Aproksymacja funkcji wielu zmiennych

Andrzej Odrzywolek

Instytut Fizyki, Zakład Teorii Względności i Astrofizyki

06.05.2009, środa, 10:15

Plan referatu

1. problem do rozwiązania: efektywne odtwarzanie wartości funkcji wielu ($n \geqslant 2$) zmiennych
2. pewne zjawiska i tendencje w technologii obliczeniowej i komputerowej: rosnące rozmiary pamięci i ilość procesorów
3. ogólne uwagi na temat „*dobrych*" funkcji wielu zmiennych i fałszywej intuicji matematycznej z nimi związanej
4. przykłady *nietrywialnych* funkcji wielu zmiennych: całka Fermiego-Diraca, widmo energetyczne neutrin, równania kinetyki chemicznej/jądrowej, transport promieniowania, mimika twarzy, zapis obrazu ruchomego obiektu oglądanego pod różnymi kątami, rozwiązania równań cząstkowych
5. Rozwiązania problemu: siłowe, matematyczne, analityczne, praktyczne, bioneuroinformatyczne

Zdefiniowanie problemu

Założenia

1. dana jest pewna nietrywialna, nieliniowa funkcja $f(x_1, \ldots, x_n)$ i znany (zaimplementowany) jest algorytm jej wyznaczenia
2. czas wymagany do policzenia wartości w pewnym punkcie jest relatywnie „duży" w porównaniu do spodziewanej ilości wywołań
3. wartości funkcji f nie są całkowicie przypadkowe, f jest co najmniej ciągła; ilość „informacji" potrzebna do odtworzenia funkcji f *w przybliżeniu* jest „niewielka"

Cel do osiągnięcia: funkcja aproksymująca $\hat{f}$ taka że:

1. ilośc danych niezbędna do obliczenia $\hat{f}$ nie może przekroczyć rozmiaru pamięci komputera
2. czas wymagany do obliczenia $\hat{f}$ musi być znacznie mniejszy niż ten potrzebny do obliczeń z aproksymowaną funkcją f (z generowaniem tablic/ wartości funkcji f **włącznie**)

Wyjaśnienie dotyczace aspektów mnogościowych

Kardynalność zbiorów funkcji

Przypomnienie:

- $\aleph_0$ (aleph zero): liczby naturalne, całkowite, wymierne, ...
- $\mathfrak{c}$ (continuum): liczby rzeczywiste $\mathfrak{R}$, punkty w przestrzeni euklidesowej $\mathfrak{R}^n$, zbiór wszystkich podzbiorów liczb naturalnych
- $\mathfrak{c} > \aleph_0$

Wyjaśnienie dotyczace aspektów mnogościowych

Kardynalność zbiorów funkcji

Przypomnienie:

- $\aleph_0$ (aleph zero): liczby naturalne, całkowite, wymierne, ...
- $\mathfrak{c}$ (continuum): liczby rzeczywiste $\mathfrak{R}$, punkty w przestrzeni euklidesowej $\mathfrak{R}^n$, zbiór wszystkich podzbiorów liczb naturalnych
- $\mathfrak{c} > \aleph_0$

Kardynalność zbioru wszystkich funkcji ciągłych $\mathfrak{R} \to \mathfrak{R}$ wynosi $\mathfrak{c}$, identycznie jak zbioru funkcji ciągłych $\mathfrak{R}^n \to \mathfrak{R}$
Nie ma więc mnogościowego powodu dla którego funkcja ciągła n zmiennych nie mogła by być przedstawiona przez funkcje ciągłe jednej zmiennej. Zaskakujące jest, że ilość wymaganych do tego celu funkcji jest niewielka, $2n + 2$.

