(\bar{X} | n, \theta) \times P(n, \theta) =$$

$$= \prod_{i=1}^k \binom{n}{X_i} \theta^{X_i} (1 - \theta)^{n - X_i} = \dots = \theta^{k\bar{X}} (1 - \theta)^{nk - k\bar{X}} \prod_{i=1}^k \binom{n}{X_i}$$

$$(\theta | n, \bar{X}) \sim \theta^{k\bar{X}} (1 - \theta)^{nk - k\bar{X}} \equiv \text{beta}(k\bar{X} + 1, k(n - \bar{X}) + 1)$$

$$(n | \theta, \bar{X}) \sim (1 - \theta)^{nk} \prod_{i=1}^k \binom{n}{X_i}$$





Gibbs Sampling with People

- Najistotniejszym problemem w kogniwistyce i machine learningu jest zrozumienie jak ludzie uzyskują reprezentację semantyczną z obiektów percepcyjnych. [3]
- Markov Chain Monte Carlo with People (MCMCP) jest ważną metodą analizy tych zależności, asymptotycznie jest zadowalający, ale opiera się na binarnych wyborach.
- Okazuje się, że lepszy jest Gibbs Sampling with People (GSP) oparty na ciągłych suwaczkach.
- Eksperymenty sprawdzające GSP w czterech dziedzinach: kolorów, akordów muzycznych, emocji wynikających z głosu i mimiki twarzy



Algorytm

Niech $p(z_1, z_2, \dots, z_n)$ będzie docelowym, n-wymiarowym rozkładem, z którego chcemy próbować.

Algorytm:

- ① Wybierz początkowy wektor stanu $z^{(1)} = (z_{(1)1}, \dots, z_{(1)n})$
- ② Aktualizuj współrzędne za pomocą próbkowania z

$$p(z_{(i+1)k} \mid z_{(i+1)1}, \dots, z_{(i+1)k-1}, z_{(i)k+1}, \dots, z_{(i)n})$$

W naszym przypadku to uczestnik zapewnia próbę z warunkowego rozkładu. Osiągane jest to za pomocą suwaka związanego z aktualnym wymiarem bodźca z_k i przesunięcie go tak, aby jak najbardziej pasował do danego zagadnienia, np. jak bardzo przyjemny jest dany dźwięk czy podobieństwo wyglądu owocu do truskawki.

Kolory

Sparametryzowana przestrzeń kolorów za pomocą schematu HSL (Hue, Saturation, Lightness) o wartościach z zakresu odpowiednio [0,360], [0,100], [0,100]

A

MCMCP Markov Chain Monte Carlo with People

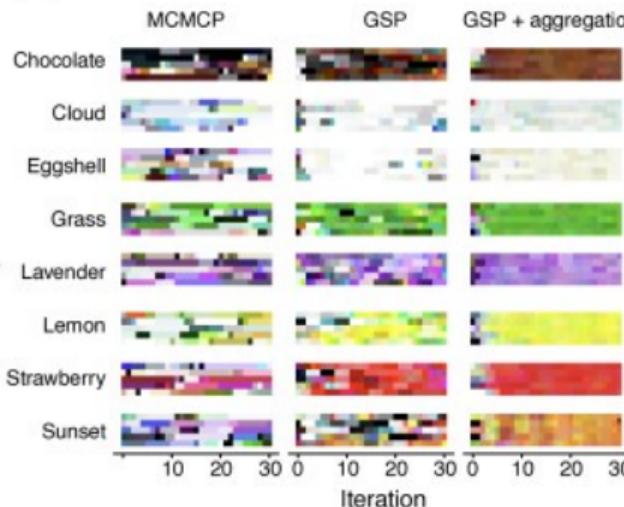
Choose which color best matches the following word:
lavender



GSP Gibbs Sampling with People



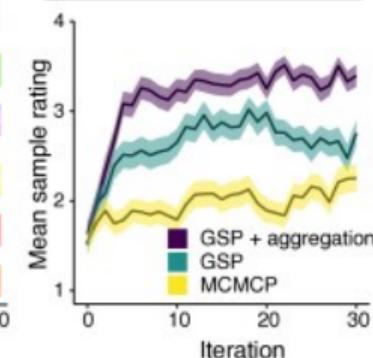
Adjust the slider to match the following word as well as possible: **lavender**

**B****C**

Validation

How well does the color match the following word: **lavender**

not at all a little quite a lot very much

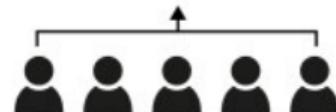


Twarze

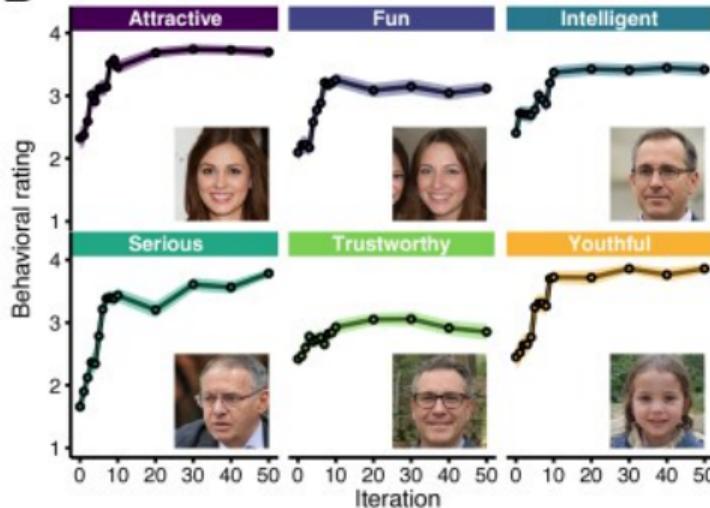
A



Adjust the slider to match the following word as well as possible: **trustworthy**



B



C



Rozkłady warunkowe nie zawsze poprawnie wyznaczają rozkłady brzegowe

Weźmy X i Y , które mają rozkłady brzegowe:

$$\begin{aligned}f(x \mid y) &\propto ye^{-yx}, \text{ gdzie } 0 < x < \infty \\g(y \mid x) &\propto xe^{-xy}, \text{ gdzie } 0 < y < \infty\end{aligned}$$

Równanie całkowatego punktu stałego (fixed point integral equation):

$$f_X(x) = \int \left[\int f_{X \mid Y}(x \mid y) f_{Y \mid X}(y \mid t) dy \right] f_X(t) dt$$

którego rozwiązaniem jest $f_X(x)$

$$f_X(x) = \int \left[\int y e^{-yx} t e^{-ty} dy \right] f_X(t) dt = \int \left[\frac{t}{(x+t)^2} \right] f_X(t) dt$$

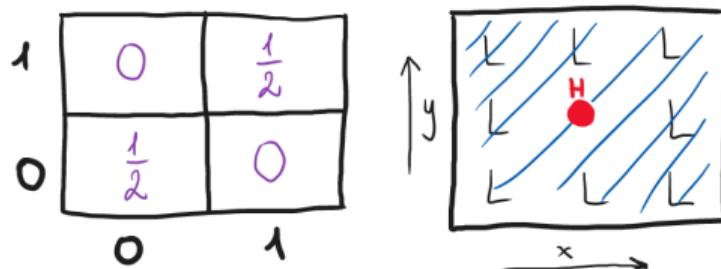
podstawiając $f_X(t) = \frac{1}{t}$:

$$\frac{1}{x} = \int \left[\frac{t}{x+t} \right] \frac{1}{t} dt$$

Rozwiązaniem jest zatem $\frac{1}{x}$, jednak nie jest to funkcja gęstości.
 Ważnym zatem założeniem jest, aby $\int f_X(x) dx < \infty$

Wady

- Długi okres zbieżności, w szczególności przy przestrzeniach o wyższych wymiarach. Ma na to wpływ także kształt rozkładu.
- Początkowe próbki zazwyczaj nie odwzorowują dokładnie oczekiwanej rozkładu, dlatego też często są pomijane.



$$(0,0) \not\Rightarrow (0,1)$$

$$(0,0) \not\Rightarrow (1,0)$$

Bibliografia

-  **Seth Billiau.**
Gibbs sampling explained, May 2021.
-  **George Casella and Edward I George.**
Explaining the gibbs sampler.
The American Statistician, 46(3):167–174, 1992.
-  **Peter M. C. Harrison.**
Gibbs sampling with people, Aug 2020.

DZIĘKUJĘ ZA UWAGĘ :)

Przegląd algorytmów ewolucyjnych

Krzysztof Sawicki

March 2023

Rozkład jazdy

O czym będę mówić:

- Co to jest algorytm ewolucyjny?
- Zastosowania
- Jak on działa?
- Podział algorytmów ewolucyjnych
- Czym obecnie zajmuje się EA
- Przyszłość EA

Co to algorytm ewolucyjny?

Algorytm ewolucyjny

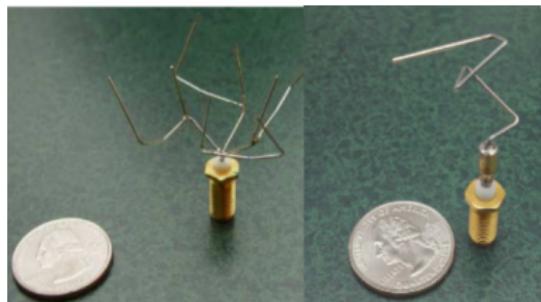
Algorytm wzorowany na biologicznej ewolucji, stosowany do zadań optymalizacyjnych i modelowania.

Zastosowania

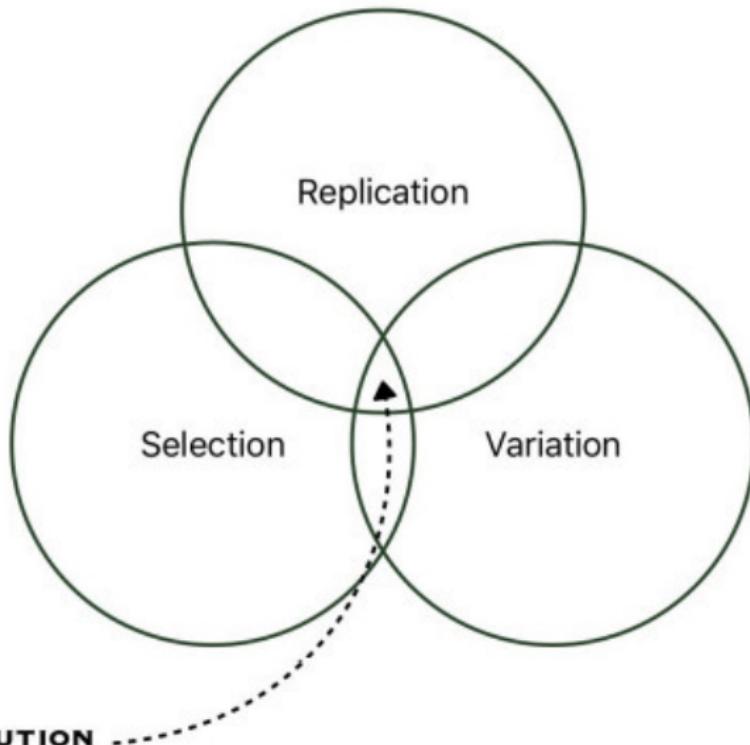
Algorytmy ewolucyjne stosujemy, gdy nic innego nie pomaga. Problemy którymi zajmuje się EA można podzielić na 3 sekcje:

- Optymalizacja zmiennych
- Nowy design
- Usprawnienie nowego rozwiązania

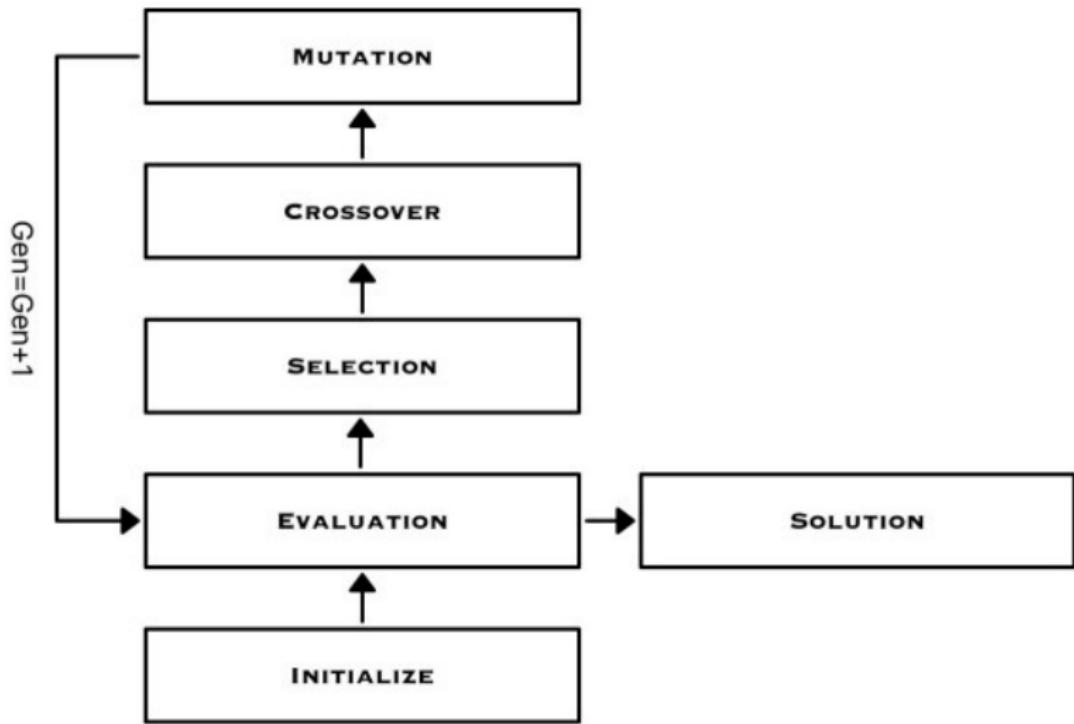
Zastosowania



Ewolucja cyfrowa



Jak to działa?



Prosty podział algorytmów ewolucyjnych

Istnieje wiele rodzajów algorytmów ewolucyjnych, jednakże skupimy się na prostym podziale wyróżniającym:

- Strategie ewolucyjne
- Algorytmy genetyczne
- Programowanie genetyczne

Strategie ewolucyjną

Jest to jeden z najstarszych algorytmów ewolucyjnych. Zwykle skupia się tylko na procesie mutacji i selekcji. ES jest stosowane do optymalizacji ciągłych parametrów. Parametr jest definiowany przez swój typ oraz zakres przedziału (górna i dolna granica). Parametr ciągły może przyjąć dowolną wartość w przedziale wartości. Precyzja określa minimalną wartość zmiany.

Algorytmy genetyczne

Operuje na ciągach, których długość reprezentuje wymiarowość problemu. Ciągi te reprezentują zmienne lub parametry i są przydatne do eksploracji dużej liczby przestrzeni problemowych, które zwykle przekraczają możliwości ludzkie lub tradycyjne metody. Zmienne lub parametry są przekształcane w ciągi o stałej długości, ciągi stanowią jednostki populacji i są ewoluowane za pomocą okien krzyżowania i mutacji.

Jakie problemy rozwiązuje GA?

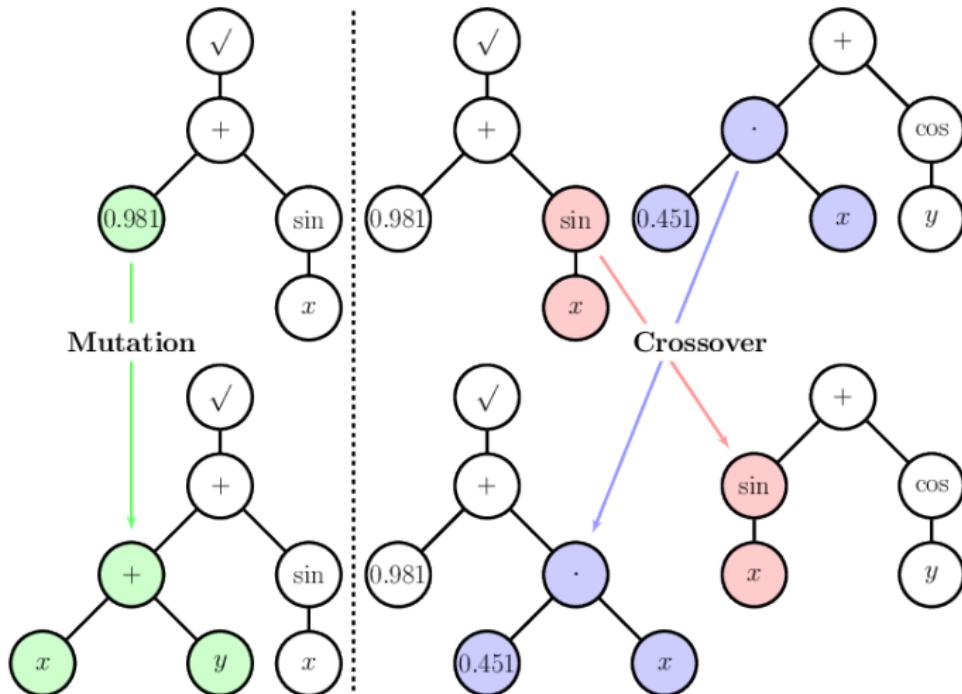
GA radzą sobie z problemami optymalizacji i konfiguracji, w których zbyt wiele zmiennych lub parametrów utrudnia osiągnięcie sukcesu przy użyciu tradycyjnych metod.

- Rozwiązywanie problemów NP
- Projektowanie genetyczne
- Projektowanie obwodów elektrycznych
- Przeszukiwanie

Programowanie genetyczne

zautomatyzowana metoda mająca na celu tworzenie programów komputerowych w oparciu o ogólną definicję problemu. Innymi słowy programowanie genetyczne pozwala, w oparciu o wysokopoziomową definicję mówiącą co ma być zrobione, automatycznie stworzyć program, który owo zagadnienie rozwiąże.

Jakie problemy rozwiązuje GP?



Czym obecnie zajmuje się EA

- Produkcja nowego kodu bez udziału człowieka
- Testowanie/Symulacje
- Chodzące roboty
- Przewidzenie stanu rynku finansowego
- Wizja przyszłych krajobrazów miast
- Projektowanie/Architektura

Co przyniesie przyszłość?

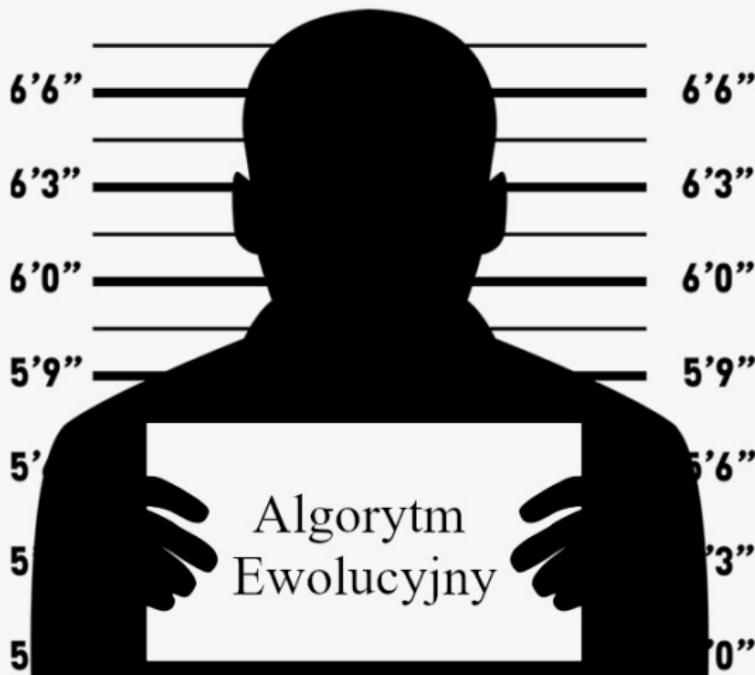
Świat bez algorytmów ewolucyjnych



Świat z algorytmami ewolucyjnymi



Co przyniesie przyszłość?



Kryteria UCA

- Limity odnoszą się do ograniczenia zdolności algorytmów do działania w określonym środowisku
- Wyjaśnialność odnosi się do możliwości wyjaśnienia, w jaki sposób algorytm doszedł do określonego wyniku.
- Przyczynowość odnosi się do możliwości uzyskania odpowiedzi na pytanie "dlaczego?" w kontekście wyniku generowanego przez algorytm.
- Sprawiedliwość odnosi się do możliwości generowania wyników równych i pozbawionych ludzkich uprzedzeń.
- Korekcja polega na naprawianiu problemu po jego zidentyfikowaniu.

Bibliografia



Andrew N. Sloss and Steven Gustafson (2019) *2019 Evolutionary Algorithms Review*.

O wybieraniu najlepszych funkcji optymalizacyjnych

Łukasz Grabarski

Plan prezentacji

- ① Wprowadzenie
- ② CEC
- ③ BBOB
- ④ Testy statystyczne

Metaheurystyka

Metaheurystyka – ogólny algorytm (heurystyka) do rozwiązywania problemów obliczeniowych. Algorytmu metaheurystycznego można używać do rozwiązywania **dowolnego problemu**, który można opisać za pomocą pewnych definiowanych przez ten algorytm pojęć.

Metaheurystyka

- Jak znaleźć najlepszy algorytm?
- Na jakiej podstawie porównywać ze sobą algorytmy?



Metaheurystyka

- Jak znaleźć najlepszy algorytm?
- Na jakiej podstawie porównywać ze sobą algorytmy?

Odpowiedź: Benchmarki i testy



Dlaczego benchmarki?

- Uważa się je za podstawę do opracowywania bardziej złożonych problemów optymalizacyjnych np. wielokryterialnych
- Muszą one symulować stopień trudności rzeczywistych problemów optymalizacyjnych w świecie rzeczywistym.
- Muszą być w stanie wykryć słabości i mocne strony nowych algorytmów optymalizacyjnych, które znacząco się poprawiły w ciągu ostatnich kilku lat.

CEC - to znaczy?

CEC (ang. "IEEE Congress on Evolutionary Computation") to coroczna konferencja poświęcona algorytmom ewolucyjnym i innym algorytmom heurystycznym. Konferencja ta organizowana jest przez Instytut Inżynierów Elektryków i Elektroników (IEEE).

CEC - to znaczy?

Benchmarki służą do oceny skuteczności algorytmów optymalizacji i porównywania wyników różnych algorytmów na tych samych zestawach testowych. CEC Benchmark Functions składają się z 28 funkcji testowych, które zostały zaprojektowane specjalnie dla konkursów optymalizacyjnych w ramach tejże konferencji.

CEC 2021 - Kraków

<https://cec2021 mini.pw.edu.pl/en/organization/organizing-committee.html>

The screenshot shows the official website for the 2021 IEEE Congress on Evolutionary Computation (CEC) held in Krakow, Poland. The page is titled "ORGANIZING COMMITTEE". It features six portraits of the committee members, each with their name below it: Jacek Mardziuk, Hussein Abbass, Yew-Soon Ong, Hemant Singh, Daniel Ashlock, and Oscar Cordón. The website has a navigation bar with links for Home, Calls, Paper Submission, Registration, Program, and Organization.

ORGANIZING COMMITTEE

General Co-Chair



Jacek Mardziuk

General Co-Chair



Hussein Abbass

Program Co-Chair



Yew-Soon Ong

Program Co-Chair



Hemant Singh

Technical Co-Chair



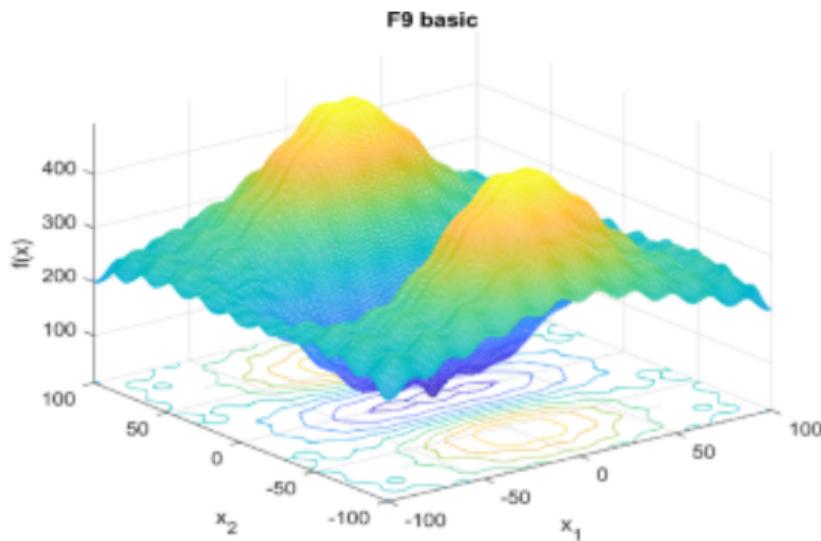
Daniel Ashlock

Technical Co-Chair

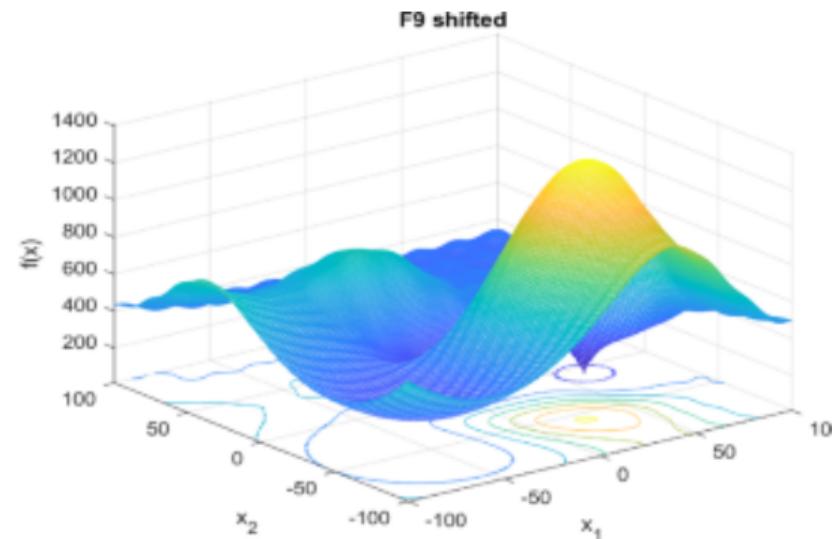


Oscar Cordón

CEC 2021 - benchmarki



(a) Basic function

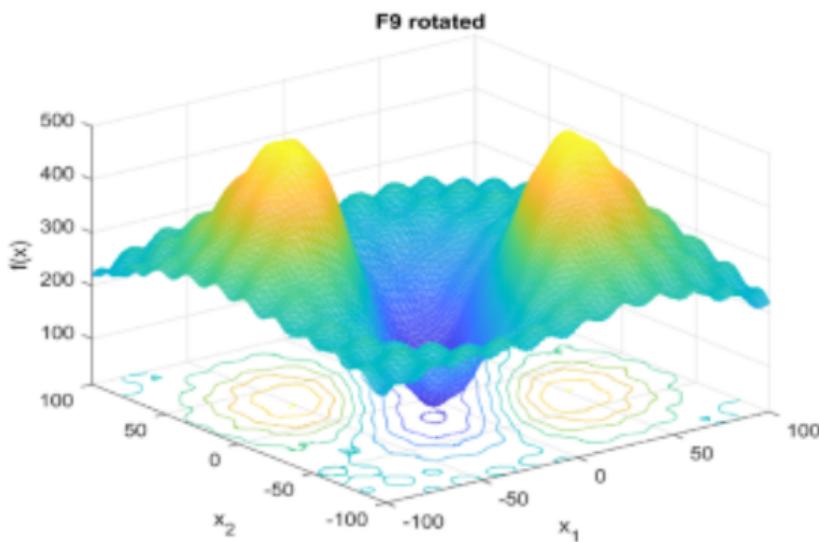


(b) Shifted function

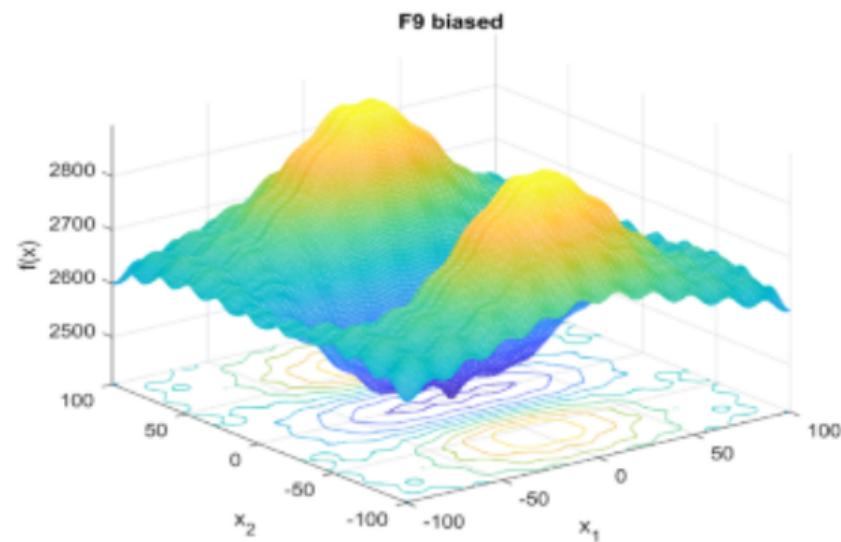


F9 biased

CEC 2021 - benchmarki

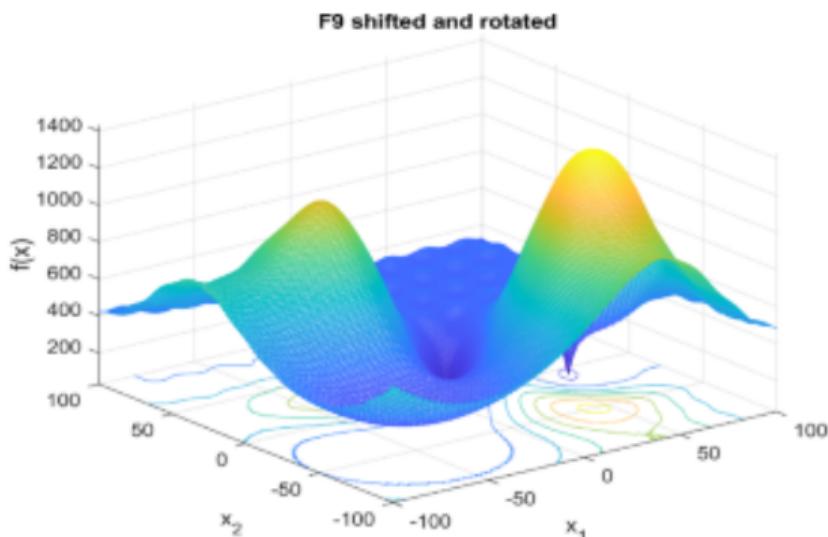


(c) Rotated function

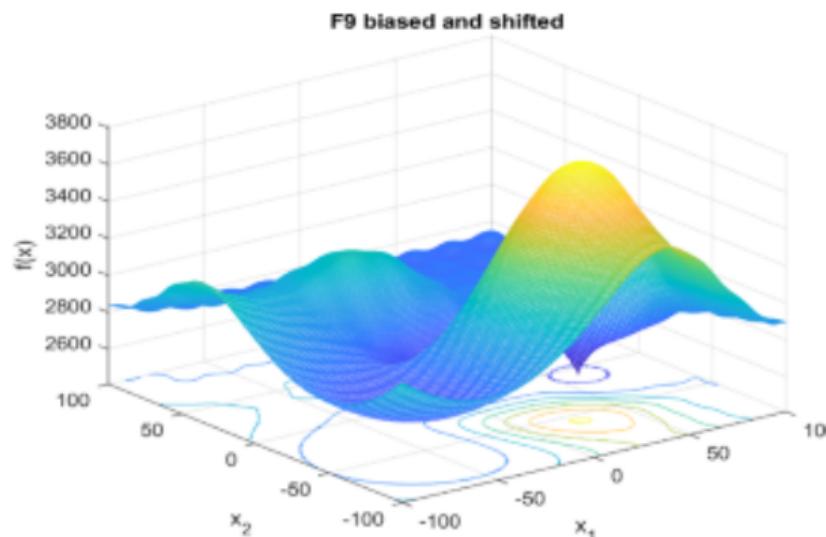


(d) Biased function

CEC 2021 - benchmarki

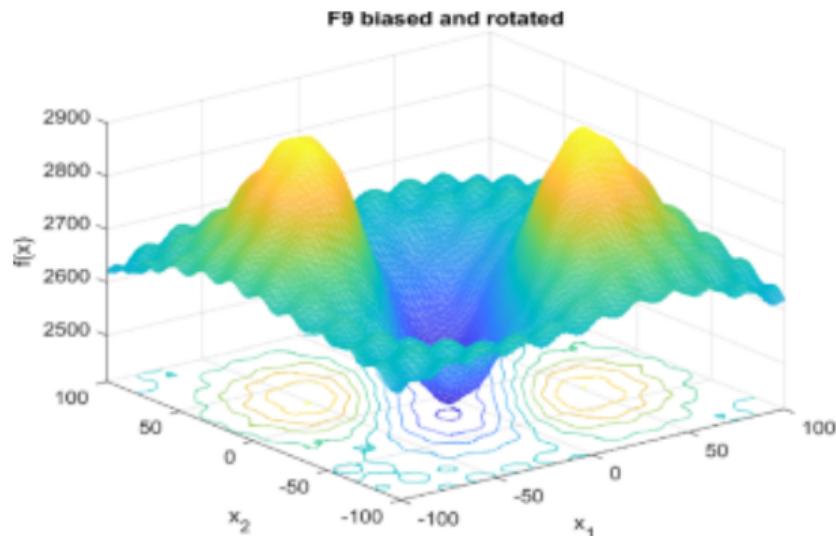


(e) Shifted and rotated function

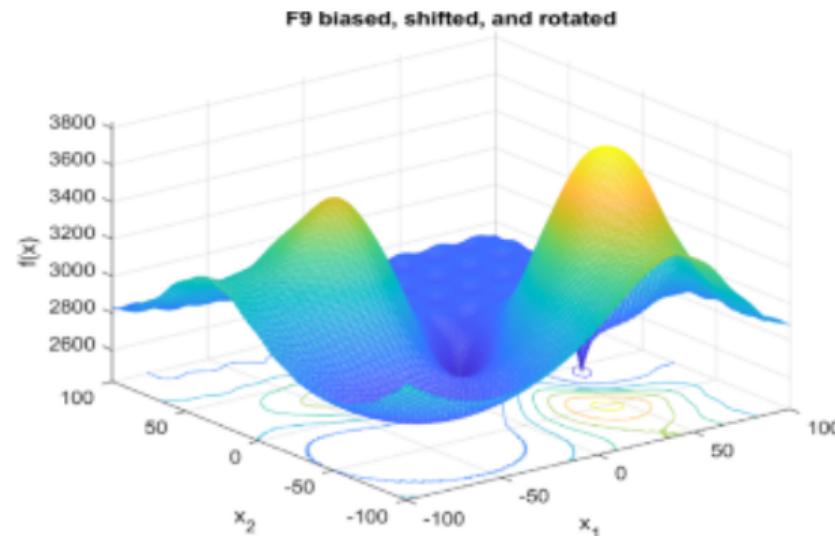


(f) Biased and shifted function

CEC 2021 - benchmarki



(g) Biased and rotated function



(h) Biased,shifted and rotated function

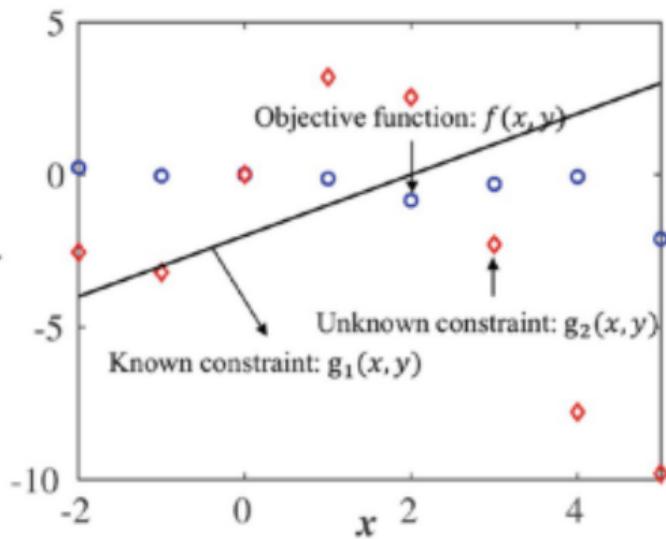
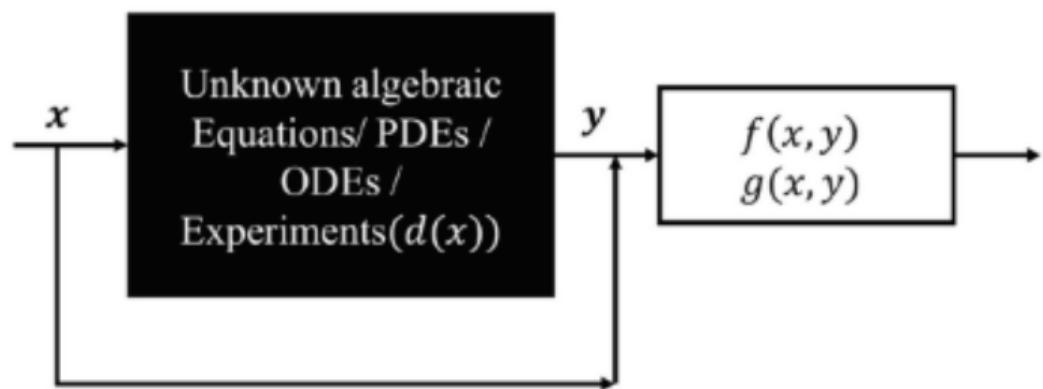
BBO - życiowo



BBO - życiowo



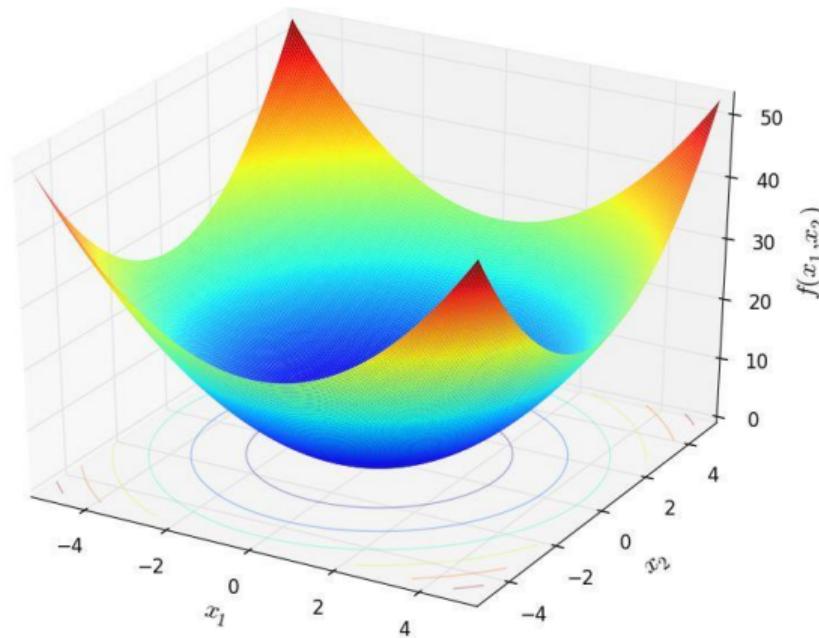
BBO - formalniej



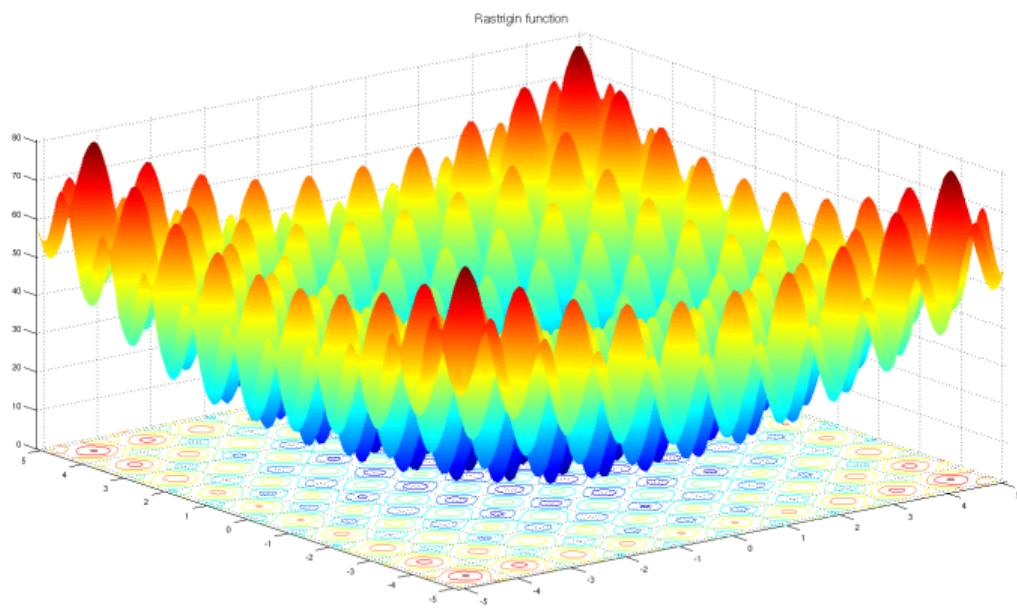
BBOB - co to?

Black Box Optimization Benchmark (BBOB) to zbiór testów, które służą do oceny skuteczności algorytmów optymalizacji globalnej. Jest to benchmark, ponieważ zawiera zestaw standardowych funkcji celu, które pozwalają porównać wyniki różnych algorytmów na tych samych danych testowych.

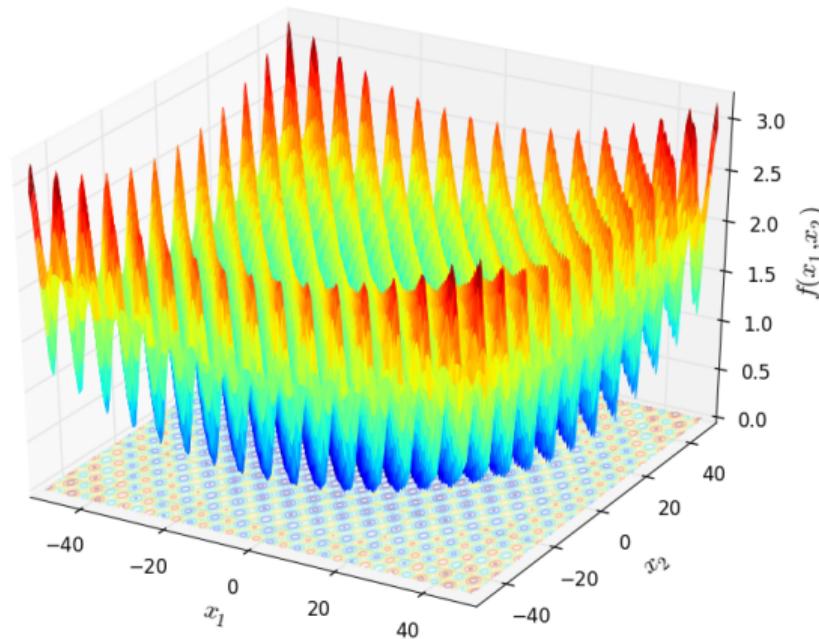
Sphere Function



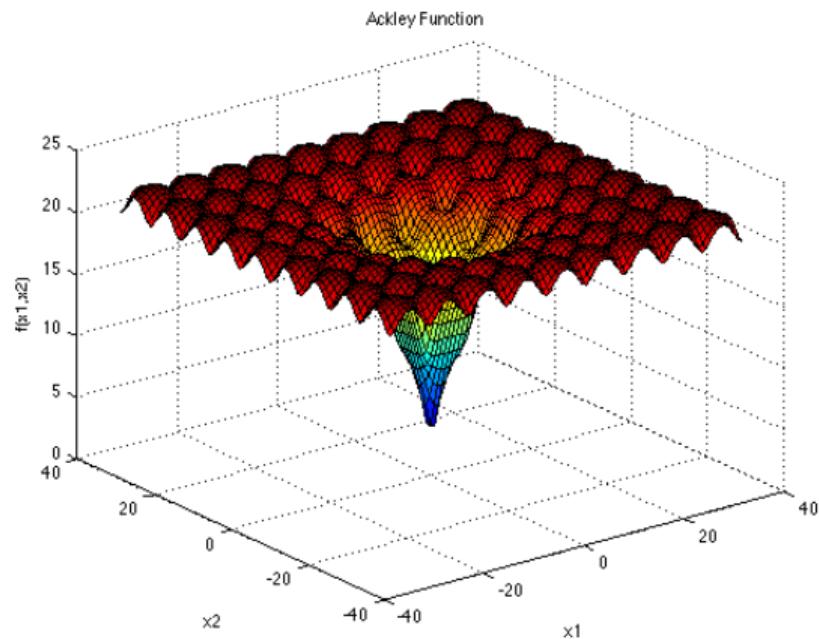
Rastrigin Function



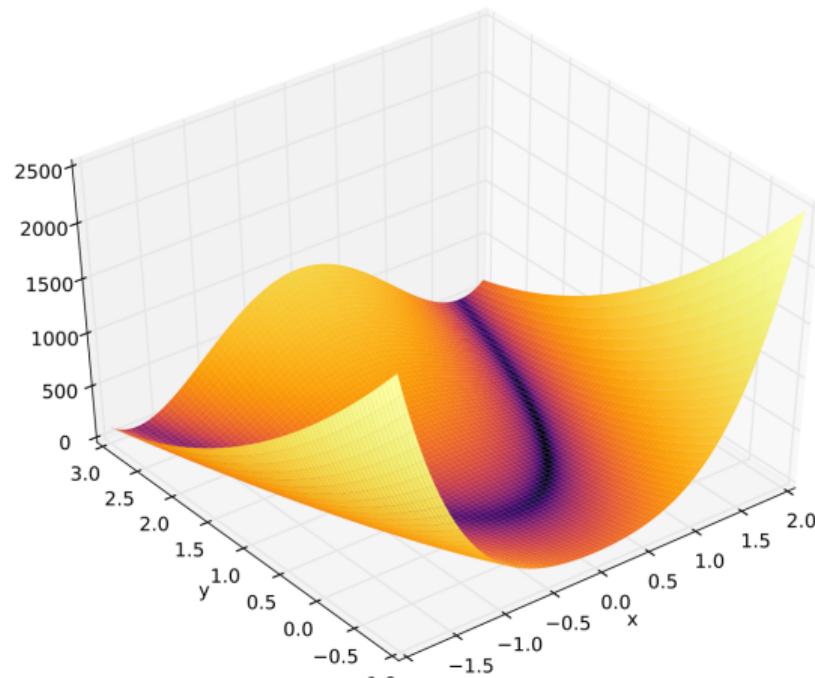
Griewank Function



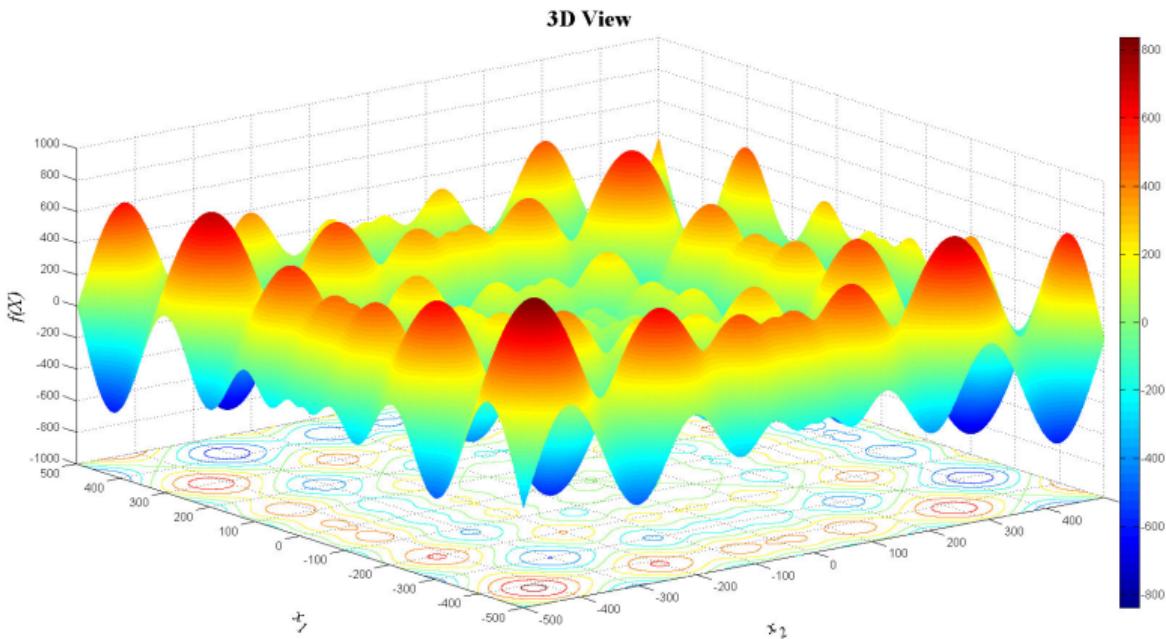
Ackley Function



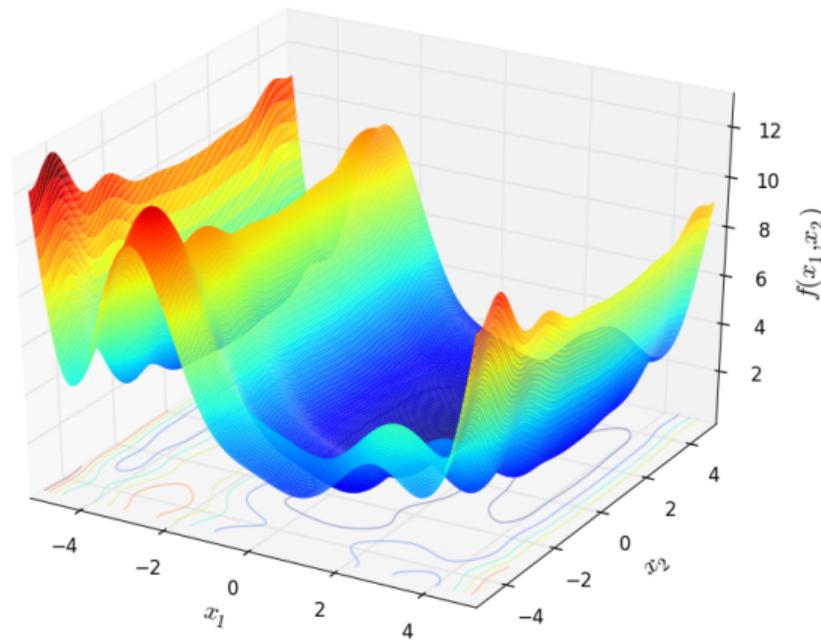
Rosenbrock (banana) Function



Schwefel Function



Levy Function



O testach statystycznych w ogóle

Test statystyczny

Testem statystycznym nazywamy statystykę $\varphi : X \rightarrow \{0, 1\}$, której wartość 1 interpretujemy jako wskazanie odrzucenia hipotezy zerowej, natomiast wartość 0 — jako brak podstaw do odrzucenia hipotezy zerowej.