Rozwiązania problemu aproksymacji funkcji n zmiennych

I. Tablicowanie i interpolacja, triangulacja, mesh refinement itd. (siłowe, $n = 2 \ldots 7$, górna granica stale rośnie)

II. Metody analityczne (szeregi potęgowe, funkcyjne, asymptotyczne, funkcje wymierne, itp., w praktyce $n = 2, 3$)

III. HDR (High Dimensional Rendering, $n = 2 \ldots 1000$)

IV. Wykorzystanie rozwiązania 13 problemu Hilberta (dekompozycja Arnolda/Kołmogorowa, 1957, czysto matematyczne (?), n dowolne)

V. Sieci neuronowe (2-warstwowy perceptron, $n = 2 \ldots ?$)

VI. (?) Wielokrotna superpozycja pewnych funkcji 2 zmiennych (n dowolne ?)

`Dobrym rozwiązaniem może okazać się też agresywna optymalizacja znanego algorytmu obliczania funkcji` f`; nie rozważam tutaj tego podejścia.`

Uwagi na temat aproksymacji

Aproksymacja w jednej zmiennej

Zagadnienie aproksymacji funkcji $\phi(x)$ jednej zmiennej x w klasie dowolnych funkcji jest źle określone: najlepszym aproksymantem jest bowiem sama funkcja ϕ! Musimy zawęzić klasę dopuszczalnych funkcji.

Aproksymacja w wielu zmiennych

Aproksymacja funkcji $f(x_1, \ldots, x_n)$ w klasie funkcji $\phi_k(x_1, \ldots, x_m)$ gdzie $2 \leqslant m < n$ zawsze ma sens bo istnieje nieskończenie wiele złożeń funkcji typu:

$$f(x, y, z) = \phi\Big(\phi\left(x, \phi\left(y, y\right)\right), z\Big)$$

Pytanie czy i jak w ten sposób można uzyskać z góry zadaną dokładność jest jednak bardzo nietrywialne.

Dlaczego to zagadnienie nabiera znaczenia dzisiaj?

20 lat temu nikomu nie przyszłoby do głowy tablicowanie funkcji o $n \geqslant 2$ zmiennych z powodu braku pamięci komputera!

Rozmiar pamięci komputera osobistego

1989 ZX Spectrum 48 kB ($n = \log_{100} 48kB = 2.35$: maksymalny wymiar tablicy o 100 elementach/zmienną)

1991 Amiga 500, 1MB ($n = 3.0$)

1997 PC, 32 MB ($n = 3.8$)

2009 PC, 4 GB + 768 MB (karta graf.) ($n = 4.9$)

- przestrzeń adresowa $2^{64} = 16\ 777\ 216$ TB, ($n = 9.6$)

Co wiecej, generowanie tablic wartosci funkcji jest problemem trywialnie rownoleglizowalnym, skalujacym sie liniowo z iloscia procesorow!

Przykłady funkcji wielu zmiennych (1)

Uwaga metodyczna

Nietrywialne funkcje więcej niż jednej zmiennej są rzadkością w matematyce „rachunkowej”. Szacuję, że 99% funkcji występujących w zbiorach zadań można sprowadzić do wyłącznie *eksponenty*, *dodawania* (dwuargumentowego!) i ich f. odwrotnych. Przykłady konwersji:

$$\sqrt{x} = \exp e^{\ln \ln x - \ln 2}, \qquad \mathrm{XLL2MEE}$$

$$\frac{x}{1+xy} = \exp\left[\ln x - \ln\left(1 + e^{\ln x + \ln y}\right)\right], \quad \mathrm{XLYLPE1PLXLME}$$

$$\frac{xy}{z} = \exp\left(\ln x + \ln y - \ln z\right), \qquad \mathrm{XLYLPZLME}$$

Bardzo upraszczając, „dociekliwy student” może uznać, że znajomość 1 funkcji 1 zmiennej (exp) i jednej funkcji 2 zmiennych (+, plus) *(posiadanie* **suwaka logarytmicznego***)* wystarcza.