O testach statystycznych w ogóle

Hipoteza

Hipoteza (gr. *hypóthesis* = założenie) — zdanie przyjęte jako założenie, w celu wyjaśnienia jakiegoś zjawiska i wymagające sprawdzenia. Zazwyczaj mamy **hipotezę zerową** H_0 i odpowiadającą jej **hipotezę alternatywną** H_1 .

Jak sprawdzać hipotezy?

- ① Stawiamy hipotezę zerową H_0 i alternatywną H_1
- ② Ustalamy model matematyczny
- ③ Przyjmujemy poziom istotności α
- ④ Liczymy wartość statystyki testowej T
- ⑤ Liczymy wartość istotności testu p (ang. $p - value$)
- ⑥ Decydujemy: jeśli $p \leq \alpha$ odrzucamy H_0

Test Kołmogorowa - Smirnowa

Założenia

Weźmy dwie próbki proste z ciągłych rozkładów:

$X = (X_0, \dots, X_n)$ F oraz $Y = (Y_0, \dots, Y_m)$ G .

Interesuje nas problem: $H_0 : F = G$ vs. $H_1 = F \neq G$

Test Kołmogorowa - Smirnowa

Statystyka testowa

Rozważamy następującą statystykę:

$$D_{n,m} = \sup_x |\hat{F}_n(x) - \hat{G}_m(x)|$$

Hipotezę zerową będziemy odrzucać na poziomie istotności α gdy: $D_{n,m} \geq D_{n,m}(1 - \alpha)$, gdzie $D_{n,m}(1 - \alpha)$ jest kwantylem rzędu $1 - \alpha$ rozkładu statystyki testowej (przy założeniu prawdziwości H_0).

Reasumując...



Tyle wystarczy...

Dziękuję za uwagę!

Bibliografia

<https://link.springer.com/article/10.1007/s00521-022-07788-z>
<https://pl.wikipedia.org/wiki/Metaheurystyka>
<https://github.com/numbbo/coco/tree/v2.3>
<https://pages.mini.pw.edu.pl/~grzegorzewskip/>

Algorytm SHADE

Dawid Płudowski

Wydział Matematyki i Nauk Informatycznych

18-03-2023

Differential Evolution (DE)

Pomysł: Wylosujmy punkty w przestrzeni (populacja). Pozwólmy im, zgodnie z ustaloną strategią, losowo łączyć się ze sobą (mutacja) z każdą kolejną iteracją (generacja). Każda mutacja to dodanie do oryginalnego punktu przeskalowanego kierunku wyznaczonego poprzez inne punkty. To, czy mutacja zajdzie, jest wyznaczane losowo.

Notacja

- N - liczność populacji
- U - zbiór strategii mutacji
- F - skala mutacji
- CR - *Crossover rate*, częstość mutacji
- G - numer generacji mutacji

Przykładowe strategie

Częścią wspólną każdej strategii jest losowanie permutacji populacji (r_1, \dots, r_N) . Zmutowane punkty oznaczamy (ν_1, \dots, ν_N) i wyznaczamy za pomocą strategii:

Strategie

- **rand/1:** $\nu_{i,G} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G})$
- **best/1:** $\nu_{i,G} = x_{best,G} + F(x_{r_1,G} - x_{r_2,G})$
- **current-to-best/1:**

$$\nu_{i,G} = x_{i,G} + F(x_{best,G} - x_{i,G}) + F(x_{r_1,G} - x_{r_2,G})$$

Crossover rate

Po dokonaniu mutacji decydujemy się, które z nich uwzględnimy w kolejnej generacji. **Uwaga:** decyzja dotyczy elementu wektora, nie całego punktu!

$$u_{j,i,G} = \begin{cases} \nu_{j,i,G}, & CR > \text{unif}(0, 1) \\ x_{j,i,G}, & \text{w.p.p} \end{cases} \quad (1)$$

Jeżeli mutacja była dla nas pomyślna to ją zachowujemy, w przeciwnym wypadku powracamy do oryginalnego punktu:

$$x_{i,G+1} = \begin{cases} u_{i,G}, & f(u_{i,G}) < f(x_{i,G}) \\ x_{i,G}, & \text{w.p.p} \end{cases} \quad (2)$$

Problemy DE

Dla całego procesu wybieramy liczność (N), strategię (U_i), skalę mutacji (F) i częstość mutacji (CR). Parametry globalne mogą być w pewnych miejscach przestrzeni nieadekwatne, niektóre mogą charakteryzować się złym dobraniem do konkretnego problemu optymalizacyjnego.

JADE

JADE to prekursor algorytmu **SHADE**. Rozwiązuje problem globalnych parametrów przydzielając dla każdego osobnika z osobna dynamicznie parametry mutacji.

Nowa strategia

current-to-pbest/1: $\nu_{i,G} = x_{i,G} + F(x_{pbest,G} - x_{i,G}) + F(x_{r_1,G} - x_{r_2,G})$

x_{pbest} oznacza losowo wybrany punkt z p najlepszych.

Dynamiczna zmiana parametrów

Dla każdego osobnika z populacji stosowany jest algorytm:

- w pierwszej iteracji zainicjalizuj $F_i = 0.5$ i $CR_i = 0.5$
- wyznacz średnią wartość $mean_L(F)$ i $mean_A(CR)$ dla tych osobników, których mutacja zakończyła się sukcesem (nie została odrzucona w ostatnim kroku)
- wyznacz średnią wartość $\mu_{F_G} = (1 - c)\mu_{F_{G-1}} + c * mean_L(F)$ i analogicznie μ_{CR} (c - tzw. *learning_rate*, parametr algorytmu)
- wylosuj nowe F_i i CR_i z rozkładów $Cauchy(\mu_F, 0.1)$ i $N(\mu_{CR}, 0.1)$ odpowiednio
- powtarzaj kroki 2-4 w kolejnych iteracjach

$mean_A$ oznacza średnią arytmetyczną, a $mean_L$ średnią Lehmera.

Opcjonalne archiwum

W tradycyjnym algorytmie podczas każdej generacji "usuwamy" poprzednią populację. Jeżeli korzystamy z archiwum (P), osobnik, który nie przetrwał (znaleźliśmy lepszy punkt na jego podstawie), trafia do archiwum. Punkt $x_{r_2, G}$ z poprzedniego slajdu losowany jest z sumy zbioru X i P . Co daje nam taka strategia:

- zachowujemy punkty, które mogłyby być potencjalnie obiecujące, ale źle zmutowały
- zwiększamy wielkość zbioru, z którego losujemy mutacje (większa różnorodność)

Problemy JADE

W algorytmie JADE dobieramy dynamicznie parametry na podstawie podpopulacji, której udało się ewoluować z sukcesem. Probabilistycznie możliwe jest, że taka podpopulacja miała bardzo złe parametry i jej "sukces" był przypadkowy. Potrzebujemy algorytmu, który będzie odporny na pojedyncze "szczęśliwe" generacje, które bazują na złych parametrach.

SHADE

SHADE (*Success History DE*) rozwija algorytm JADE o zachowywanie informacji o poprzednich sukcesach.

- jeżeli parametry sprawdziły się w pewnej generacji, kolejne (nawet odległe) generacje mogą z nich ponownie skorzystać
- "szczęśliwe" parametry, które w rzeczywistości nie są odpowiednie do zadania nie determinują kolejnych generacji
- zwiększenie entropii algorytmu - mniejsza szansa na zatrzymanie się na niekorzystnych parametrach

Pamięć poprzednich generacji

Zamiast:

$$\mu_F, \mu_{CR}$$

Zdefiniujmy:

$$M_F = (M_{F,1}, \dots, M_{F,H}), \quad M_{CR} = (M_{CR,1}, \dots, M_{CR,H})$$

Podczas tworzenia nowej generacji wybierzmy parametry μ_F, μ_{CR} ze zbiorów M_F, M_{CR} . Wyznaczmy, tak samo jak w JADE, nowe parametry F i CR i nadpiszmy nim $M_{F,k}, M_{CR,k}$, gdzie $k = mod(G, H)$.

Wyznaczanie średniej

Zamiast wyznaczać zwykłą średnią, możemy nadać jej odpowiednie wagi, które będą faworyzowały szczególnie "wybitne" osobniki, tzn. te, które w trakcie trwania generacji najbardziej zmniejszyły wartość $f(x)$.

Średnia ważona

$$mean_{WA}(S_{CR}) = \sum_{k=1}^{|S_{CR}|} \omega_k * S_{CR,k}$$

$$mean_{WL}(S_F) = \frac{\sum_{k=1}^{|S_{CR}|} \omega_k * S_{F,k}^2}{\sum_{k=1}^{|S_{CR}|} \omega_k * S_{F,k}}$$

$$\omega_k = \frac{\Delta f_k}{\sum_{i=1}^{|S_{CR}|} \Delta f_i}$$

Dynamiczna zmiana current-to-*p*best

Z jednej strony zmiana w kierunku najlepszego punktu jest intuicyjnie uzasadniona; z drugiej strony ryzykujemy, że najlepszy punkt utknął w minimum lokalnym i cała reszta populacji również tam trafi. Kompromisem pomiędzy tymi podejściami jest losowanie p dla każdego punktu z pewnego rozkładu $rand[p_{min}, 0.2]$.

Benchamark

<i>F</i>	SHADE Mean (Std Dev)	CODE Mean (Std Dev)	EPSDE Mean (Std Dev)	JADE Mean (Std Dev)	dynNPjDE Mean (Std Dev)
<i>F</i> ₁	0.00e+00 (0.00e+00)	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈
<i>F</i> ₂	9.00e+03 (7.47e+03)	9.78e+04 (4.81e+04)−	1.37e+06 (5.23e+06)−	7.67e+03 (5.66e+03)≈	9.52e+04 (4.09e+04)−
<i>F</i> ₃	4.02e+01 (2.13e+02)	1.08e+06 (3.03e+06)−	1.75e+08 (5.39e+08)−	4.71e+05 (2.35e+06)−	1.71e+06 (2.54e+06)−
<i>F</i> ₄	1.92e+04 (3.01e+04)	8.18e-02 (1.09e-01)−	8.08e+03 (2.56e+04)−	6.09e+03 (1.33e+04)−	4.76e+01 (4.75e+01)−
<i>F</i> ₅	0.00e+00 (0.00e+00)	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈
<i>F</i> ₆	5.96e-01 (3.73e+00)	4.16e+00 (9.00e+00)−	9.27e+00 (1.33e+00)−	2.07e+00 (7.17e+00)−	1.19e+01 (1.66e+00)−
<i>F</i> ₇	4.60e+00 (5.39e+00)	9.32e+00 (6.34e+00)−	5.88e+01 (4.29e+01)−	3.16e+00 (4.13e+00)≈	2.62e+00 (1.59e+00)≈
<i>F</i> ₈	2.07e+01 (1.76e+01)	2.08e+01 (1.18e+01)≈	2.09e+01 (5.32e+02)−	2.09e+01 (4.93e+02)−	2.10e+01 (3.98e+02)−
<i>F</i> ₉	2.75e+01 (1.77e+00)	1.45e+01 (2.90e+00)≈	3.50e+01 (4.21e+00)−	2.65e+01 (1.96e+00)+	2.20e+01 (5.12e+00)+
<i>F</i> ₁₀	7.69e-02 (3.58e-02)	2.71e-02 (1.50e-02)+	1.02e-01 (5.65e-02)−	4.04e-02 (2.37e-02)+	3.63e-02 (2.34e-02)+
<i>F</i> ₁₁	0.00e+00 (0.00e+00)	0.00e+00 (0.00e+00)≈	1.95e-02 (1.39e-01)≈	0.00e+00 (0.00e+00)≈	0.00e+00 (0.00e+00)≈
<i>F</i> ₁₂	2.30e+01 (3.73e+00)	3.98e+01 (1.21e+01)−	4.94e+01 (9.28e+00)−	2.29e+01 (5.45e+00)≈	4.07e+01 (8.81e+00)−
<i>F</i> ₁₃	5.03e+01 (1.34e+01)	8.04e+01 (2.74e+01)−	7.68e+01 (1.72e+01)−	4.67e+01 (1.37e+01)≈	7.10e+01 (1.72e+01)−
<i>F</i> ₁₄	3.18e-02 (2.33e-02)	3.60e+00 (4.09e+00)−	3.99e-02 (6.00e-01)−	2.86e-02 (2.35e-02)≈	9.39e-03 (1.40e-02)+
<i>F</i> ₁₅	3.22e+03 (2.64e+02)	3.66e+03 (5.31e+02)−	6.75e+03 (7.60e+02)−	3.24e+03 (3.17e+02)≈	4.39e+03 (4.72e+02)−
<i>F</i> ₁₆	9.13e-01 (1.85e-01)	3.38e-01 (2.03e-01)+	2.48e+00 (2.88e-01)−	1.84e+00 (6.27e-01)−	2.32e+00 (2.83e-01)−
<i>F</i> ₁₇	3.04e+00 (3.83e-14)	3.04e+01 (1.17e-02)−	3.04e+01 (2.51e-02)−	3.04e+01 (1.95e-14)−	3.04e+01 (1.78e-03)≈
<i>F</i> ₁₈	7.25e+00 (5.58e+00)	6.69e+01 (9.23e+00)+	1.37e+02 (1.12e+01)−	7.76e+01 (5.91e+00)−	1.35e+02 (1.24e+01)−
<i>F</i> ₁₉	1.36e+00 (1.20e-01)	1.61e+00 (3.58e-01)−	1.84e+00 (2.00e-01)−	1.44e+00 (8.71e-02)−	1.27e+00 (1.09e-01)+
<i>F</i> ₂₀	1.05e+01 (6.04e-01)	1.06e+01 (6.69e-01)≈	1.30e+01 (6.33e-01)−	1.04e+01 (5.82e-01)≈	1.13e+01 (4.14e-01)−
<i>F</i> ₂₁	3.09e+02 (5.65e+01)	3.02e+02 (9.02e+01)≈	3.05e+02 (8.06e+01)≈	3.04e+02 (6.68e+01)≈	2.94e+02 (8.29e+01)≈
<i>F</i> ₂₂	9.81e+01 (2.52e+01)	1.17e+02 (9.96e+00)−	3.09e+02 (1.12e+02)−	9.39e+01 (3.08e+01)≈	1.03e+02 (2.57e+01)−
<i>F</i> ₂₃	3.51e+03 (4.11e+02)	3.56e+03 (6.12e+02)≈	6.74e+03 (8.20e+02)−	3.26e+03 (4.01e+02)≈	4.36e+03 (4.61e+02)−
<i>F</i> ₂₄	2.05e+02 (5.29e+00)	2.21e+02 (9.28e+00)−	2.91e+02 (7.08e+00)−	2.17e+02 (1.57e+01)−	2.04e+02 (4.22e+00)≈
<i>F</i> ₂₅	2.59e+02 (1.96e+01)	2.57e+02 (6.55e+00)≈	2.99e+02 (3.29e+00)−	2.74e+02 (1.06e+01)−	2.55e+02 (7.91e+00)≈
<i>F</i> ₂₆	2.02e+02 (1.48e+01)	2.18e+02 (4.48e+01)−	3.56e+02 (6.49e+01)−	2.15e+02 (4.11e+01)−	2.00e+02 (3.06e+03)+
<i>F</i> ₂₇	3.88e+02 (1.09e+02)	6.20e+02 (1.01e+02)−	1.21e+03 (7.42e+01)−	6.70e+02 (2.40e+02)−	3.90e+02 (9.12e+01)≈
<i>F</i> ₂₈	3.00e+02 (0.00e+00)	3.00e+02 (0.00e+00)≈	3.00e+02 (0.00e+00)≈	3.00e+02 (0.00e+00)≈	3.00e+02 (0.00e+00)≈
	+	4	0	2	5
	-	15	23	10	13
	≈	9	5	16	10

Rysunek: Benchmark algorytmów.

Bibliografia

- *Success-History Based Parameter Adaptation for Differential Evolution*, Ryoji Tanabe, Alex Fukunaga,
<http://metahack.org/CEC2013-SHADE.pdf>

CMA-ES

Antoni Zajko

Warsaw University of Technology

2023

CMA-ES wysokopoziomowo

Cel : Zminimalizować funkcję f

1. Zainicjalizuj parametry:

$$1.1 \Sigma^{(0)} = I$$

$$1.2 \mu^{(0)} \in \mathbb{R}^n$$

$$1.3 \sigma^{(0)} \in \mathbb{R}$$

2. Dopóki nie zostanie spełnione ustalone kryterium:

2.1 Wygeneruj p punktów x_i tak, że : $x_i \sim \mathcal{N}(\mu^{(g)}, \sigma^{(g)}\Sigma^{(g)})$

2.2 Zaktualizuj μ , Σ , σ na podstawie najlepszych punktów z x_i

Wizualizacja działania

<https://blog.otoro.net/assets/20171031/rastrigin/cmaes.gif>

Aktualizacja μ (wartości oczekiwanej rozkładu)

1. Wybierz $q \leq p$ "najlepszych" punktów spośród x
2. Posortuj te punkty od najlepszych do najgorszych
3. $\mu^{(g+1)} = \mu^{(g)} + c_\mu \sum_{i=1}^q w_i (x_i - \mu)'$

Aktualizacja μ (wartości oczekiwanej rozkładu)

- ▶ q jest hiperparametrem tej metody.
- ▶ Często $w_i = \frac{1}{q}$, chociaż niekoniecznie.
- ▶ Często się przyjmuje, że $\sum_{i=1}^q w_i = 1$, chociaż tak niekoniecznie musi być. Suma tych wag nie musi być nawet dodatnia.
- ▶ c_μ jest parametrem uczenia dla μ (jednym z wielu w tej metodzie).

Variance effective selection mass

$$q_{eff} = \left(\frac{||w||_1}{||w||_2} \right)^2 = \frac{1}{\sum_{i=1}^q w_i^2}$$

Dobór wag

1. Dla $w_i = \frac{1}{q}$, $q_{eff} = q$
2. Uważa się, że jeżeli $q_{eff} \approx \frac{q}{2}$, to dobór wag jest w porządku
3. Przykładowy dobór wag: $w_i \propto q - i + 1$, gdzie $q \approx \frac{p}{2}$

Adaptacja macierzy kowariancji

Estymacja macierzy kowariancji

$$\Sigma = \sum_{i=1}^q w_i (x_i - \mu)(x_i - \mu)^\top$$

Rank- q -update

$$\Sigma^{(g+1)} = (1 - c_q) \Sigma^{(g)} + c_q \sum_{i=1}^q w_i \left(\frac{x_i - \mu}{\sigma^{(g)}} \right) \left(\frac{x_i - \mu}{\sigma^{(g)}} \right)^\top$$

Gdzie c_q to jest learning rate. Ważny jest odpowiedni jego dobór. Za mały learning rate będzie powodował zbyt wolne uczenie, natomiast za duży spowoduje degeneracje macierzy kowariancji. Eksperymenty pokazują, że $c_q \approx \frac{q_{\text{eff}}}{n^2}$ jest optymalnym wyborem.

Ścieżka ewolucji

$$p_c^{(g)} = \begin{cases} 0, & g = 0 \\ (1 - c_c)p_c^{(g-1)} + \sqrt{c_c(2 - c_c)\mu_{\text{eff}} \frac{m^{(g)} - m^{(g-1)}}{c_m \sigma^{(g)}}}, & g > 0 \end{cases}$$

Gdzie c_c jest kolejnym learning rate.

Rank-One-Update

Wykorzystując ścieżkę ewolucji, można aktualizować macierz kowariancji:

$$\Sigma^{(g+1)} = (1 - c_1)\Sigma^{(g)} + c_1 p_c^{(g+1)}(p_c^{(g+1)})^\top$$

Gdzie c_1 jest kolejnym learning rate.

Finalny rezultat

$$\begin{aligned}\Sigma^{(g+1)} &= (1 - c_1 - c_q) \Sigma^{(g)} \\ &\quad + c_1 p_c^{(g+1)} (p_c^{(g+1)})^\top \\ &\quad + c_q \sum_{i=1}^q w_i \left(\frac{x_i - \mu^{(g)}}{\sigma^{(g)}} \right) \left(\frac{x_i - \mu^{(g)}}{\sigma^{(g)}} \right)^\top\end{aligned}$$

Zalety

- ▶ Dużo lepiej sobie radzi z optimami lokalnymi, niż tradycyjne algorytmy np. BFGS,
- ▶ Nie wymaga dużej populacji, dzięki odpowiedniemu dopasowaniu współczynników uczenia,
- ▶ Inwariancja ze względu na skalowanie, transformacje monotnoniczne, transformacje przestrzeni zachowujące kąty.

Wady

- ▶ Kosztowność obliczeniowa, ponieważ w każdej iteracji jest wymagane policzenie kilku macierzy kowariancji,
- ▶ inwariancja ze względu na skalowanie, transformacje monotoniczne, transformacje przestrzenie zachowujące kąty,
- ▶ Nie nadaje się do problemów z dynamicznym zbiorem, po którym się optymalizuje tzn. takim, którego postać zależy od optymalizowanych parametrów,
- ▶ Nie działa ze zmiennymi kategorycznymi.

Zbieżność

Function	f_{stop}	init	n	CMA-ES	DE	RES	LOS
$f_{\text{Ackley}}(\mathbf{x})$	1e-3	$[-30, 30]^n$	20	2667	.	.	6.0e4
			30	3701	12481	1.1e5	9.3e4
			100	11900	36801	.	.
$f_{\text{Griewank}}(\mathbf{x})$	1e-3	$[-600, 600]^n$	20	3111	8691	.	.
			30	4455	11410 *	$8.5e-3/2e5$.
			100	12796	31796	.	.
$f_{\text{Rastrigin}}(\mathbf{x})$	0.9	$[-5.12, 5.12]^n$ DE: $[-600, 600]^n$	20	68586	12971	.	9.2e4
			30	147416	20150 *	1.0e5	2.3e5
			100	1010989	73620	.	.
$f_{\text{Rastrigin}}(\mathbf{A}\mathbf{x})$	0.9	$[-5.12, 5.12]^n$	30	152000	$171/1.25e6$ *	.	.
			100	1011556	$944/1.25e6$ *	.	.
$f_{\text{Schwefel}}(\mathbf{x})$	1e-3	$[-500, 500]^n$	5	43810	2567 *	.	7.4e4
			10	240899	5522 *	.	5.6e5

Bibliografia

1. Opis algorytmu
2. Benchmark algorytmu

Optymalizacja rojem cząstek czyli dwa słowa o PSO

Maciej Szpetmański

April 2023

Plan prezentacji

- ① Krótko o PSO
- ② Czy działa?
- ③ Badania nad PSO
- ④ Modyfikacje w algorytmie

Co to jest?

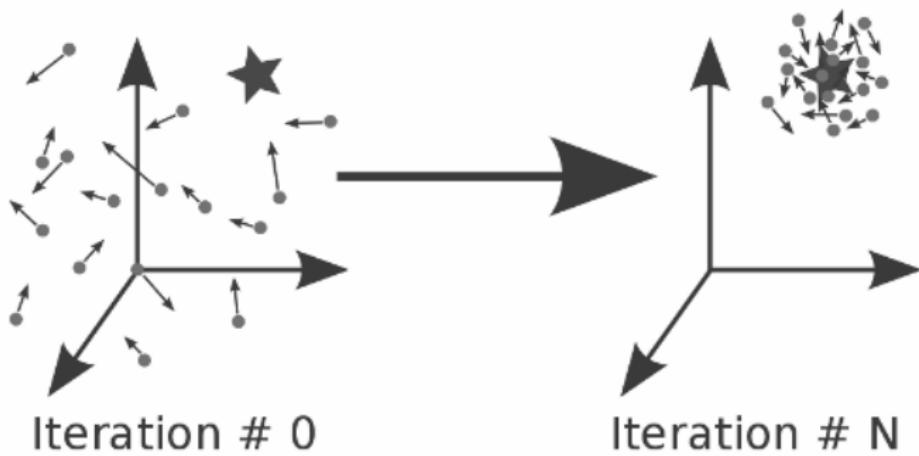
PSO

Algorytm stochastyczny obliczeniowy, naśladujący zachowania grup zwierząt.

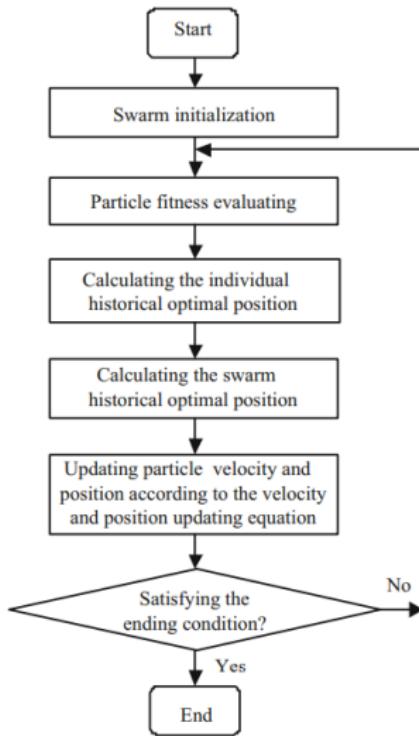
Co to jest?



Jak działa?



Wprowadźmy to w życie



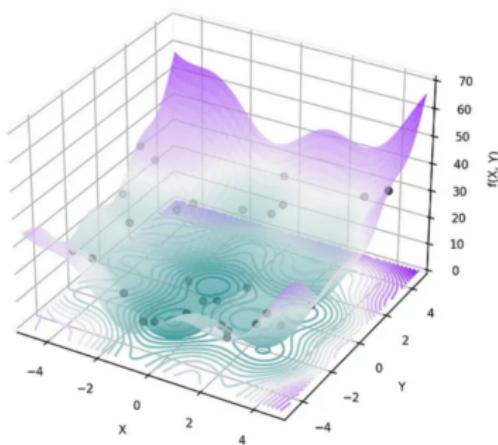
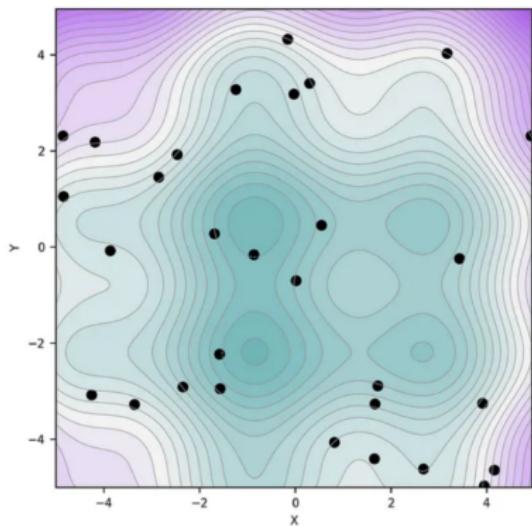
$$p_{i,t+1}^d = \begin{cases} x_{i,t+1}^d, & \text{if } f(X_{i,t+1}) < f(P_{i,t}) \\ p_{i,t}^d, & \text{otherwise} \end{cases}$$

$$\begin{aligned} v_{i,t+1}^d &= \omega * v_{i,t}^d + c_1 * rand * (p_{i,t}^d - x_{i,t}^d) \\ &\quad + c_2 * rand * (p_{l,t}^d - x_{i,t}^d) \end{aligned}$$

Wzór: PSO podstawowe, [1]

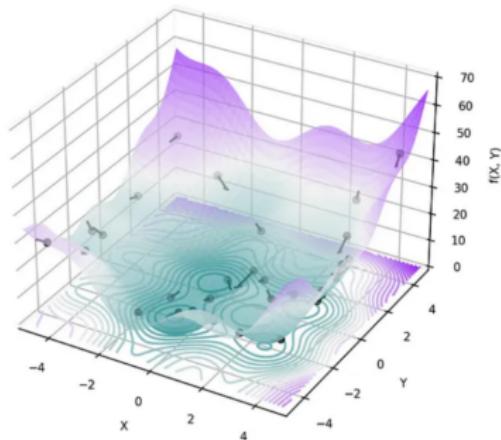
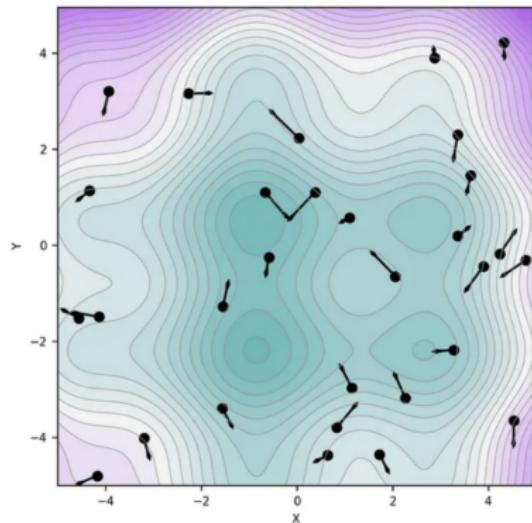
Skuteczność

Random initialization of $N = 30$ particles

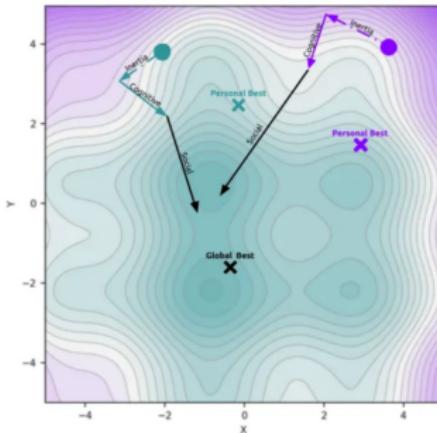


Skuteczność

Random initialization of $N = 30$ particles with velocity



Skuteczność



Salesman

Kto się tym interesuje

Obecnie algorytm PSO jest przedmiotem wielu badań na całym świecie. Skupiają się one głównie na następujących obszarach:

- Badania nad teorią
- Zmiany w strukturze
- Wpływ parametrów
- Wpływ różnych struktur topologicznych
- Algorytm równoległy (PPSO)
- Dyskretny PSO
- Wieloobiektowe PSO

Bibliografia



Dongshu Wang, Dapei Tan, Lei Liu (2017)
Particle swarm optimization algorithm: an overview

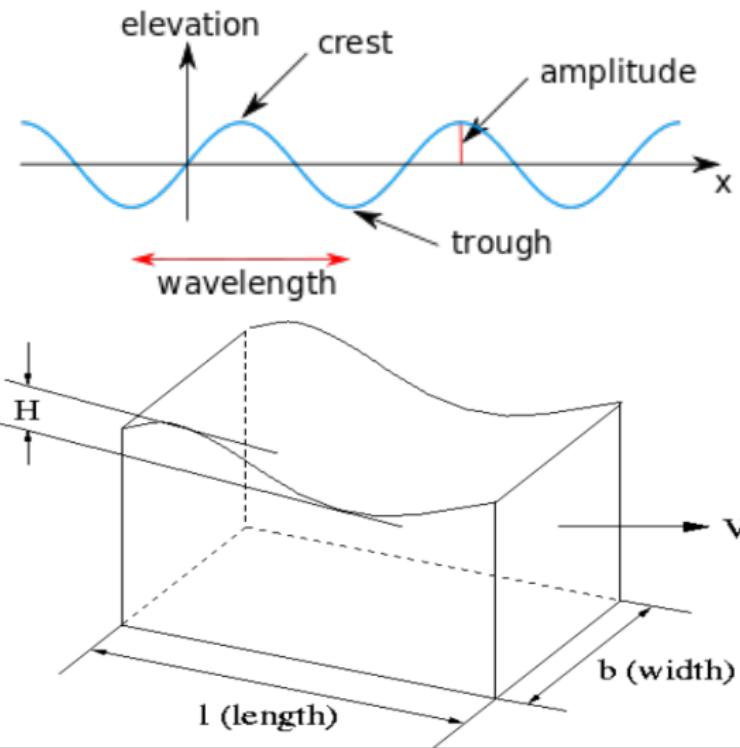
Water wave optimization

Karolina Maczka

Wydział Matematyki i Nauk Informacyjnych
Politechnika Warszawska

March, 2023

Teoria fal wodnych - pierwszy Newton



$$x = A(e^{z_0/c} - e^{-z_0/c}) \sin \frac{x_0}{c} \quad (1)$$

$$z = A(e^{z_0/c} - e^{-z_0/c}) \cos \frac{x_0}{c} \quad (2)$$

- x i z określają odpowiednio poziome i pionowe przesunięcia poszczególnych cząstek o położeniu początkowym (x_0, z_0)
- A to funkcja czasu
- c to pewna stała

$$z = h + (e^{\alpha z} - e^{-\alpha z}) \sin(\alpha(ct - x)) \quad (3)$$

- $\alpha = 2\pi/\lambda$, gdzie λ to długość fali
- c to predkość fali
- h to głębokość

- pierwsza generacja - lata 60'
- druga generacja - lata 70'
- trzecia generacja - modele WAM (tylko podstawowe równanie transportu widmowego) i SWAN

Równanie modelu SWAN

$$\frac{d}{dt} N(\sigma, \theta) + \nabla_{x,y}(c_{x,y} N(\sigma, \theta)) + \frac{d}{d\sigma}(c_\sigma N(\sigma, \theta)) + \frac{d}{d\theta}(c_\theta N(\sigma, \theta)) = \frac{S(\sigma, \theta)}{\sigma}$$

Od lewej jest to suma tempa zmian gęstości siły fali w czasie, zmiana gęstości mocy fali na jednostkę powierzchni w danym punkcie przestrzeni, zmiany częstotliwości fali w wyniku jej propagacji w ośrodku, propagacji w przestrzeni θ . Gdzie c_σ i c_θ to predkości rozchodzenia się fal w odpowiednio σ -space i θ -space. $S(\sigma, \theta)$ - dodatkowa energia w układzie np. dodana przez wiatr, interakcje fal, stłumiona przez tarcie pomiędzy woda i dnem

Bez utraty ogólności, załóżmy, że mamy problem maksymalizacji funkcji celu f . W WWO, przestrzeń rozwiazań X jest analogiczna do obszaru dna morskiego, a fitness punktu x należącego do X jest odwrotnie proporcjonalny do jego głębokości wody: im krótsza odległość do poziomu wody, tym wyższy fitness $f(x)$. Analogicznie trójwymiarowa przestrzeń dna morskiego uogólniamy do przestrzeni n-wymiarowej.

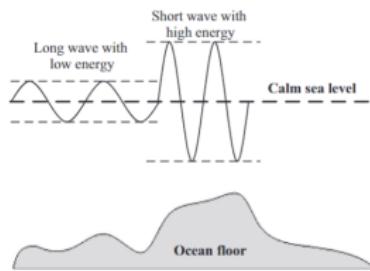
Populacja rozwiazań to fala posiadająca wysokość (lub amplitude), długość.

Podczas procesu rozwiązywania problemu rozważamy: rozchodzenie się (propagacje), refrakcje i załamanie fal.

Propagacja

Dla każdej nowej generacji:

$$x'(d) = x(d) + \text{rand}(-1, 1) \cdot \lambda L(d)$$



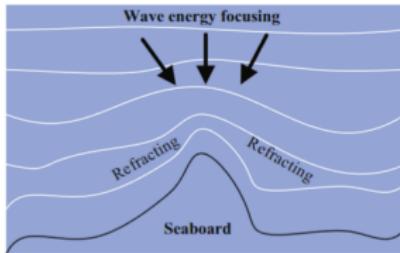
Długość fali aktualizuje się w następujący sposób:

$$\lambda = \lambda \cdot \alpha^{-(f(x) - f_{min} + c)/(f_{max} - f_{min} + c)}$$

Refrakcja

Pozycja po refrakcji

$$x'(d) = N\left(\frac{x^*(d)+x(d)}{2}, \frac{|x^*(d)-x(d)|}{2}\right)$$

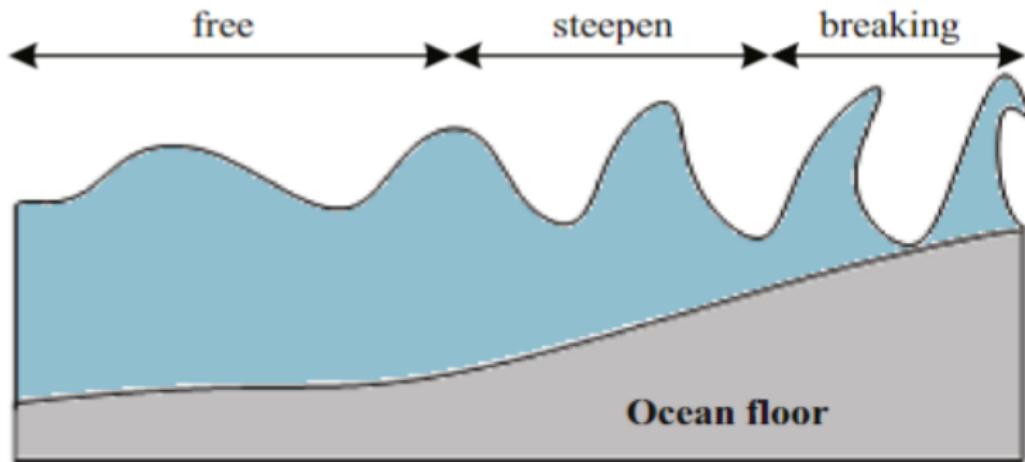


Długość fali aktualizuje się w następujący sposób:

$$\lambda' = \lambda \frac{f(x)}{f(x')}$$

Pozycja po załamaniu

$$x'(d) = x(d) + N(0, 1) \cdot \beta L(d)$$



Algorithm 1. The WWO algorithm.

```
1 Randomly initialize a population  $P$  of  $n$  waves (solutions);  
2 while stop criterion is not satisfied do  
3   for each  $\mathbf{x} \in P$  do  
4     Propagate  $\mathbf{x}$  to a new  $\mathbf{x}'$  based on Eq. (6);  
5     if  $f(\mathbf{x}') > f(\mathbf{x})$  then  
6       if  $f(\mathbf{x}') > f(\mathbf{x}^*)$  then  
7         Break  $\mathbf{x}'$  based on Eq. (10);  
8         Update  $\mathbf{x}^*$  with  $\mathbf{x}'$ ;  
9         Replace  $\mathbf{x}$  with  $\mathbf{x}'$ ;  
10      else  
11        Decrease  $\mathbf{x}.h$  by one;  
12        if  $\mathbf{x}.h = 0$  then  
13          Refract  $\mathbf{x}$  to a new  $\mathbf{x}'$  based on Eq. (8) and (9);  
14          Update the wavelengths based on Eq. (7);  
15      return  $\mathbf{x}^*$ .
```

[1]

Właściwości

- dobrze sobie radzi z małą populacją (5-10 fal)
- im większa jest wysokość fali tym dłuższy jest średni czas życia fali
- mała wysokość powoduje, że fale będą często zastępowane - zwiększy się różnorodność rozwiązań
- duże α powoduje, że algorytm bada duży obszar, a im jest mniejsze, tym dokładniej bada określony teren

Funkcje unimodalne

Comparative results on unimodal benchmark functions.

		IWO	BBO	GSA	HuS	BA	WWO
f_1	max	2.77E+06	8.09E+07	5.31E+07	1.26E+07	5.51E+08	1.17E+06
	min	3.44E+05	5.75E+06	4.56E+06	1.61E+06	1.18E+08	1.44E+05
	median	² 1.42E+06	⁵ 2.14E+07	⁴ 8.37E+06	³ 5.10E+06	⁶ 3.10E+08	¹ 6.26E+05
	std	5.72E+05	1.67E+07	1.32E+07	2.62E+06	1.05E+08	2.45E+05
f_2	max	4.06E+04	8.04E+06	1.61E+04	2.41E+04	6.35E+09	1.48E+03
	min	6.09E+03	1.15E+06	3.47E+03	3.09E+02	1.13E+09	2.00E+02
	median	⁴ 1.52E+04	⁵ 3.95E+06	² 8.38E+03	³ 9.09E+03	⁶ 2.49E+09	¹ 2.68E+02
	std	8.67E+03	1.55E+06	2.90E+03	6.01E+03	7.55E+08	2.02E+02
f_3	max	1.50E+04	5.07E+04	7.58E+04	3.36E+03	1.11E+05	1.32E+03
	min	3.50E+03	5.92E+02	2.04E+04	3.00E+02	3.44E+04	3.15E+02
	median	³ 7.29E+03	⁴ 7.65E+03	⁵ 4.51E+04	¹ 3.02E+02	⁶ 7.19E+04	² 4.87E+02
	std	2.69E+03	1.28E+04	1.04E+04	5.41E+02	1.75E+04	1.85E+02

[1]

Funkcje multimodalne cz.1

Comparative results on multimodal benchmark functions.

		IWO	BBO	GSA	HuS	BA	WWO
f_4	max	5.45E+02	6.54E+02	8.49E+02	5.64E+02	1.26E+04	5.42E+02
	min	4.02E+02	4.23E+02	5.73E+02	4.04E+02	2.01E+03	4.00E+02
	median	³ 5.11E+02	⁴ 5.42E+02	⁵ 6.82E+02	² 5.03E+02	⁶ 3.05E+03	¹ 4.02E+02
	std	2.88E+01	3.84E+01	5.15E+01	3.66E+01	1.97E+03	3.64E+01
f_5	max	5.20E+02	5.20E+02	5.20E+02	5.21E+02	5.21E+02	5.20E+02
	min	5.20E+02	5.20E+02	5.20E+02	5.21E+02	5.21E+02	5.20E+02
	median	³ 5.20E+02	⁴ 5.20E+02	⁵ 5.20E+02	² 5.21E+02	⁶ 5.21E+02	² 5.20E+02
	std	3.77E-03	4.22E-02	6.47E-04	7.83E-02	4.81E-02	6.98E-04
f_6	max	6.05E+02	6.18E+02	6.24E+02	6.29E+02	6.39E+02	6.13E+02
	min	6.00E+02	6.08E+02	6.17E+02	6.19E+02	6.32E+02	6.01E+02
	median	¹ 6.02E+02	³ 6.14E+02	⁴ 6.20E+02	⁵ 6.23E+02	⁶ 6.37E+02	² 6.06E+02
	std	1.12E+00	2.35E+00	1.83E+00	2.18E+00	1.56E+00	2.62E+00
f_7	max	7.00E+02	7.01E+02	7.00E+02	7.00E+02	9.63E+02	7.00E+02
	min	7.00E+02	7.01E+02	7.00E+02	7.00E+02	8.19E+02	7.00E+02
	median	⁴ 7.00E+02	⁵ 7.01E+02	² 7.00E+02	³ 7.00E+02	⁶ 9.12E+02	¹ 7.00E+02
	std	1.21E-02	2.64E-02	9.55E-04	5.56E-02	3.23E-01	6.26E-03
f_8	max	8.75E-02	9.39E+02	8.01E+02	9.75E+02	1.12E+03	8.15E+02
	min	8.27E+02	8.39E+02	8.00E+02	9.10E+02	9.76E+02	8.00E+02
	median	³ 8.43E+02	⁴ 8.79E+02	² 8.00E+02	⁵ 9.40E+02	⁶ 1.07E+03	¹ 8.00E+02
	std	1.01E+01	2.07E+01	2.06E-01	1.27E+01	2.56E+01	2.34E+00
f_9	max	9.78E+02	9.84E+02	1.10E+03	1.09E+03	1.34E+03	9.84E+02
	min	9.30E+02	9.35E+02	1.02E+03	9.59E+02	1.15E+03	9.35E+02
	median	¹ 9.46E+02	² 9.49E+02	⁵ 1.06E+03	⁴ 1.01E+03	⁶ 1.25E+03	³ 9.61E+02
	std	1.14E+01	1.14E+01	1.74E+01	2.60E+01	4.41E+01	1.11E+01
f_{10}	max	3.57E+03	1.00E+03	5.25E+03	3.21E+03	7.45E+03	2.71E+03
	min	1.59E+03	1.00E+03	3.45E+03	1.36E+03	5.26E+03	1.02E+03
	median	⁴ 2.58E+03	¹ 1.00E+03	⁵ 4.37E+03	³ 2.17E+03	⁶ 6.47E+03	² 1.49E+03
	std	3.80E+02	6.80E-01	3.61E+02	4.33E+02	5.19E+02	3.62E+02

[1]

Funkcje multimodalne cz.2

f_{11}	max	3.80E+03	4.51E+03	6.35E+03	4.23E+03	8.75E+03	3.89E+03
	min	1.48E+03	2.12E+03	3.70E+03	2.20E+03	7.20E+03	2.49E+03
	median	¹ 2.92E+03	³ 3.32E+03	⁵ 4.99E+03	² 3.24E+03	⁶ 8.24E+03	⁴ 3.38E+03
	std	4.48E+02	5.12E+02	5.67E+02	4.66E+02	3.62E+02	2.89E+02
f_{12}	max	1.20E+03	1.20E+03	1.20E+03	1.20E+03	1.20E+03	1.20E+03
	min	1.20E+03	1.20E+03	1.20E+03	1.20E+03	1.20E+03	1.20E+03
	median	¹ 1.20E+03					
	std	1.48E-02	5.62E-02	1.00E-03	7.77E-02	3.34E-01	5.61E-02
f_{13}	max	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03
	min	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03
	median	² 1.30E+03	⁵ 1.30E+03	³ 1.30E+03	⁴ 1.30E+03	⁶ 1.30E+03	¹ 1.30E+03
	std	6.50E-02	1.06E-01	6.65E-02	6.50E-02	5.48E-01	6.41E-02
f_{14}	max	1.40E+03	1.40E+03	1.40E+03	1.40E+03	1.50E+03	1.40E+03
	min	1.40E+03	1.40E+03	1.40E+03	1.40E+03	1.44E+03	1.40E+03
	median	² 1.40E+03	⁵ 1.40E+03	⁴ 1.40E+03	³ 1.40E+03	⁶ 1.47E+03	¹ 1.40E+03
	std	1.19E-01	1.99E-01	4.23E-02	4.74E-02	1.39E-01	4.41E-02
f_{15}	max	1.51E+03	1.53E+03	1.51E+03	1.52E+03	5.92E+05	1.50E+03
	min	1.50E+03	1.51E+03	1.50E+03	1.51E+03	1.59E+04	1.50E+03
	median	³ 1.50E+03	⁴ 1.51E+03	² 1.50E+03	⁵ 1.52E+03	⁶ 1.55E+05	¹ 1.50E+03
	std	8.48E-01	4.30E+00	7.30E-01	3.27E+00	1.40E+05	7.75E-01
f_{16}	max	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03
	min	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03
	median	³ 1.61E+03	¹ 1.61E+03	⁶ 1.61E+03	⁴ 1.61E+03	⁵ 1.61E+03	² 1.61E+03
	std	6.14E-01	5.92E-01	3.43E-01	7.25E-01	1.90E-01	4.67E-01

On f_{11} - f_{16} , the values in bold are better than those seemingly same values not in bold, because the digits after the second decimal place are omitted.