Przykłady funkcji wielu zmiennych (2)

Całka Fermiego-Diraca

$$fd_k(\alpha,\eta) = \int_0^\infty \frac{x^k\sqrt{1+\frac{x}{2\alpha}}}{1+e^{x-\eta}}\,dx$$

Inna definicja:

$$G_n^{\pm}(\alpha,\beta) = \frac{1}{\alpha^{3+2n}}\int_\alpha^\infty \frac{x^{2n+1}\sqrt{x^2-\alpha^2}}{1+\exp(x\pm\beta)}\,dx,$$

Widmo energetyczne neutrin

$$\phi(\mathcal{E}_\nu, T, \rho, Y_e) = \int_A h(x_1,x_2,x_3;\mathcal{E}_\nu)\,d^3\mathbf{x} +$$

$$\sum_{i,j,k}\frac{(2J_{ik}+1)e^{-\frac{E_{ik}}{kT}}}{G(N,Z,kT)}\,\lambda_{ijk}^p\,\frac{\mathcal{E}_\nu^2(Q_{ijk}\pm\mathcal{E}_\nu)^2}{1+\exp\left(\frac{\pm\mathcal{E}_\nu\pm Q_{ijk}\pm\mu}{kT}\right)}\Theta_p, \quad p=\varepsilon^{\pm},\beta^{\pm}$$

Przykłady funkcji wielu zmiennych (3)

Kinetyka reakcji chemicznych/jądrowych

Rozwiązanie układu n nieliniowych (sztywnych) równań różniczkowych zwyczajnych typu:

$$\frac{dX_i}{dt} = \sum_j r_j X_j + \sum_{jk} \rho N_A r_{jk} X_j X_k + \sum_{jkl} \rho^2 N_A^2 r_{jkl} X_j X_k X_l$$

Rozwiązaniem tego równania jest n funkcji $n+3$ zmiennych:

$$X_i(t, \rho, T, X_1, \ldots, X_n)$$

Jeżeli bylibyśmy w stanie efektywnie odtwarzać raz obliczone wartości tych funkcji tych funkcji dla $n = 7 \ldots 50$ szybkość symulacji supernowych Ia itp. w 3D, jak również symulacji procesów np: chemicznych mogłaby wzrosnąć o rzędy wielkości.

Przykłady funkcji wielu zmiennych (4)

Nagrzewanie powierzchni Ziemi

Skomplikowana funkcja, zachmurzenia, zapylenia, szerokości geograficznej, aktywności słonecznej, dnia roku, temperatury, ciśnienia, wysokości n.p.m. itd.; w sumie około 60 zmiennych.

Ponownie, raz porządnie obliczona funkcja może być używana we *wszystkich* symulacjach pogody i klimatu, jeżeli tylko będziemy w stanie ją efektywnie odtwarzać.

Tablicowanie funkcji n zmiennych

- generujemy tyle punktów np. na siatce „prostokątnej” ile potrzebujemy do osiągnięcia wymaganej dokładności i dokonujemy pomiędzy nimi interpolacji.
- metoda jest skuteczna w niewielkiej liczbie wymiarów $n = 2, 3, 4$ z powodu błyskawicznie rosnącego rozmiaru tablicy
- sytuacja w której tablica nie zmieści się w fizycznej pamięci RAM powoduje katastrofalny spadek szybkości o ile program w ogóle się uruchomi/skompiluje
- drastyczna redukcja ilości punktów siatki często jest skuteczna; np. w pewnej pracy o supernowych użyto tylko 7 (!) wartości energii neutrina bez istotnej utraty dokładności (dla porównania ja używam 100-200 punktów)
- teoretycznie rozmiar przechowywanej tablicy można zoptymalizować np. poprzez symplicizację i zagęszczanie punktów w obszarze o dużej zmienności (np. krzywizny hiperpowierzchni R); w praktyce **efektywny algorytm** wyznaczenia optymalnego rozłożenia punktów przy zadanej metodzie interpolacji *jest nieznany* (?)

Metody matematyczne (całka Fermiego-Diraca)

Jeżeli funkcja jest zadana w postaci analitycznej można ją często sprowadzić do prostszej postaci przybliżonej.
W zależności od parametrów α, β rozwijamy w funkcji podcałkowej w szereg wyrażenie w mianowniku:

$$\frac{1}{1+e^{x\pm\beta}} = \sum_{m=1}^{\infty}(-1)^{m+1}e^{-(x\pm\beta)}$$

bądź pierwiastek w liczniku:

$$\sqrt{1+\frac{2\alpha}{x}} = 1+\alpha/x-\frac{1}{2}(\alpha/x)^2+\ldots$$

i całkujemy wyraz po wyrazie. Otrzymujemy dzięki temu pokrycie płaszczyzny α, β pewnymi wyrażeniami poskładanymi z funkcji specjalnych jednej zmiennej: f. Fermiego (polilogarytmu) i f. Macdonalda. **Eleganckie, efektywne ale pracochłonne.**

High Dimensional Rendering

Rozważmy dokładny, ale niejednoznaczny rozkład funkcji f:

$$f(x_1,\ldots,x_n) = f_0 + \sum_i f_i(x_i) \sum_{ij}{}' f_{ij}(x_i,x_j) + \sum_{ijk}{}' f_{ijk}(x_i,x_j,x_k) + \ldots + \delta f$$

Praktyka pokazuje, że przyjmując np:

$$f_0 = \int_{A^n} f(\mathbf{x}) d^n x$$

$$f_i(x_i) = \int_{A^{n-1}} f(\mathbf{x}) d^{n-1} x - f_0$$

$$f_{ij}(x_i,x_j) = \int_{A^{n-2}} f(\mathbf{x}) d^{n-2} x - \sum_k f_k(x_k) - f_0$$

$$\ldots$$

otrzymujemy szybkozbieżny rozkład funkcji n zmiennych na funkcje $m \ll n$ zmiennych. Zmienne muszą jednak wpływać na f w skorelowany sposób.

13 problem Hilberta

Oryginalne sformułowanie hipotezy (D. Hilbert, 1900)

Rozwiązania f(a,b,c) równania:

$$x^7 + a\,x^3 + b\,x^2 + c\,x + 1 = 0$$

nie mogą być przedstawione jako superpozycje **ciągłych** funkcji **dwóch zmiennych**.

Rozwiązanie podane przez W. Arnolda i A. Kołmogorowa (1957)

13 problem Hilberta

Oryginalne sformułowanie **hipotezy** (D. Hilbert, 1900)

Rozwiązania f(a,b,c) równania:

$$x^7 + a\,x^3 + b\,x^2 + c\,x + 1 = 0$$

nie mogą być przedstawione jako superpozycje **ciągłych** funkcji **dwóch zmiennych**.

Rozwiązanie podane przez W. Arnolda i A. Kołmogorowa (1957)

Każda ciągła funkcja n-zmiennych $f(x_1, \ldots, x_n)$ określona w obszarze $[0,1]^n$ może być przedstawiona jako:

$$f(x_1, \ldots, x_n) = \sum_{q=0}^{2n} g_p \left(\sum_{p=1}^{n} \phi_{pq}(x_p) \right)$$

gdzie funkcje jednej zmiennej g_p, ϕ_{pq} są ciągłe.

Współczesna wersja twierdzenia

Twierdzenie Kołmogorowa o superpozycji

Każda ciągła funkcja n-zmiennych $f(x_1, \ldots, x_n)$ określona w obszarze $[0,1]^n$ może być przedstawiona jako:

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} g\left(\sum_{p=1}^{n} \lambda_p \phi_q(x_p)\right)$$

gdzie funkcje jednej zmiennej g, ϕ_q są ciągłe a stałe rzeczywiste $0 < \lambda_p \leqslant 1$. $2n+1$ ściśle rosnących funkcji $\phi_q : [0,1] \to [0,1]$ zależy tylko od wymiaru przestrzeni n. Spełniają one warunek Lipschitza.
Jedna funkcja ciągła $g : [0,n] -> \mathbb{R}$ zależy od funkcji f.