[1]

Funkcje hybrydowe

Comparative results on hybrid benchmark functions.

		IWO	BBO	GSA	HuS	BA	WWO
f_{17}	max	3.50E+05	2.31E+07	1.14E+06	1.10E+06	9.90E+06	6.16E+04
	min	5.37E+03	1.26E+06	1.85E+05	1.43E+04	1.45E+06	6.71E+03
	median	² 6.75E+04	⁵ 3.13E+06	⁴ 6.63E+05	³ 1.51E+05	⁶ 4.24E+06	1.26E+04
	std	6.85E+04	4.19E+06	2.20E+05	1.61E+05	1.79E+06	1.24E+04
f_{18}	max	1.80E+04	1.03E+05	4.20E+03	1.09E+04	3.64E+08	2.73E+03
	min	2.26E+03	6.74E+03	2.02E+03	2.02E+03	1.33E+07	1.85E+03
	median	⁴ 4.35E+03	⁵ 2.28E+04	² 2.13E+03	³ 2.73E+03	⁶ 8.54E+07	1.201E+03
	std	3.69E+03	1.97E+04	3.78E+02	2.25E+03	1.00E+08	1.25E+02
f_{19}	max	1.91E+03	1.98E+03	2.00E+03	2.04E+03	2.06E+06	1.91E+03
	min	1.90E+03	1.91E+03	1.91E+03	1.91E+03	1.95E+03	1.90E+03
	median	² 1.91E+03	³ 1.91E+03	⁵ 2.00E+03	⁴ 1.92E+03	⁶ 2.01E+03	1.91E+03
	std	1.65E+00	2.77E+01	3.43E+01	3.31E+01	2.03E+01	1.38E+00
f_{20}	max	5.34E+03	8.62E+04	6.82E+04	6.03E+04	4.44E+04	1.58E+04
	min	2.30E+03	8.64E+03	2.32E+03	2.22E+04	5.40E+03	2.14E+03
	median	¹ 2.74E+03	² 2.72E+04	⁴ 1.77E+04	⁶ 3.68E+04	³ 1.63E+04	2.425E+03
	std	7.00E+02	1.76E+04	1.39E+04	8.49E+03	1.03E+04	3.18E+03
f_{21}	max	9.03E+04	1.67E+06	3.09E+05	1.66E+05	3.34E+06	1.76E+05
	min	6.74E+03	6.70E+04	5.87E+04	1.07E+04	1.43E+05	3.70E+03
	median	² 3.35E+04	⁵ 4.22E+05	⁴ 1.71E+05	³ 4.70E+04	⁶ 9.17E+05	1.292E+04
	std	2.30E+04	3.35E+05	6.53E+04	4.24E+04	7.51E+05	3.50E+04
f_{22}	max	2.52E+03	3.28E+03	3.63E+03	3.67E+03	3.56E+03	2.85E+03
	min	2.23E+03	2.25E+03	2.63E+03	2.37E+03	2.72E+03	2.22E+03
	median	¹ 2.36E+03	³ 2.71E+03	⁶ 3.15E+03	⁴ 3.08E+03	⁵ 3.14E+03	2.48E+03
	std	7.34E+01	2.34E+02	2.50E+02	2.67E+02	2.05E+02	1.43E+02

On f_{19} , the values in bold are better than those seemingly same values not in bold, because the digits after the second decimal place are omitted.

[1]

Funkcje złożone

Comparative results on composition benchmark functions.

		IWO	BBO	GSA	HuS	BA	WWO
f_{23}	max	2.62E+03	2.62E+03	2.65E+03	2.62E+03	2.88E+03	2.62E+03
	min	2.62E+03	2.62E+03	2.50E+03	2.62E+03	2.51E+03	2.62E+03
	median	2.62E+03	2.62E+03	2.56E+03	2.62E+03	2.51E+03	2.62E+03
	std	7.95E-02	1.32E+00	6.45E+01	8.45E-01	1.28E+02	1.45E-01
f_{24}	max	2.63E+03	2.65E+03	2.60E+03	2.71E+03	2.60E+03	2.63E+03
	min	2.60E+03	2.63E+03	2.60E+03	2.63E+03	2.60E+03	2.62E+03
	median	2.62E+03	2.63E+03	2.60E+03	2.66E+03	2.60E+03	2.63E+03
	std	1.08E+01	5.97E+00	1.71E-02	1.25E+01	1.20E+00	6.89E+00
f_{25}	max	2.71E+03	2.72E+03	2.71E+03	2.75E+03	2.76E+03	2.72E+03
	min	2.70E+03	2.71E+03	2.70E+03	2.71E+03	2.70E+03	2.70E+03
	median	2.70E+03	2.71E+03	2.70E+03	2.72E+03	2.70E+03	2.71E+03
	std	8.08E-01	3.01E+00	1.32E+00	6.27E+00	1.50E+01	2.00E+00
f_{26}	max	2.70E+03	2.80E+03	2.80E+03	2.80E+03	2.70E+03	2.70E+03
	min	2.70E+03	2.70E+03	2.80E+03	2.70E+03	2.70E+03	2.70E+03
	median	2.70E+03	2.70E+03	2.80E+03	2.80E+03	2.70E+03	2.70E+03
	std	5.43E-02	2.20E+01	5.43E-03	3.53E+01	5.37E-01	6.50E-02
f_{27}	max	3.10E+03	3.51E+03	4.43E+03	6.47E+03	3.53E+03	3.50E+03
	min	3.01E+03	3.24E+03	3.10E+03	3.57E+03	3.21E+03	3.10E+03
	median	3.10E+03	3.40E+03	3.82E+03	6.84E+03	3.31E+03	3.10E+03
	std	3.38E+01	6.35E+01	3.51E+02	6.83E+02	6.46E+01	5.90E+01
f_{28}	max	3.85E+03	4.27E+03	6.92E+03	6.65E+03	6.10E+03	5.39E+03
	min	3.56E+03	3.61E+03	3.76E+03	4.70E+03	3.01E+03	3.10E+03
	median	3.69E+03	3.79E+03	5.43E+03	5.36E+03	4.52E+03	3.78E+03
	std	4.12E+01	9.33E+01	7.15E+02	4.61E+02	5.93E+02	3.61E+02
f_{29}	max	2.79E+04	8.64E+06	2.93E+06	4.11E+07	1.36E+07	5.06E+03
	min	5.37E+03	4.26E+03	3.10E+03	4.81E+03	6.16E+05	3.56E+03
	median	1.58E+04	5.26E+03	3.10E+03	1.54E+04	4.21E+06	2.402E+03
	std	5.14E+03	1.11E+06	3.78E+05	7.70E+06	2.83E+06	3.60E+02
f_{30}	max	1.69E+04	3.75E+04	1.14E+05	3.74E+04	5.08E+05	7.66E+03
	min	6.05E+03	7.78E+03	1.22E+04	8.27E+03	6.26E+04	4.25E+03
	median	8.85E+03	5.56E+04	1.46E+04	4.51E+04	1.77E+05	5.63E+03
	std	2.08E+03	6.08E+03	1.84E+04	6.58E+03	9.11E+04	7.38E+02

On f_{24} - f_{27} , the values in bold are better than those seemingly same values not in bold, because the digits after the second decimal place are omitted.

[1]

-  Yu-Jun Zheng.
Water wave optimization: A new nature-inspired metaheuristics.

Artificial Immune Systems

Clonal Selection Algorithm

Magdalena Jeczeń

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

Artificial Immune Systems

Algorytmy ewolucyjne oparte na systemie odpornościowym.

Cechy systemu odpornościowego, które sprawiają, że jest ciekawą bazą dla algorytmów:

- Rozpoznawalność
- Pamięć
- Różnorodność
- Zdecentralizowany mechanizm kontroli

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

Przydatne pojęcia

Antygen (eng: Antigen)

Obca, szkodliwa dla organizmu substancja.

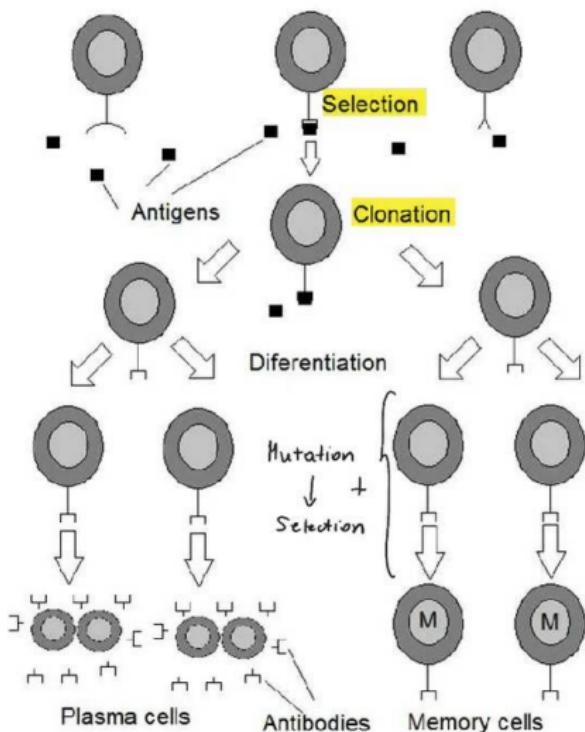
Przeciwciało (eng: Antibody)

Białko przeznaczone do walki z antygenem.

Limfocyt B

Komórka układu odpornościowego odpowiedzialna za wytwarzanie przeciwciał.

Co następuje, gdy w ciele pojawia się抗原 (antigen)?

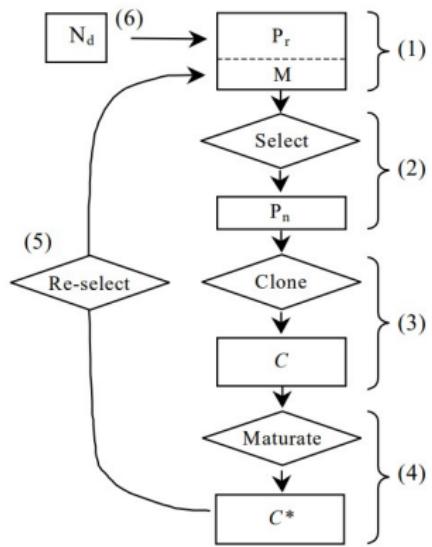


[2]

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

Schemat



- ➊ Wybór kandydatów
- ➋ Wybór n najlepszych kandydatów
- ➌ Klonowanie najlepszych kandydatów
- ➍ Mutacja najlepszych kandydatów
- ➎ Wybór nowych najlepszych kandydatów
- ➏ Zamiana najgorszych kandydatów z początkowego zbioru z nowo-powstałymi najlepszymi

[2]

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

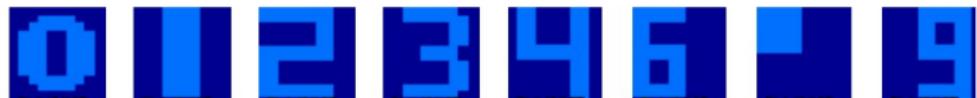
Przykład 1

Optymalizujemy funkcję

$$f : R \rightarrow R, f(x) = x^2$$

- szukamy jej minimum globalnego.

Przykład 2



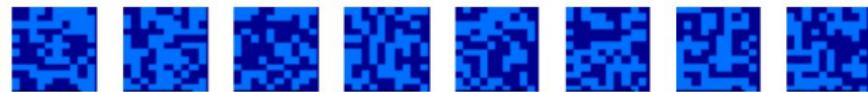
Rozpoznanie 8 'antygenów' reprezentowanych jako bitstring

Sposób w jaki mierzmy powinowactwo:

$$D = \sum_{i=1}^L \delta, \text{ where } \delta = \begin{cases} 1 & \text{if } ab_i \neq ag_i \\ 0 & \text{otherwise} \end{cases}$$

[2]

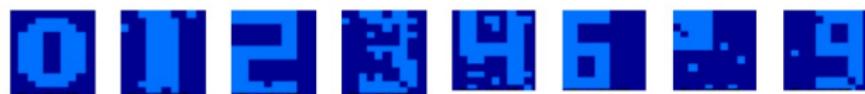
Przykład 2



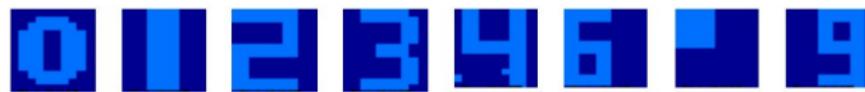
(b) 0 generations



(c) 50 generations



(d) 100 generations



(e) 200 generations

[2]

Przykład 3

2.

$$f(x) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)]$$

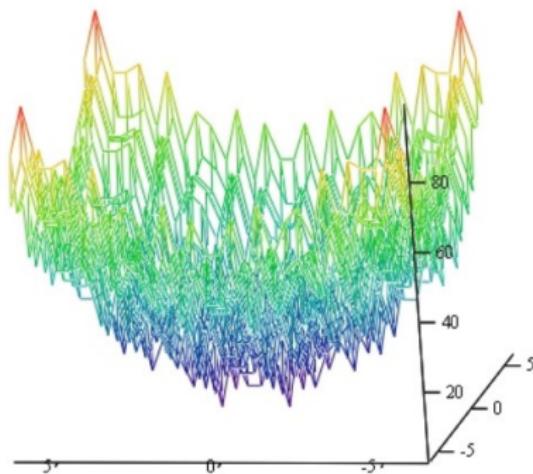


Figure 2: Rastrigin's Function in two dimensions

[3]

Przykład 3

4.

$$f(x) = \sum_{i=1}^n \sin(x_i) + n$$

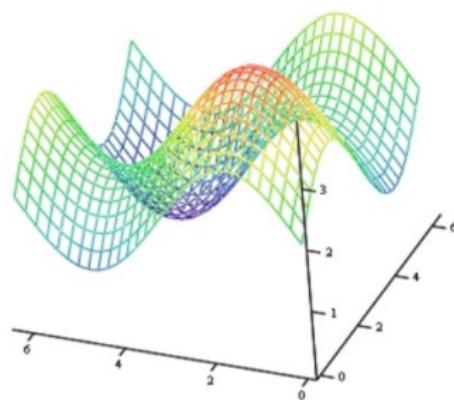


Figure 4. Modified Sinusoidal function in two dimensions

[3]

Przykład 3

Function	Type	Clonal Selection Algorithm			
		Sets for clone rate	Group number for mut. rate	Proximity	Avg. Number of Iterations
Sphere	Unimodal	2	1	6.95×10^{-7}	399
Rastrigin's	Highly Multimodal	3	3	8.51×10^{-8}	135226
Ackley's	Multimodal	2	1	8.59×10^{-4}	417
Modified Sinusoidal	Highly Multimodal	3	2	9.71×10^{-4}	14488
Sum of Different Powers	Unimodal	3	1	6.21×10^{-6}	53
Non-generalized Schwefel's	Multimodal	1	1	8.93×10^{-4}	206

[3]

Plan prezentacji

- 1 Artificial immune systems - wprowadzenie
- 2 Selekcja klonalna
- 3 Clonal Selection Algorithm - schemat
- 4 Przykłady
- 5 Inne rodzaje algorytmów opartych na selekcji klonalnej

Algorytmy oparte na selekcji klonalnej mogą rozwiązywać nie tylko problemy optymalizacyjne, czy rozpoznawania patternów, ale także klasyfikacji.

Jednym z takich algorytmów jest algorytm **CLONAX**

CLONAX

CLONAl selection Algorithm for ClaSSification.

CLONAX

- ① Wybór zbioru kandydatów
- ② Wpuszczenie antygenu A_i do zbioru
- ③ Sprawdzenie do jakiej klasy należy抗原
- ④ Wyodrębnianie ze zbioru kandydatów tylko tych, którzy mają większe powinowactwo ('przyciąganie') do klasy jakiej jest antygen
- ⑤ Klonowanie najlepszych kandydatów
- ⑥ Mutacja najlepszych kandydatów
- ⑦ Ponowny wybór najlepszych kandydatów
- ⑧ Zamiana powinowactwa tych kandydatów na średnie powinowactwo do najbliższych antygenów z tej samej klasy
- ⑨ Filtrowanie
- ⑩ Wprowadzenie do pierwotnego zbioru

CLONAX

Dataset	Gen	Memory cell size (m)	Remaining cells (r)	Replaceable antibody size (d)	best (n) antibodies picked for cloning	Best (k) picked to make clones for candidate memory cell	Max antigens per memory cell (p)	Average results of 5 test runs (%)	Individual best (%)
Breast Cancer Diagnostic	8	120	0.1xm	0.5xr	20	10	5	93.4±2.2	95.6
Haberman's Survival	8	70	0.1Xm	0.5Xr	20	10	5	73.8±6.6	80.3
Liver Disorder	16	60	0.1xm	0	30	10	8	68.1±5.8	73.9
New-Thyroid	8	60	0.1xm	0.5xr	10	10	6	90.7±7.0	97.7
Pima-Indian-Diabetes	8	100	0.1xm	0.5xr	20	10	8,10	74.4±5.6	79.9

[4]

Bibliografia

- ① **Tytuł:** "The Clonal Selection Algorithm with Engineering Applications",
Autorzy: Leandro Nunes de Castro, Fernando J. Von Zuben
- ② https://www.researchgate.net/figure/Clonal-selection-theory_fig1_335210140
- ③ **Tytuł:** "COMPARISON STUDY FOR CLONAL SELECTION ALGORITHM AND GENETIC ALGORITHM"
Autorzy: Ezgi Deniz Ülker, Sadık Ülker
- ④ **Tytuł:** "Clonal Selection Algorithm for Classification"
Autorzy: Anurag Sharma, Dharmendra Sharma
https://link.springer.com/chapter/10.1007/978-3-642-22371-6_31

KONIEC

Dziękuję za uwagę : -)

Shinybrms - rozszerzenie shiny do uprawiania statystyki Bayesowskiej

Marta Szuwarska



[1]

Wprowadzenie do statystyki Bayesowskiej

czyli co to za twór?

Przykład 1



Obrazek 2, [2]

Przykład 1



Obrazek 3, [3]

Przykład 2

Kwiecień 2019



Styczeń 2021



Obrazek 4, [4]

Przykład 2



Obrazek 5, [5]

Przykład 3



Obrazek 6, [6]

Twierdzenie Bayesa

Niech $A, B \in \mathcal{F}$ będą zdarzeniami oraz $\mathbb{P}(B) > 0$.

Wtedy:

$$\mathbb{P}(A | B) = \frac{\mathbb{P}(B | A)\mathbb{P}(A)}{\mathbb{P}(B)}$$

Twierdzenie Bayesa

Niech θ będzie wektorem parametrów.

Wtedy:

$$\mathbb{P}(\theta | dane) = \frac{\mathbb{P}(dane | \theta)\mathbb{P}(\theta)}{\mathbb{P}(dane)}$$

Twierdzenie Bayesa

Niech θ będzie wektorem parametrów.

Wtedy:

$$\mathbb{P}(\theta | dane) = \frac{\mathbb{P}(dane | \theta)\mathbb{P}(\theta)}{\mathbb{P}(dane)}$$

Wiarygodność

Prawdopodobieństwo
a priori

Prawdopodobieństwo
a posteriori

Stała normalizująca

Twierdzenie Bayesa

Niech θ będzie wektorem parametrów.