Przypadek dwóch zmiennych

$$f(x,y) = \sum_{q=1}^{5} g\Big(\phi_q(x) + \lambda\,\phi_q(y)\Big)$$

- 5 funkcji jednej zmiennej $\phi_q(t)$ posiada specjalną kwazi-schodkową strukturę pozwalającą dokonać wraz ze stałą niewymierną λ mapowania kwadratu $[0,1]\times[0,1]\to[0,2]$.
- funkcja $g(t)$ służy następnie do odtworzenia wartości funkcji $f(x,y)$ na zmapowanym odcinku $[0,2]$.

Ilość funkcji w tym twierdzeniu **ma istotne znaczenie**. Na przykład W. Arnold udowodnił, że w ogólności:

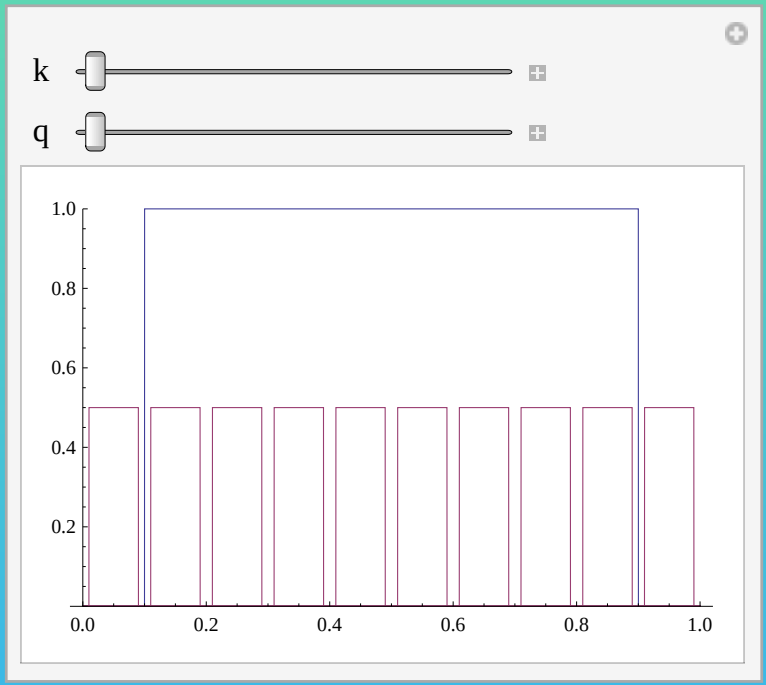
$$f(x,y) \neq \chi\Big(\phi(x) + \psi(y)\Big)$$

Szkic dowodu dla $n = 2$

- w k-tym kroku iteracji definiujemy 5 rodzin „kwadracików” $I^k_{q,i} \times I^k_{q,j}$, gdzie:

$$I^k_{q,i} = \left[\frac{i}{10^{k-1}} + \frac{3-2q}{10^k}, \frac{i}{10^{k-1}} + \frac{11-2q}{10^k}\right] \cap [0,1]$$