Wtedy:

$$\mathbb{P}(\theta | dane) = \frac{\mathbb{P}(dane | \theta)\mathbb{P}(\theta)}{\mathbb{P}(dane)}$$

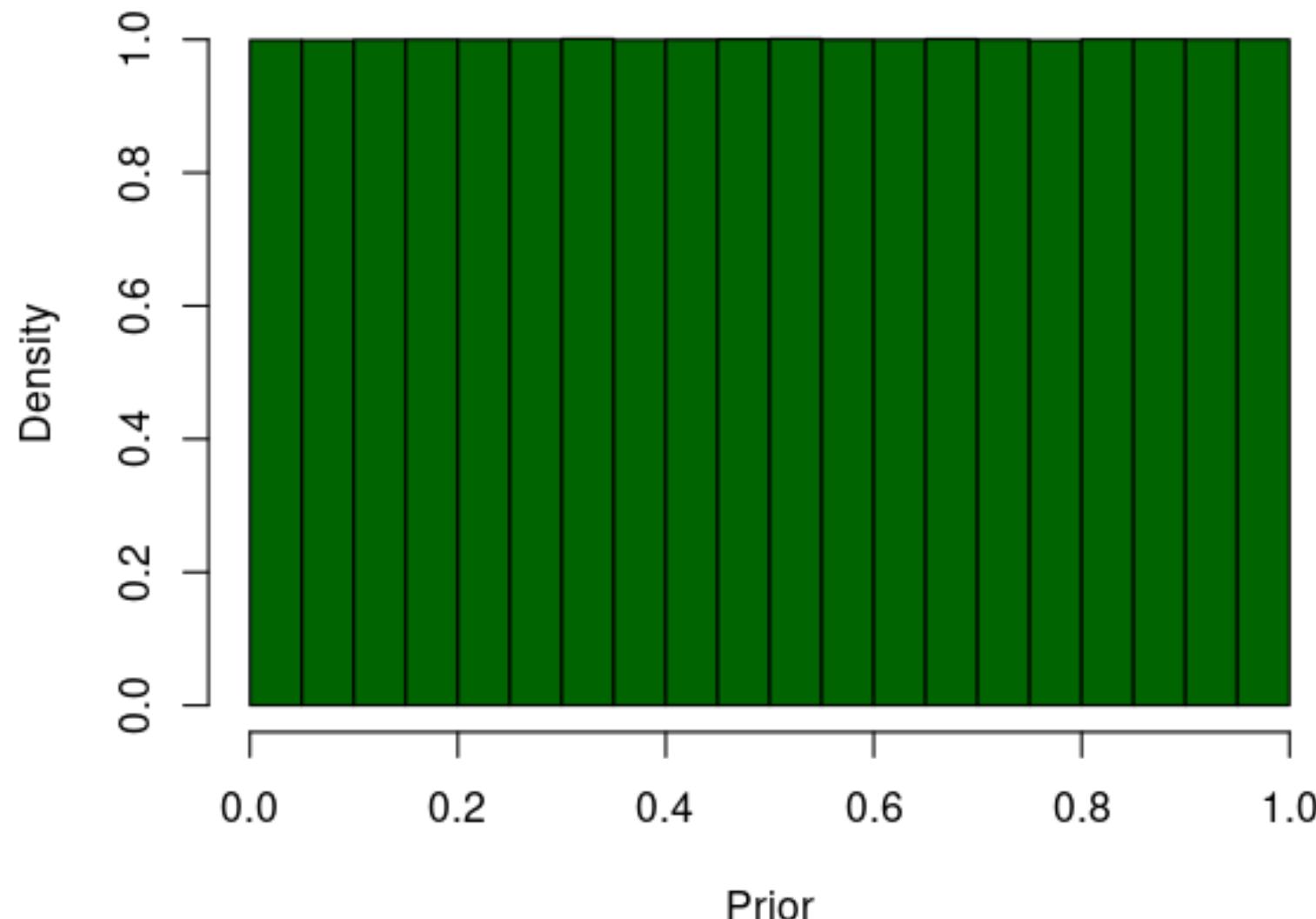
$$\mathbb{P}(\theta | dane) = \frac{\mathbb{P}(dane | \theta)\mathbb{P}(\theta)}{\int_{\Theta} \mathbb{P}(dane | \theta)\mathbb{P}(\theta)d\theta}$$

A priori i a posteriori na przykładzie



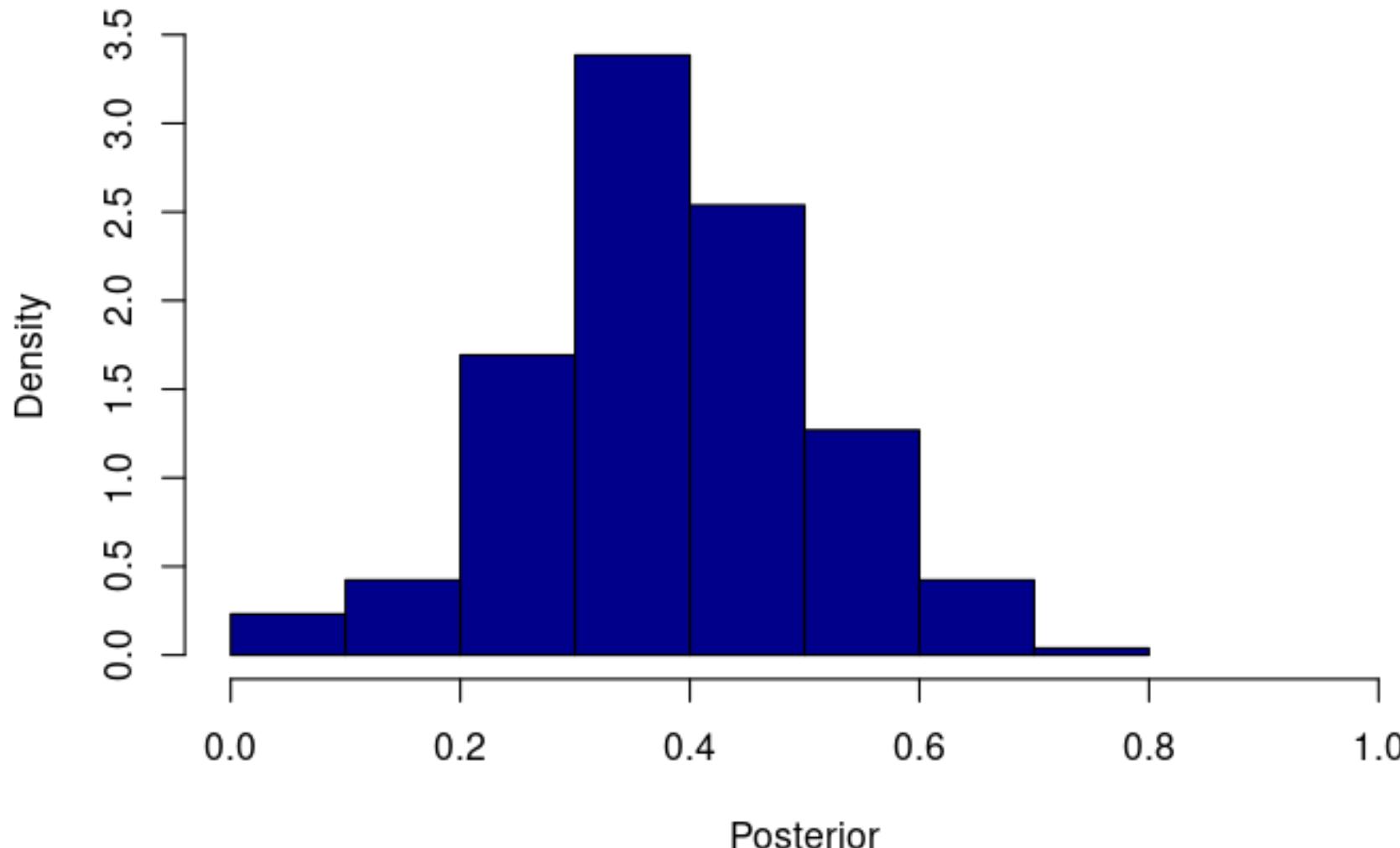
Obrazek 7, [7]

Histogram of Prior



Wykres 1, wygenerowany w R

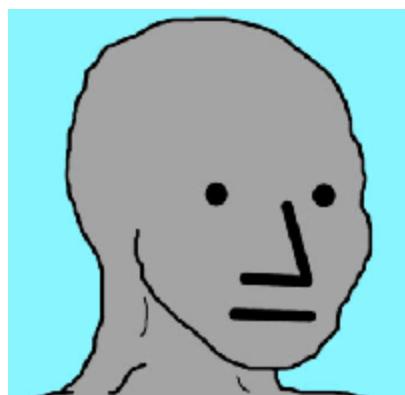
Histogram of Posterior



Wykres 2, wygenerowany w R

Podejście Bayesowskie a klasyczne

Podejście klasyczne	Podejście Bayesowskie
Parametry ustalone	Parametry zmienne
Dane zmienne	Dane ustalone



Obrazek 8, [8]



Obrazek 9, [9]

Zalety podejścia Bayesa

czyli dlaczego warto zostać bayesistą?

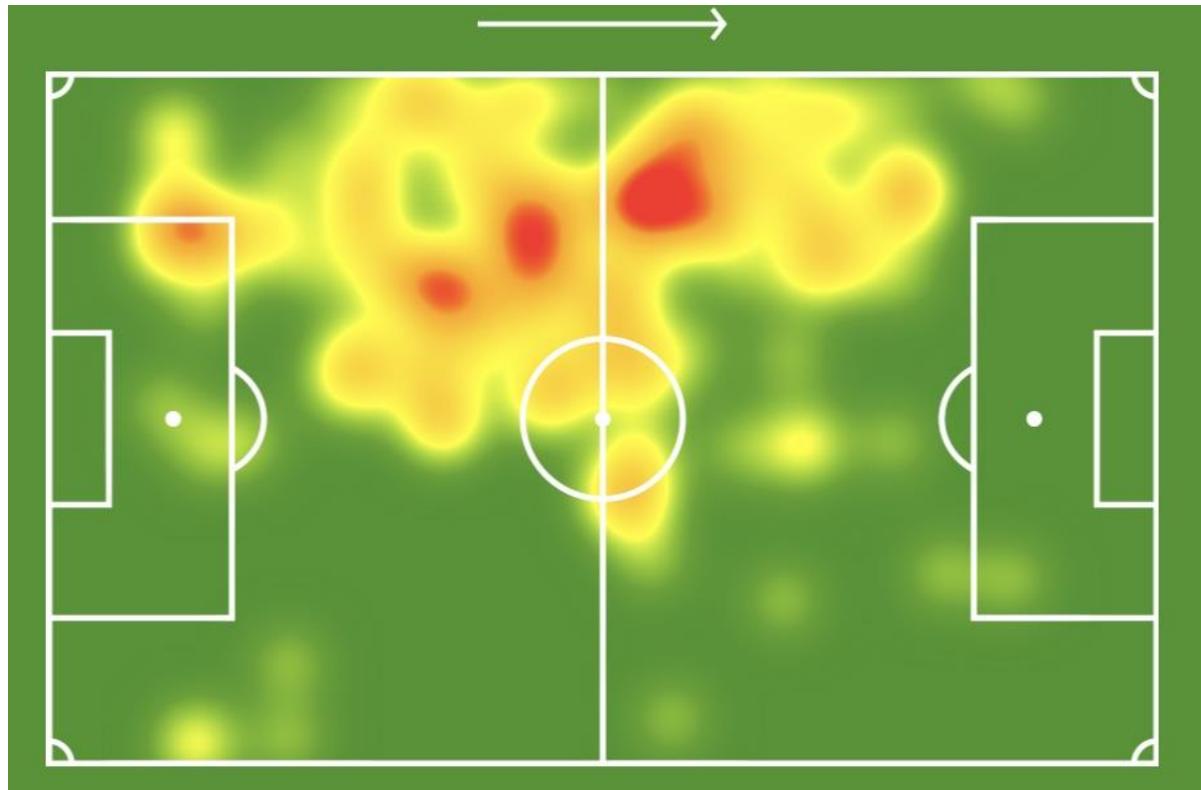
- Możliwość uwzględnienia wiedzy eksperckiej
- Możliwość łatwego narzucenia ograniczeń parametrów
- Możliwość wyliczenia pełnego rozkładu nawet przy mało licznym zbiorze
- Bardziej intuicyjna interpretacja wyników
- Łatwe wykluczenie parametrów zakłócających (*ang. nuisance parameters*)
- Rzadziej dochodzi do testowania istotności hipotezy zerowej
- Podejście bayesowskie nigdy nie jest gorsze niż klasyczne
- Można łatwo przeprowadzić diagnostykę modelu

Stosowane algorytmy

czyli gdzie tu optymalizacja?

Algorytmy MCMC

- Metropolis
- Metropolis-Hastings
- Próbnik Gibbsa
- Hamiltonian Monte Carlo
- no-U-turn sampler



Obrazek 10, [10]

NUTS

- eksploruje rozkład w każdą stronę,
- nie musi wracać,
- jest bardziej efektywny niż tradycyjne algorytmy,
- automatycznie określa długość i kierunek symulowanych trajektorii.



Ocbrazek 11, [11]

Powody powstania pakietu shinybrms

czyli czy naprawdę potrzebujemy kolejnego pakietu do R?

Starsze pakiety

GUI name	Commercial	Analytic	Algorithm (for inferring the posterior)			MCMC			Algorithm choice
			Non-MCMC	Numerical	MC	BB	Non-HMC	Static HMC	
WinBUGS (Lunn et al., 2000)	no	no	no	no	no	yes	no	no	no
OpenBUGS (Spiegelhalter et al., 2014)	no	no	no	no	no	yes	yes	no	no
BugsXLA (Woodward, 2011)	no	no	no	no	no	yes	yes	no	no
IBM SPSS Amos (Arbuckle, 2020)	yes	no	no	no	no	yes	yes	no	yes
TEET (Qian, 2011)	no ⁽ⁱ⁾	yes	yes	yes	no	yes	no	no	no
JASP (JASP Team, 2022)	no	yes	yes	yes	no	yes	no	yes ⁽ⁱⁱ⁾	no
BRNPM (Karabatsos, 2015, 2017)	no	no	no	no	no	yes	no	no	no
Stata (StataCorp, 2019b)	yes	no	no	no	no	yes	no	no	yes
BayES (Emvalomatis, 2020)	no	no	no	no	no	yes	no	no	no
IBM SPSS (IBM Corp., 2020)	yes	yes	yes	yes	no	no	no	no	no
BEsmarter (BEsmarter Team, 2020a,b; Ramírez-Hassan and Graciano-Londoño, 2021)	no	no	no	no	yes	yes	no	no	yes ⁽ⁱⁱⁱ⁾

Tabela 1, [12]

Testujemy apkę na przykładzie

czyli czy opatrunek z plazmy działa?

Podsumowanie

czyli czy warto zainstalować pakiet shinybrms?

Bibliografia

- [1] <https://fweber144.github.io/shinybrms/>
- [2] <https://wiadomosci.onet.pl/pogoda/pogoda-na-czwartek-i-piatek-rajd-burz-przez-polske-gdzie-bedzie-burza/96wld2q>
- [3] https://pl.m.wikipedia.org/wiki/Plik:Window_view_with_green_trees_1.jpg
- [4] <https://www.facebook.com/groups/altminiawa/permalink/2548697745424796/>
- [5] <https://www.facebook.com/wrsminipw/posts/pfbid0zRc2SyZk32xogEGRxHwFnty3GTwGV7RqthGrWvkWqBkAyuYHG32poXrr7kgqSZRdl>
- [6] <https://lublin.wyborcza.pl/lublin/7.48724.25433261,historia-enigmy-pelna-zniewalaczen-i-metnych-wspomnien-ten.html?disableRedirects=true>
- [7] https://www.facebook.com/WMINIPW/photos/a.2336388699931079/2336393559930593/?paipy=0&eav=AfaOYMRNRm3bo522IlzrcO1DqSPAXL6zqrV1R69FwQMIT3_HaeLWtFRv4OvQICP5GQg&_rdr
- [8] https://en.wikipedia.org/wiki/NPC_%28meme%29
- [9] https://en.wikipedia.org/wiki/Thomas_Bayes
- [10] <https://icdn.psgtalk.com/wp-content/uploads/2019/10/VerrattiHeatMap.jpg>
- [11] <https://www.pngall.com/nuts-png/download/48644>
- [12] <https://journal.r-project.org/articles/RJ-2022-027/>
- [13] <https://chat.openai.com>
- [14] https://www.youtube.com/watch?v=3OJEae7Qb_o&ab_channel=rasmusab
- [15] <https://campus.datacamp.com/courses/bayesian-regression-modeling-with-rstanarm/introduction-to-bayesian-linear-models?ex=8>
- [16] https://www.youtube.com/watch?v=QqwCqPYhatA&ab_channel=SaltLakeCityRUsersGroup

Dziękuję za uwagę

No Free Lunch Theorems for Optimization

Piotr Kosakowski

Politechnika Warszawska

27.04.2023

Etymologia

"There's no such thing as a free lunch"

Wprowadzenie

No free lunch theorems to twierdzenia, które mówią o tym, że dla każdego algorytmu, lepsza wydajność w jednej klasie problemów jest równoważona przez wydajność w innej klasie.

Sformułowanie problemu

Założenia

Ograniczamy uwagę do optymalizacji kombinatorycznej, w której przestrzeń poszukiwań X , choć być może dość duża, jest skończona. Zakładamy ponadto, że przestrzeń możliwych wartości "kosztów" Y jest również skończona. Te ograniczenia są automatycznie spełnione dla algorytmów optymalizacyjnych działających na komputerach cyfrowych, gdzie zazwyczaj Y jest 32 lub 64 bitową reprezentacją liczb rzeczywistych.

Funkcja kosztu

Problem optymalizacyjny, zwany również funkcją kosztu, f jest reprezentowany jako przekształcenie $f : X \rightarrow Y$, a $F = Y^X$ oznacza przestrzeń możliwych problemów.

Próbka

Próbką rozmiaru m nazywamy czasowo uporządkowany zbiór m odrębnych odwiedzonych punktów i oznaczamy
 $d_m \equiv \{(d_m^x(1), d_m^y(1)), \dots, (d_m^x(m), d_m^y(m))\}$, gdzie $d_m^x(i)$ oznacza wartość z X , zaś $d_m^y(i)$ to odpowiadający jej koszt z Y .

Przestrzeń próbek

Przestrzeń próbek rozmiaru m to $D_m = (X \times Y)^m$, $D \equiv \bigcup_{m \geq 0} D_m$.

Algorytm

Algorytm a jest reprezentowany jako przekształcenie
 $a : d \in D \rightarrow \{x | x \notin d^x\}$

Miara wydajności

Miarą wydajności ozaczona jest przez $\Phi(d_m^y)$.

Na przykład szukając minimum $f \Phi(d_m^y)$ może przyjąć formę

$$\Phi(d_m^y) = \min_i \{d_m^y(i) : i = 1, \dots, m\}$$

Teoria prawdopodobieństwa

Używana jest teoria prawdopodobieństwa z trzech powodów:

- ① pozwala na łatwą generalizację na algorytmy stochastyczne
- ② zapewnia proste, spójne ramy, w których można przeprowadzić dowody
- ③ kluczowym elementem jest rozkład $P(f) = P(f(x_1), \dots, f(x_{|X|}))$. Rozkład ten, zdefiniowany na F , daje prawdopodobieństwo, że dany $f \in F$ jest rzeczywistym problemem optymalizacyjnym.

Używając prawdopodobieństwa wydajność algorytmu a po m iteracjach na funkcji kosztu f jest mierzona przy pomocy $P(d_m^y | f, m, a)$.

Twierdzenie pierwsze

Dla każdej pary algorytmów a_1 i a_2

$$\sum_f P(d_m^y | f, m, a_1) = \sum_f P(d_m^y | f, m, a_2)$$

Funkcja kosztu zależna od czasu

Podobny fakt zachodzi dla klasy czasowo zależnych funkcji kosztu. Rozważamy początkową funkcję f_1 . Przed rozpoczęciem każdej kolejnej iteracji funkcja kosztu jest przekształcana do nowej funkcji zgodnie z odwzorowaniem $T : F \times \mathbb{N} \rightarrow F$. Dane przekształcenie zakładamy, że jest bijekcją i oznaczamy T_i , więc funkcja w i -tej iteracji to $f_{i+1} = T_i(f_i)$.

Możliwe są dwa schematy:

$$d_m^y = \{f_1(d_m^x(1)), \dots, T_{m-1}(f_{m-1})(d_m^x(m))\}$$

$$D_m^y = \{f_m(d_m^x(1)), \dots, f_m(d_m^x(m))\}$$

Twierdzenie drugie

Dla każdych $d_m^y, D_m^y, m > 1$, algorytmów a_1, a_2 i każdej funkcji początkowej f_1

$$\sum_T P(d_m^y | f_1, T, m, a_1) = \sum_T P(d_m^y | f_1, T, m, a_2)$$

oraz

$$\sum_T P(D_m^y | f_1, T, m, a_1) = \sum_T P(D_m^y | f_1, T, m, a_2)$$

Wniosek

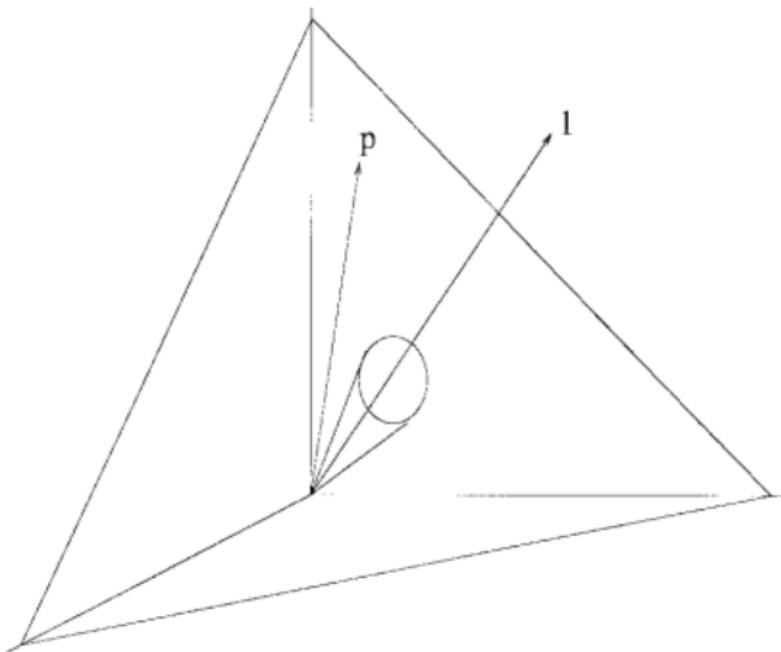
Intuicyjnie, NFL ilustruje, że jeśli wiedza o f , może dana przez $P(f)$, nie jest uwzględniona w a , to nie ma można mieć pewności, że a będzie efektywny. Wówczas efektywna optymalizacja polega na fortunnym zgraniu się f i a .

Wniosek ten jest formalnie uzasadniony przy pomocy geometrycznej reprezentacji.

Reprezentacja geometryczna

Wiemy, że $P(d_m^y|m, a) = \sum_f P(d_m^y|m, a, f)P(f)$. Ową sumę można interpretować jako iloczyn skalarny w F . Oznaczmy wektory $\vec{v}_{d_m^y, a, m}(f) \equiv P(d_m^y|m, a, f)$ i $\vec{p}(f) \equiv P(f)$. Wówczas $P(d_m^y|m, a) = \vec{v}_{d_m^y, a, m} \cdot \vec{p}$

Reprezentacja geometryczna



Rysunek: [1]

Rozważmy teraz jedynie histogram $\vec{c} = (c_{Y_1}, c_{Y_2}, \dots, c_{Y_{|Y|}})$, gdzie c_{Y_i} jest liczbą wystąpień Y_i w d_m^y .

Twierdzenie trzecie

Dla dowolnego algorytmu frakcja funkcji kosztu, które skutkują danym histogramem $\vec{c} = m\vec{\alpha}$ to

$$\rho_f(\vec{\alpha}) = \frac{\binom{m}{c_1 c_2 \dots c_{|Y|}} |Y|^{|X|-m}}{|Y|^{|X|}} = \frac{\binom{m}{c_1 c_2 \dots c_{|Y|}}}{|Y|^m}$$

Rozważmy histogram $\vec{\beta}$, taki że $N_i = \beta_i |X|$ to liczba punktów w X , dla których $f(x) = Y_i$

Twierdzenie czwarte

Dla danej funkcji f z histogramem $\vec{N} = |X|\vec{\beta}$ frakcja algorytmów dających histogram $\vec{c} = m\vec{\alpha}$ jest równa

$$\rho_{alg}(\vec{\alpha}, \vec{\beta}) = \frac{\prod_{i=1}^{|Y|} \binom{N_i}{c_i}}{\binom{|X|}{m}}$$

- ① Nie istnieje uniwersalny algorytm
- ② Wstępna wiedza lub założenia o problemie może pozwolić na dobranie odpowiedniego algorytmu
- ③ Wykonywanie benchmarków jest kluczowe w szukaniu rozwiązania dla danej klasy problemu
- ④ Zrozumienie problemu jest nawet ważniejsze
- ⑤ Wielokrotne optymalizowanie z pozoru podobnych problemów może wymagać wielokrotnego szukania algorytmu

Bibliografia



[1]

David H. Wolpert and William G. Macready (1997)

No free lunch theorems for optimization

IEEE Transactions on Evolutionary Computation 1(1), 67–82.

Protein structure prediction

AlphaFold

O czym będzie prezentacja?