Out[63]=



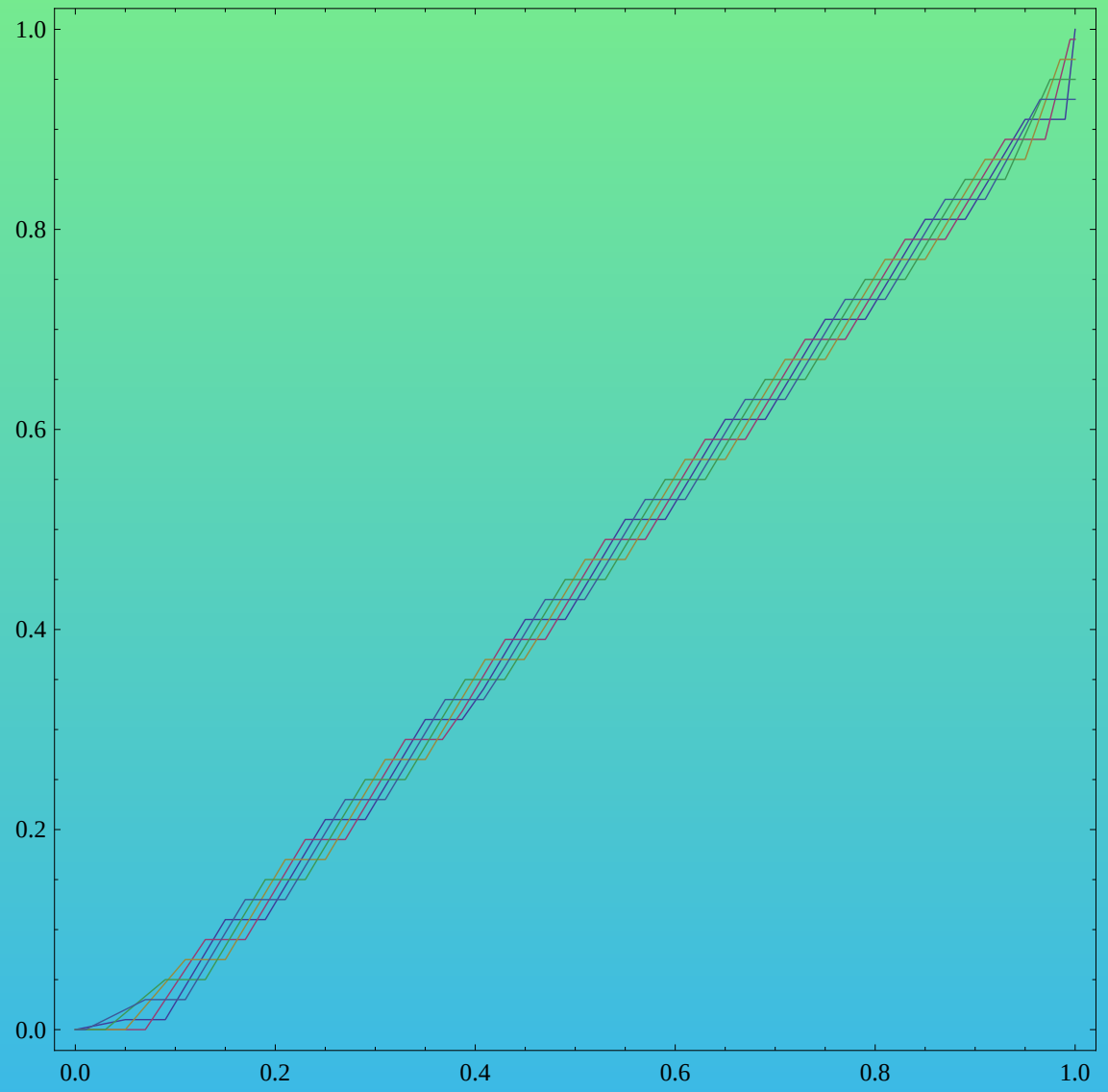
Diabelskie schody

- tworzymy 5 funkcji 1 zmiennej ϕ_q kawałkami liniowych i stałych, poprzez podanie wartości jakie mają przyjmować na końcach uprzednio zdefiniowanych przedziałów, tak aby wszystkie przedziały będące obrazami „kwadracików" poprzez funkcje $\phi_q(x) + \lambda\phi_q(y)$:

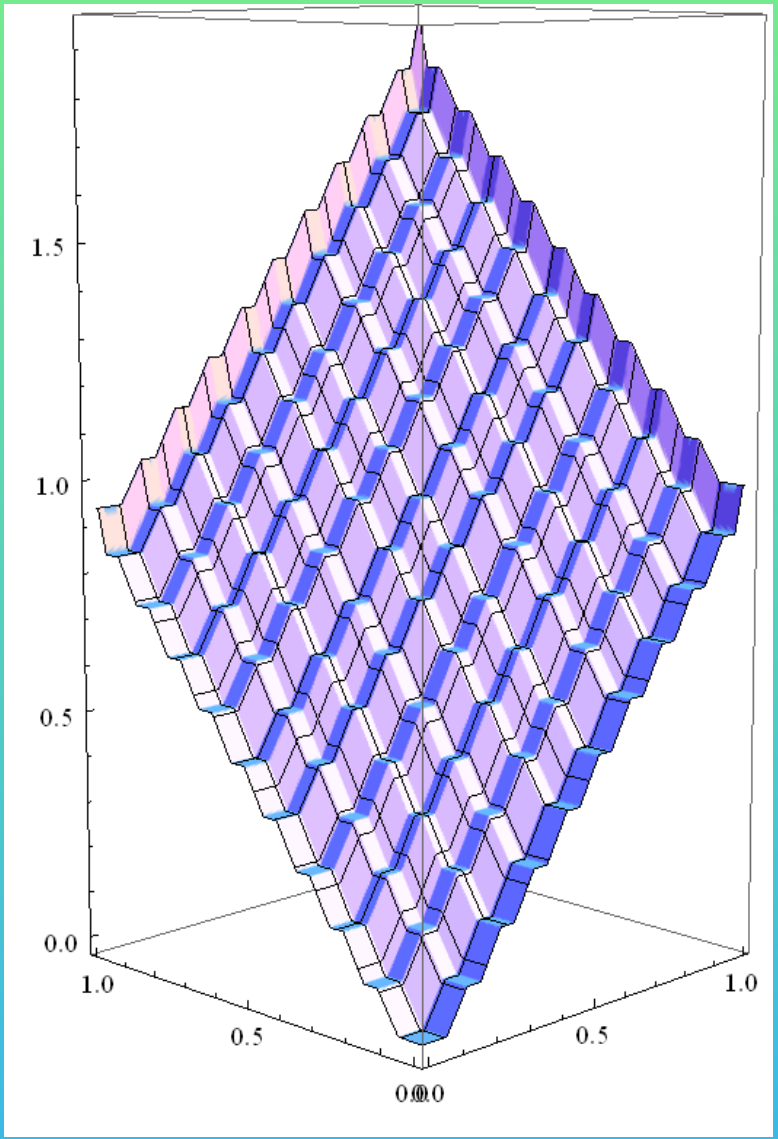
$$\Delta^k_{qij} = \phi_q\left(I^k_{qi}\right) + \lambda\phi_q\left(I^k_{qj}\right)$$

  były rozłączne.
- pokazuje się to przez indukcję; λ jest liczbą niewymierną, pozostałe liczby (argumenty i wartości funkcji) są wymierne
- w granicy $k \to \infty$ funkcje ϕ_q są ciągłe i monotoniczne rosnące