1. Białko we wszechświecie

2. Dlaczego prowadzone są badania nad białkiem

3. Powstanie AlphaFold

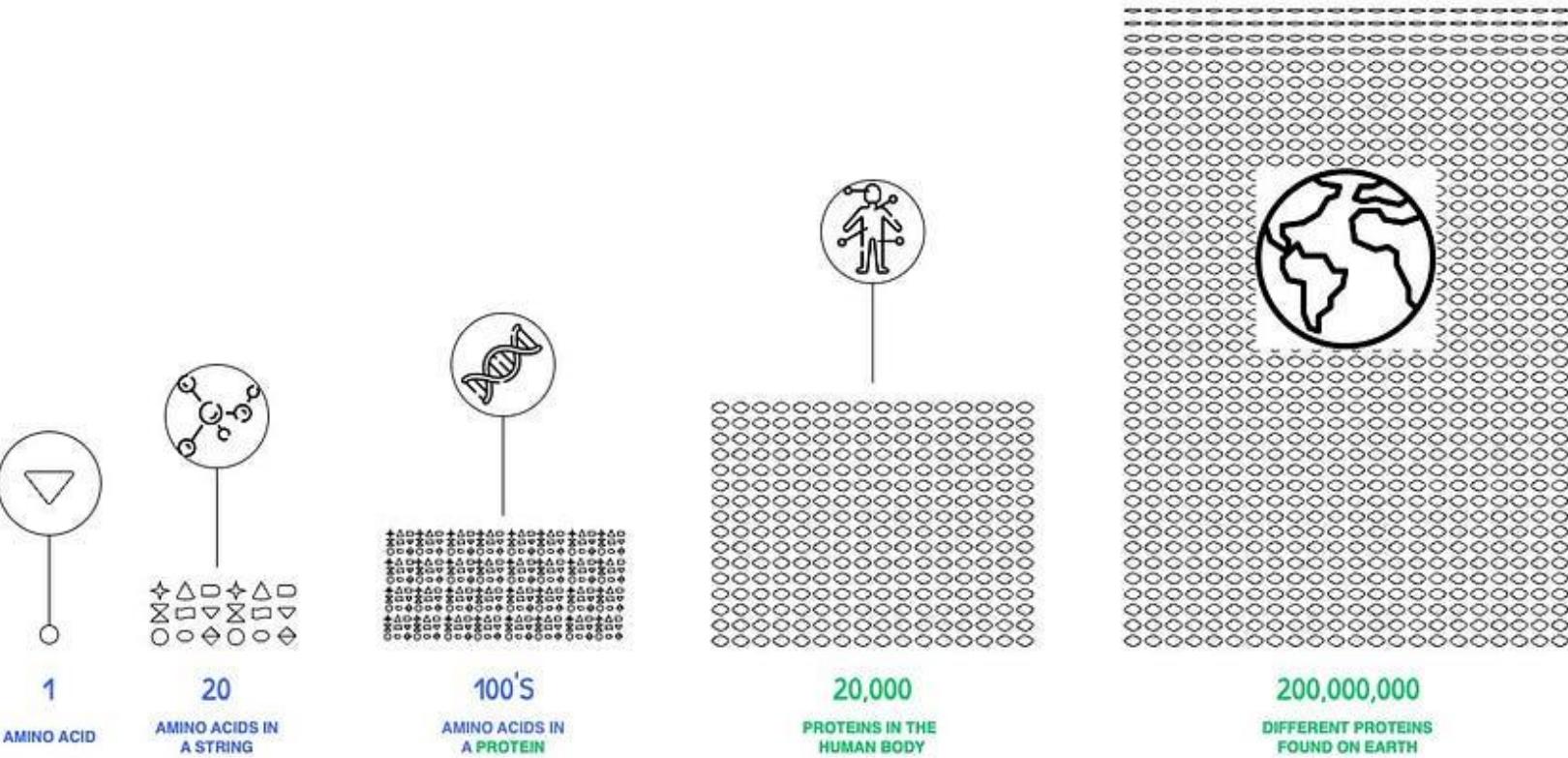
4. Podstawy budowy białka

5. Proces działania AlphaFold

6. Wykorzystane w algorytmach mechanizmy

7. Dostęp do AlphaFold

Co nieco o tym, czemu białka są ważne...





...serio są ważne

Badania nad białkiem

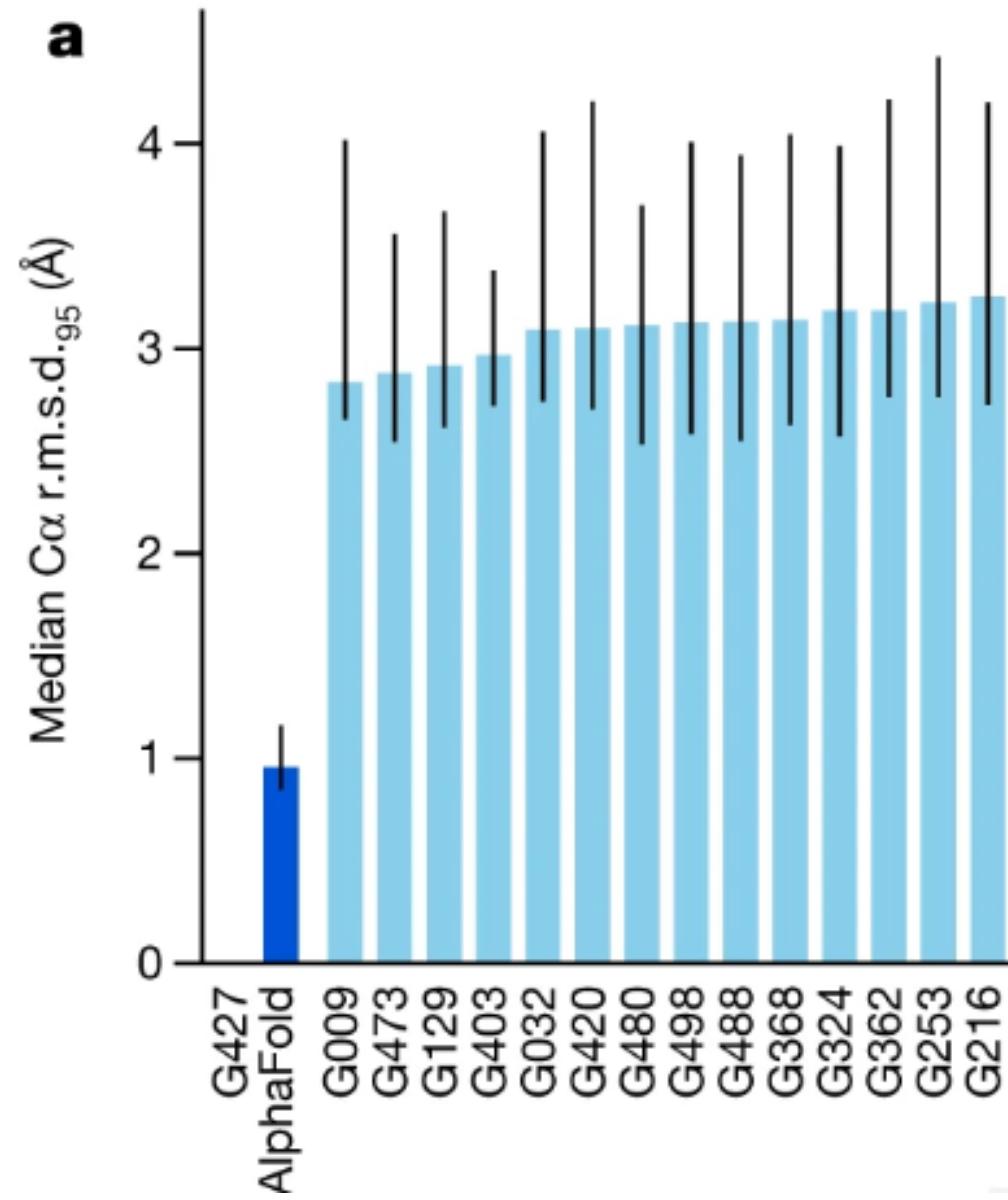


PODEJŚCIE FIZYCZNE



PODEJŚCIE EWOLUCYJNE

Zaprezentowanie AlphaFold



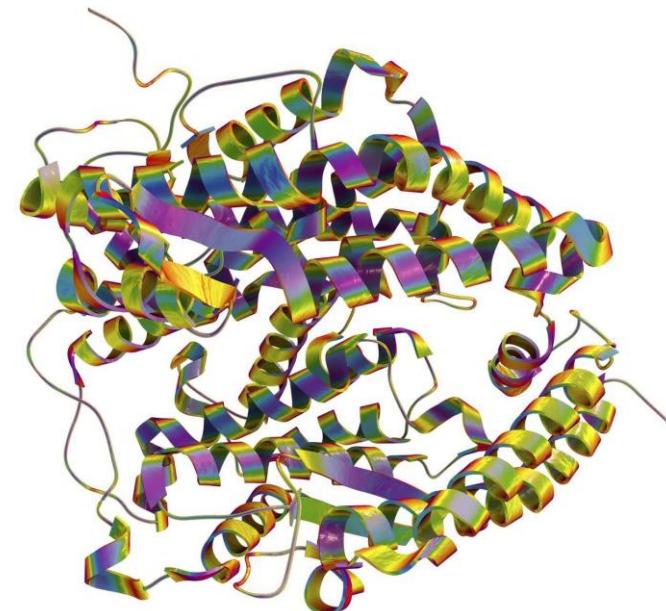
Rewolucja w badaniach nad białkiem

„To zmieni medycynę. Zmieni badania naukowe. Zmieni bioinżynierię.
Zmieni wszystko”.

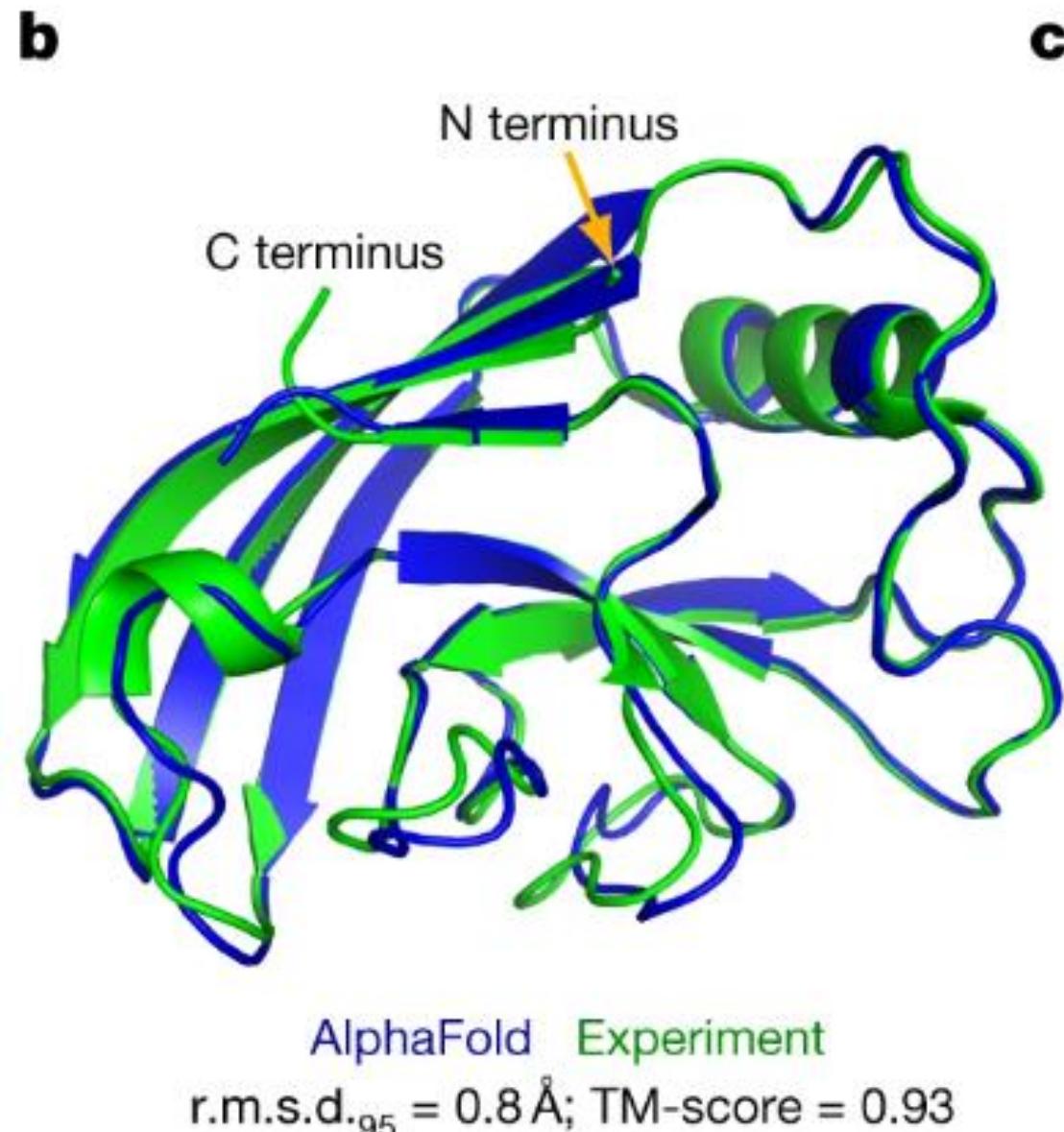
Andrei Lupas, Instytut Biologii Rozwojowej Maxa Plancka w Tybindze

Jak wygląda białko

- Główny budulec to szeregi aminokwasów

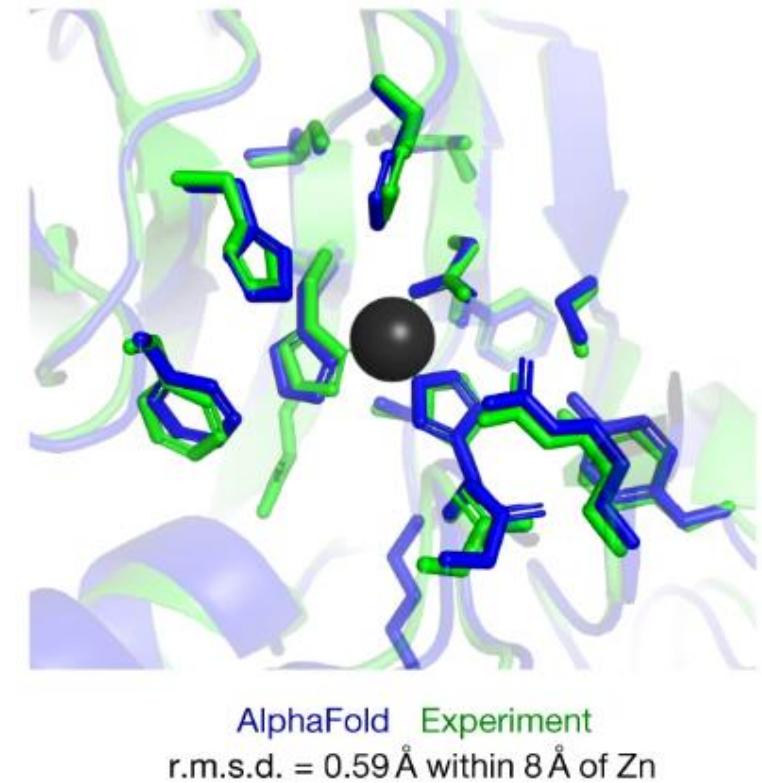


Szkielet białka



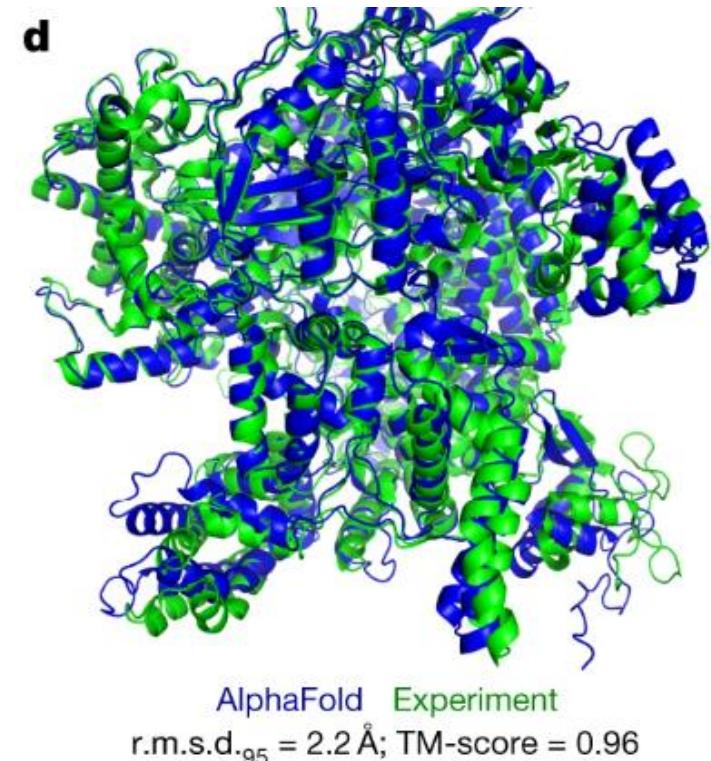
Łańcuchy boczne białka

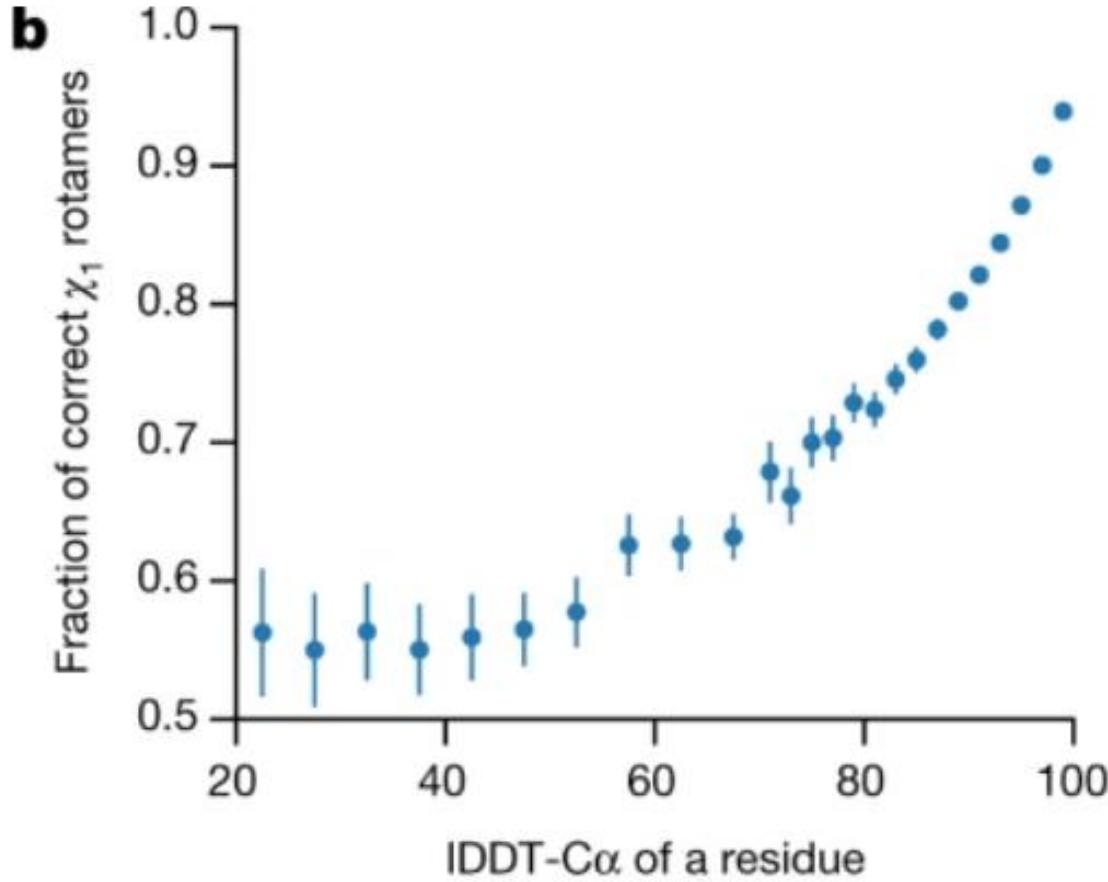
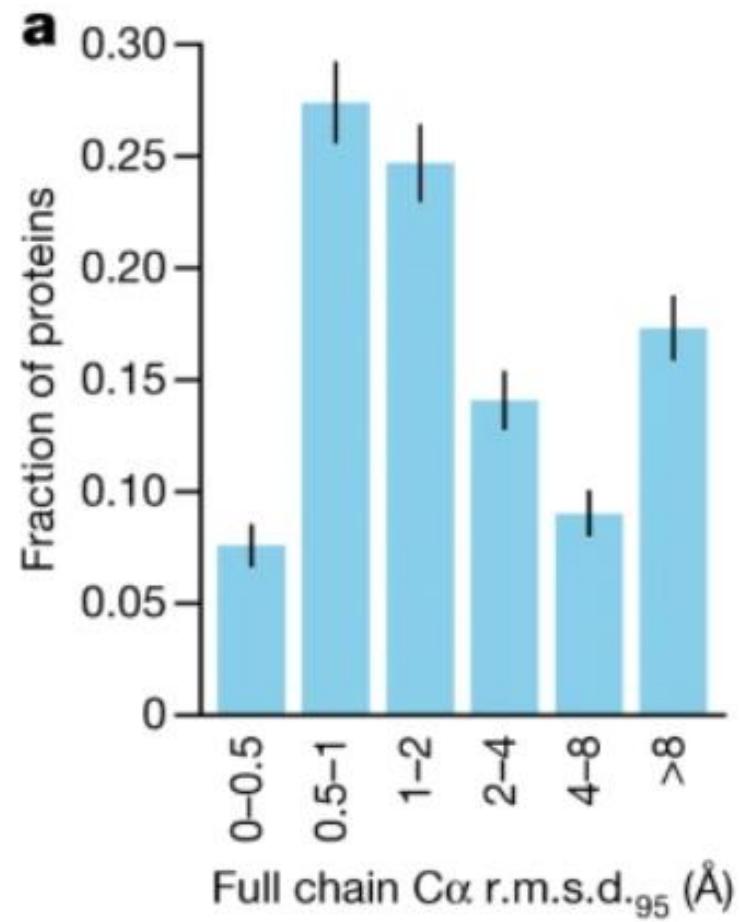
- Część białka wychodząca ze szkieletu
- Dobre przybliżenie, głównie jeśli szkielet jest poprawnie przybliżony



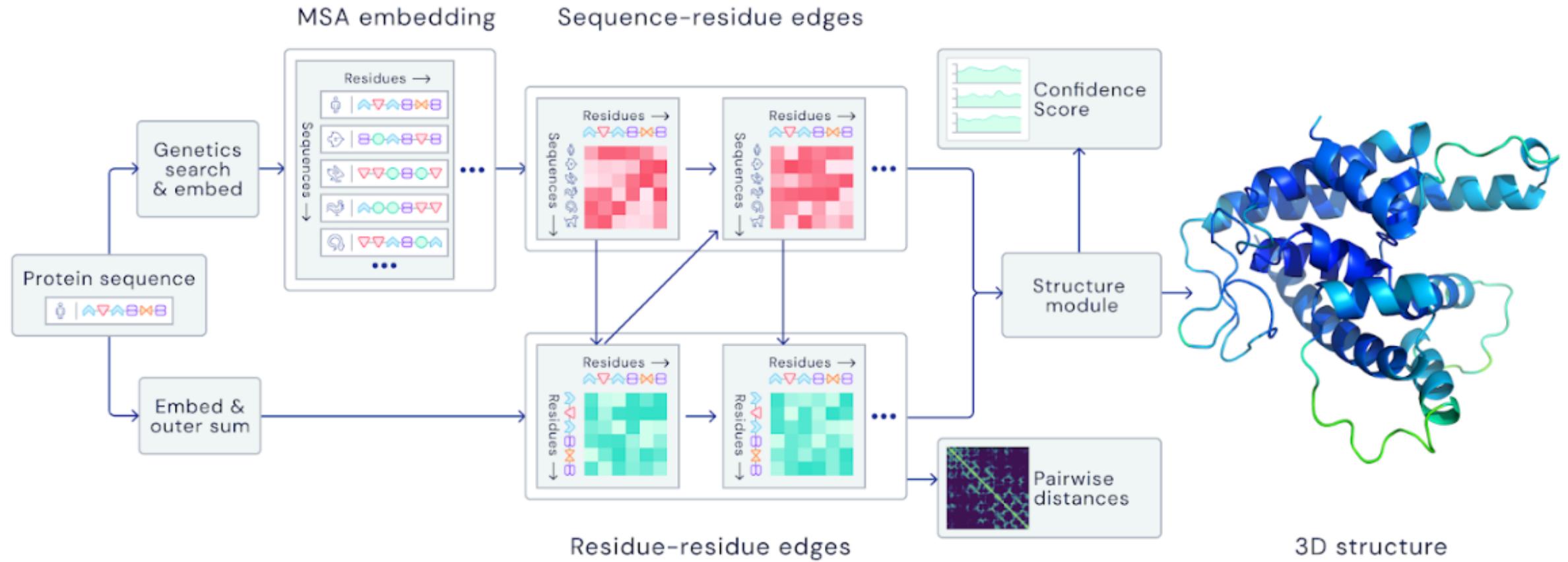
A tak wygląda całe białko

- Szkielety wraz ze swoimi łańcuchami bocznymi skręcają się i tworzą całą strukturę

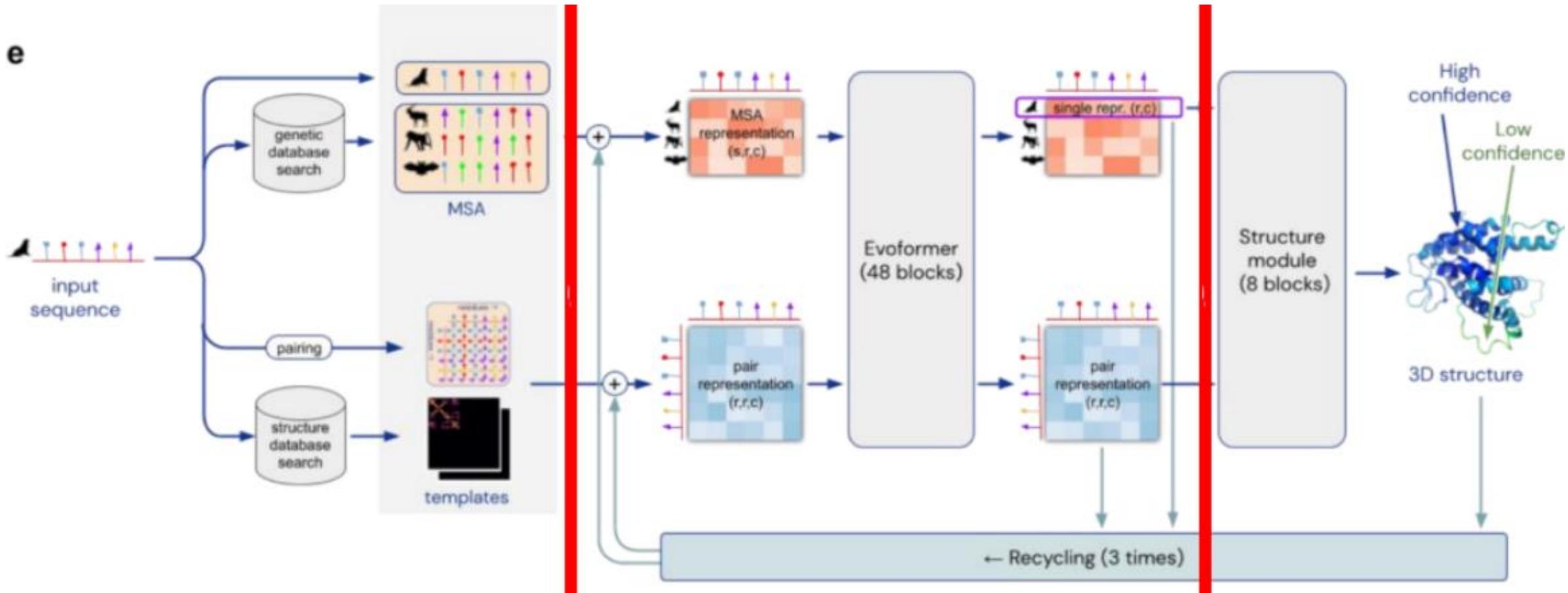




Przybliżanie struktur białka – trochę
wykresów



Pierwsze spojrzenie na AlphaFold

e

Schemat działania

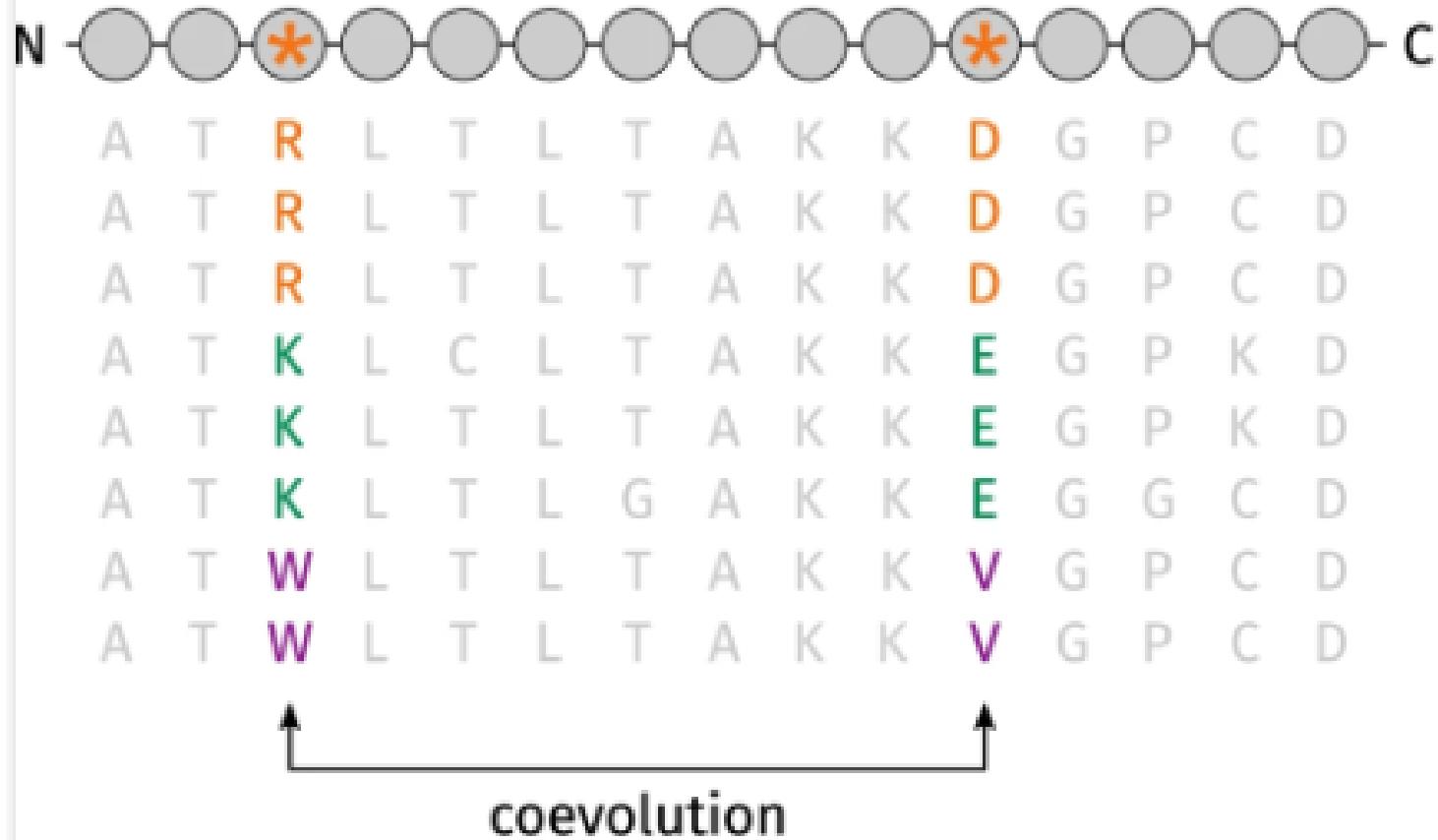
Preprocessing danych

1. Potok przetwarzania wstępnego

2. Przeszukanie dostępnych baz danych

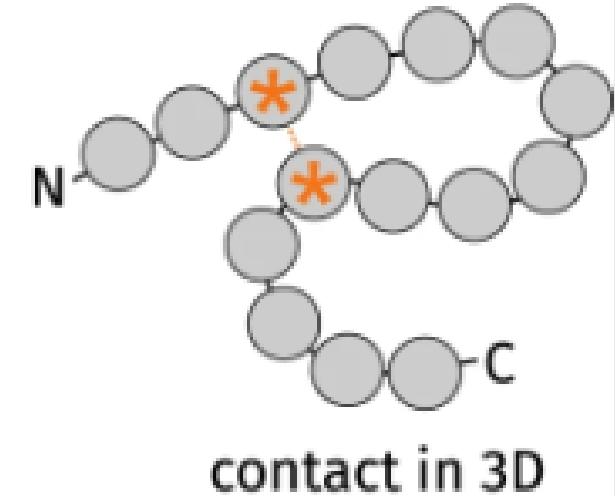
3. Utworzenie MSA

4. Utworzenie reprezentacji par reszt



inference

constraint

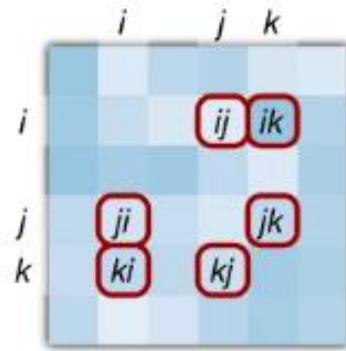


contact in 3D

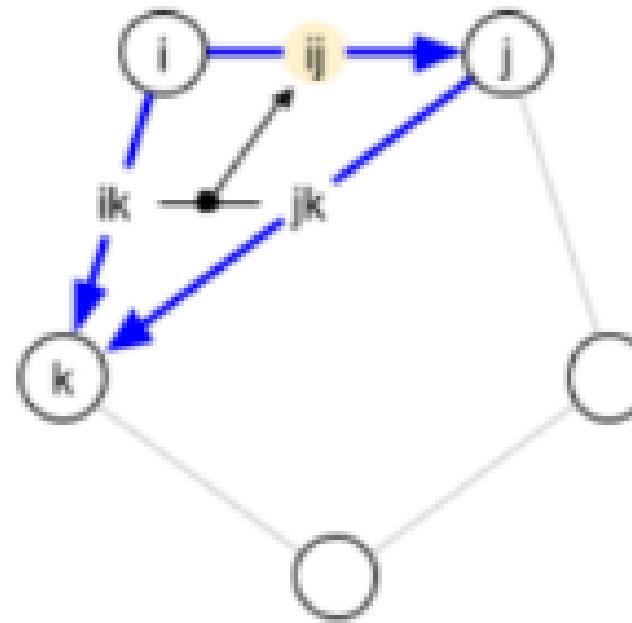
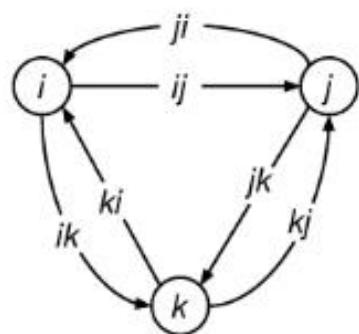
MSA – wielokrotne wyrównywanie
sekwencji

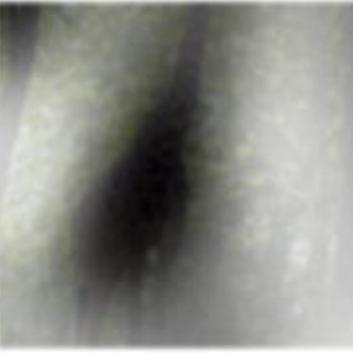
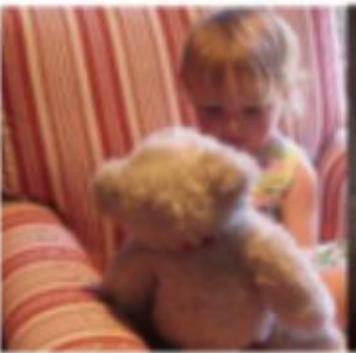
Reprezentacja par

b Pair representation
(r, r, c)



Corresponding edges
in a graph



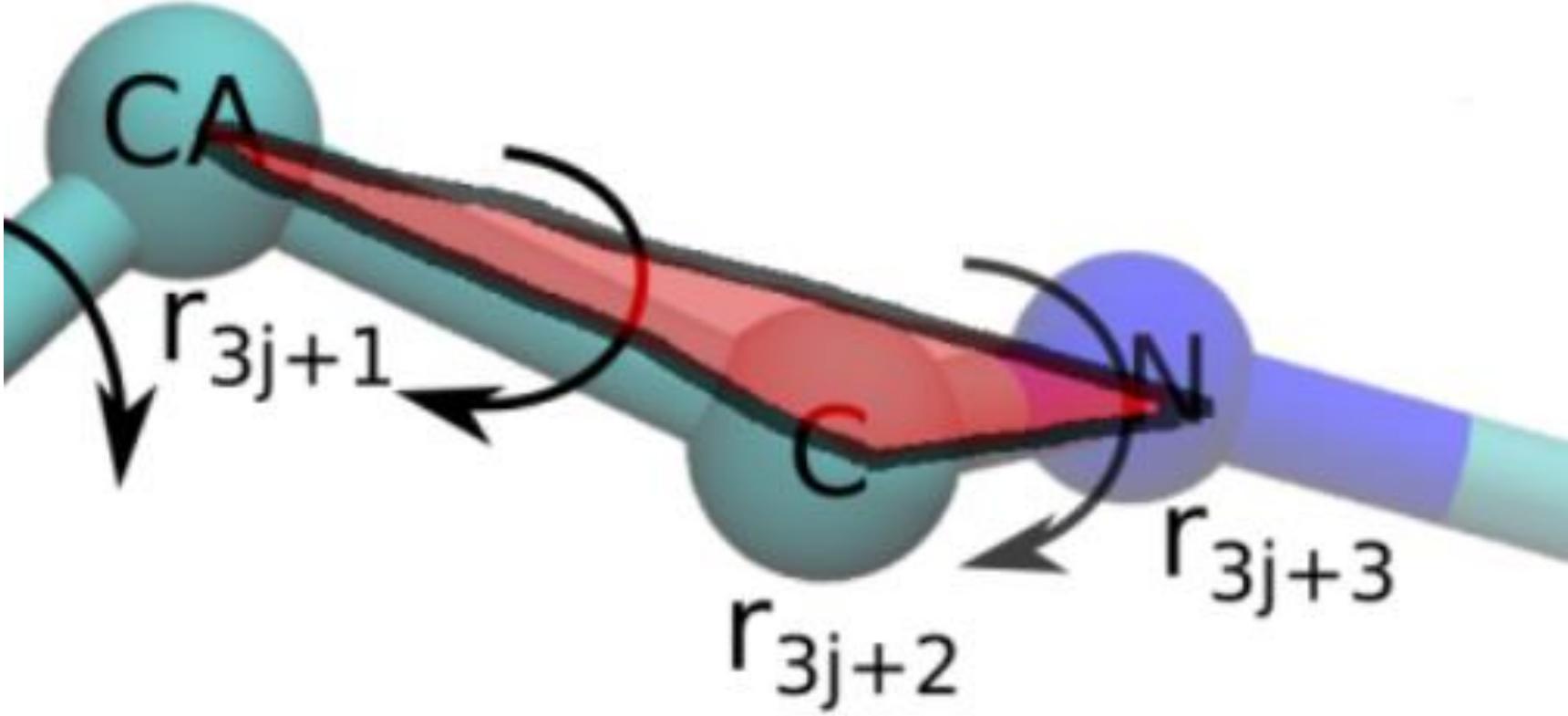


A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

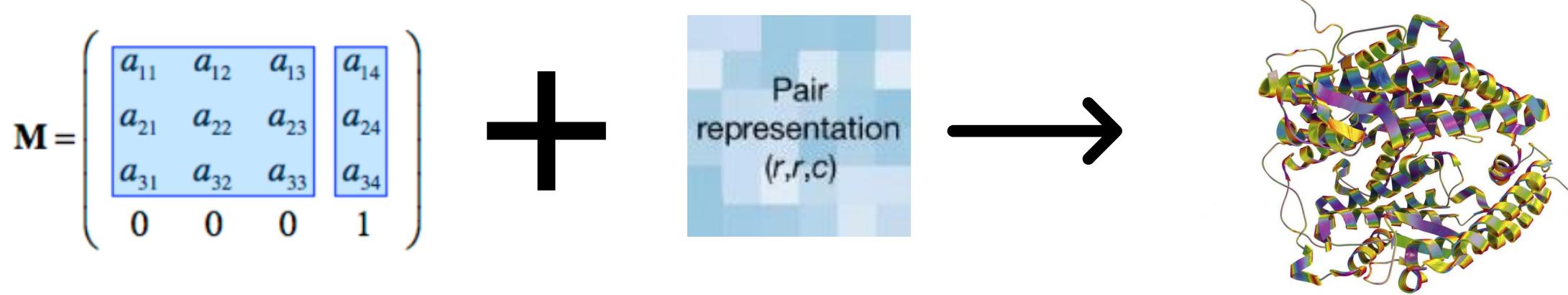
A giraffe standing in a forest with trees in the background.

Kolejne nowatorskie podejście -
Evoformer



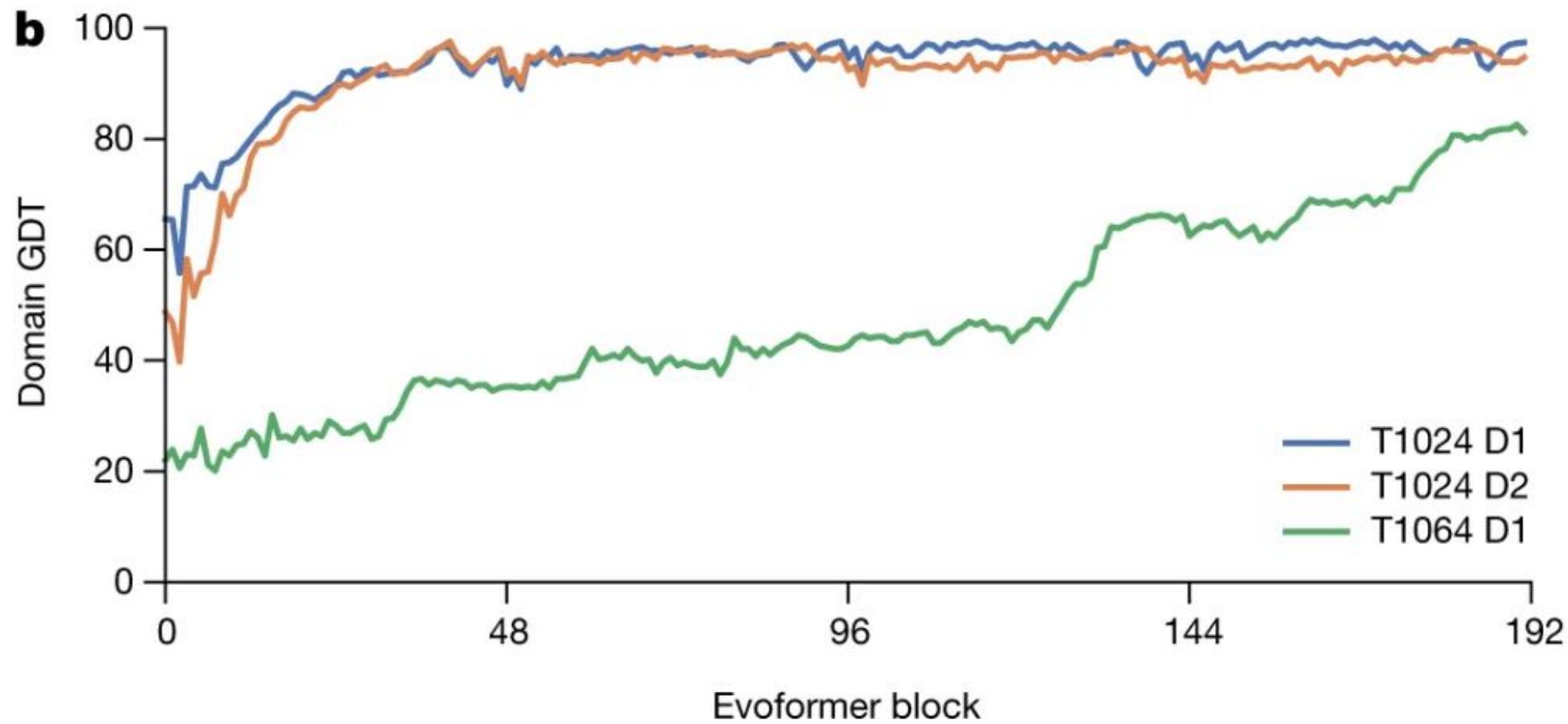
Tworzenie struktury - wstęp

Macierz Rotacji – tworzenie modelu struktury

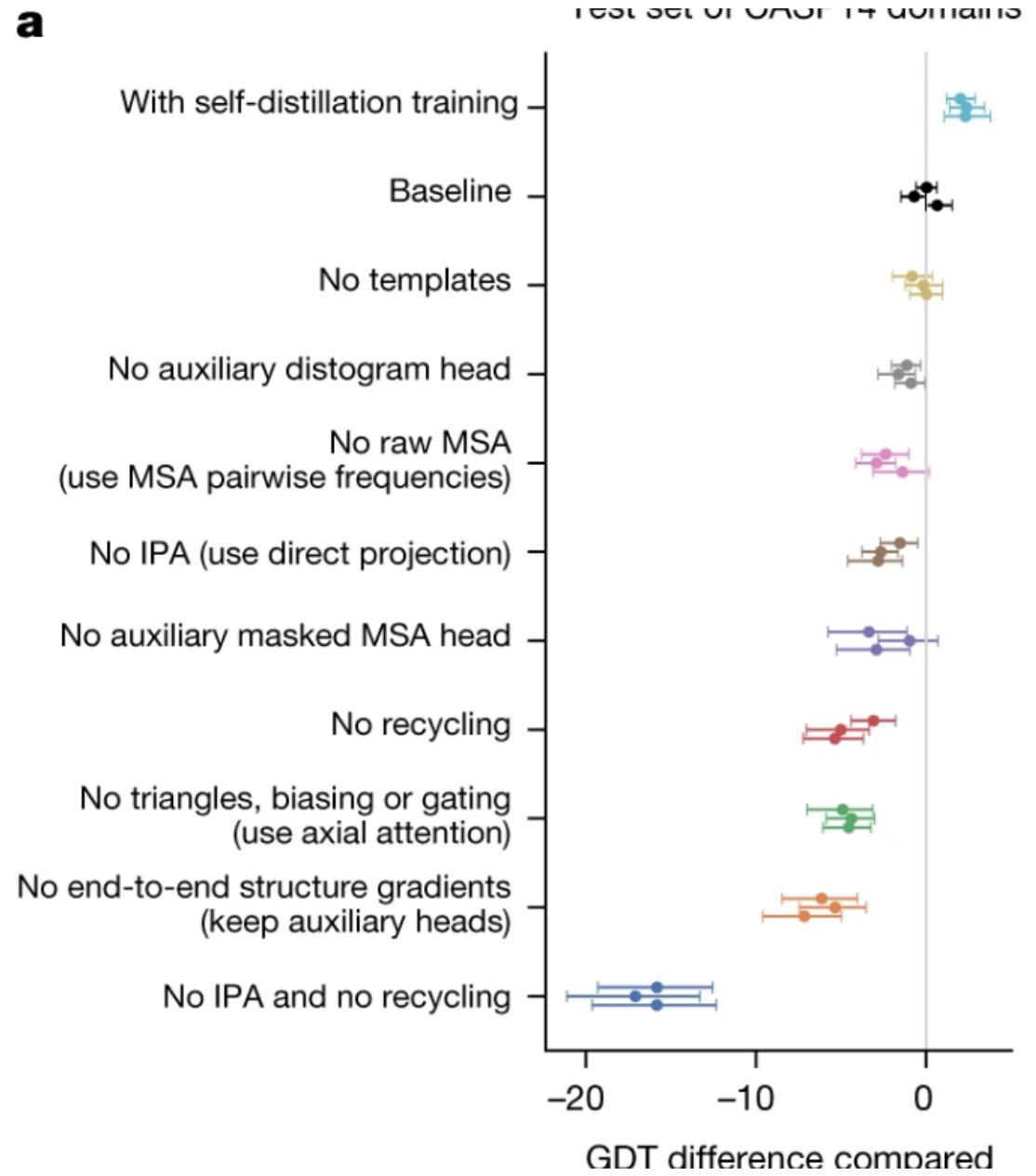


Macierz rotacji, zmienia położenie pojedynczych aminokwasów w macierzy reprezentacji par

Iteracje



Dodatkowe mechanizmy





AlphaFold w twoim domu

- Wzmianka o tym jak to zrobić:
- <https://towardsdatascience.com/what-does-alphafold-do-60b6370dafe4>

Bibliografia

- <https://www.nature.com/articles/s41586-021-03819-2>
- <https://www.blopig.com/blog/2021/07/alphafold-2-is-here-whats-behind-the-structure-prediction-miracle/>
- <https://towardsdatascience.com/what-does-alphafold-do-60b6370dafe4>
- <https://www.sztucznainteligencja.org.pl/alphafold2-przewidziec-strukture-zycia/>
- <https://www.youtube.com/watch?v=GQH-zWUylPY>
- <https://en.wikipedia.org/wiki/AlphaFold>
- <https://www.frontiersin.org/articles/10.3389/frai.2022.875587/full>

Dziękuję za uwagę

Franciszek Szczepaniak