Funkcje ϕ_q dla $k = 2$



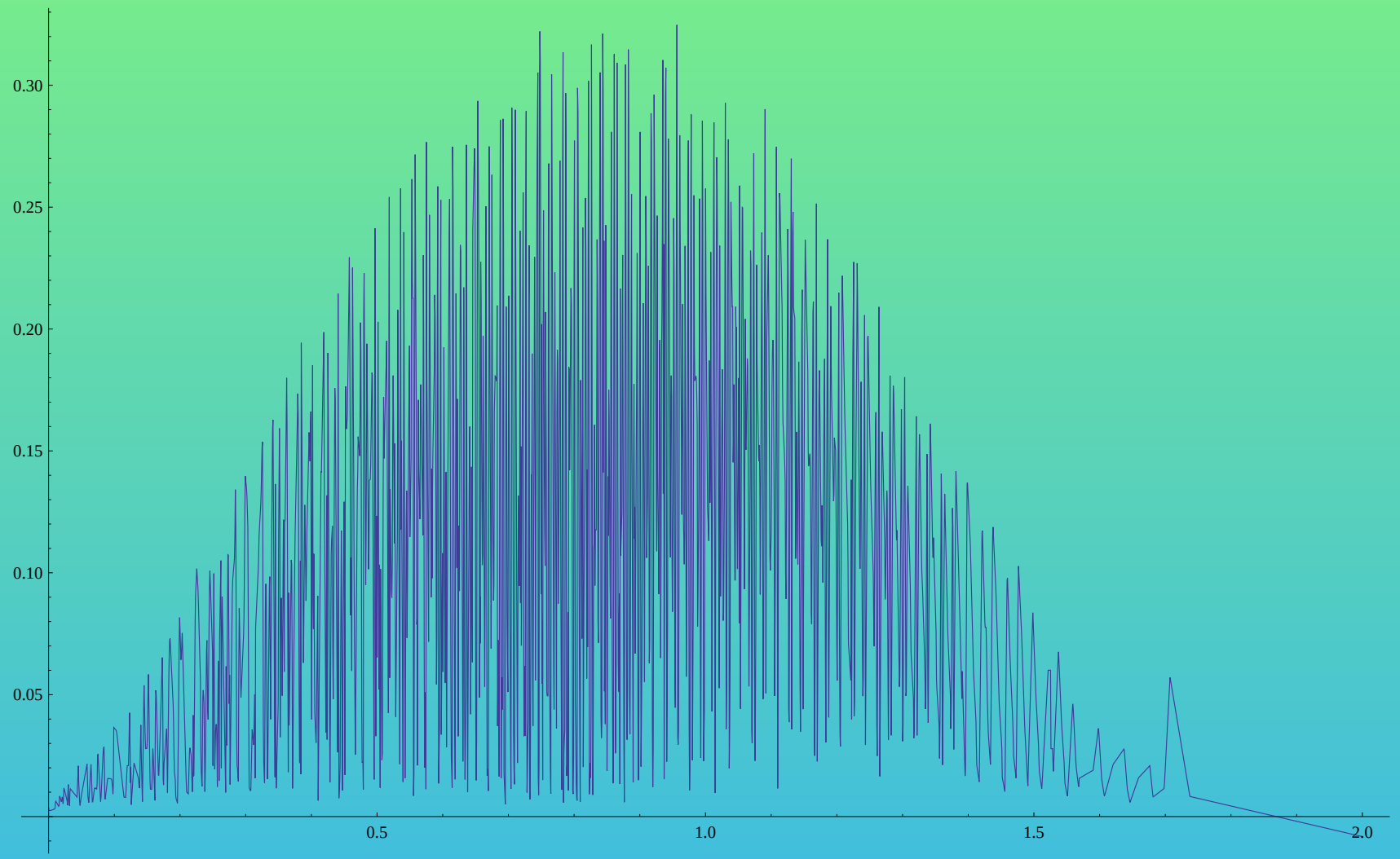
Funkcja $\phi_2(x) + \lambda\phi_2(y)$ dla $\lambda = \sqrt{97/103}$



Konstrukcja zewnętrznej funkcji g

- poprzez 5 funkcji ϕ_q możemy zmapować kwadrat $[0,1] \times [0,1]$ na odcinek $[0,2]$ jako $\phi_q(x) + \lambda\phi_q(y)$, a małe „kwadraciki" na rozłączne przedziały
- na k-tym kroku iteracji bierzemy dla każdego zmapowanego przedziału wartość $g = f(X,Y)/3$, gdzie (X,Y) to współrzędne środka odpowiadającego mu „kwadracika"; w obszarach poza przedziałami interpolujemy liniowo
- ponieważ „szansa" że dany punkt (X,Y) zostanie pokryty przez więcej niż 3 „kwadraciki" jest mniejsza niż 1, otrzymujemy szereg kolejnych przybliżeń który jest zbieżny

Przykład zerowego przybliżenia dla funkcji g

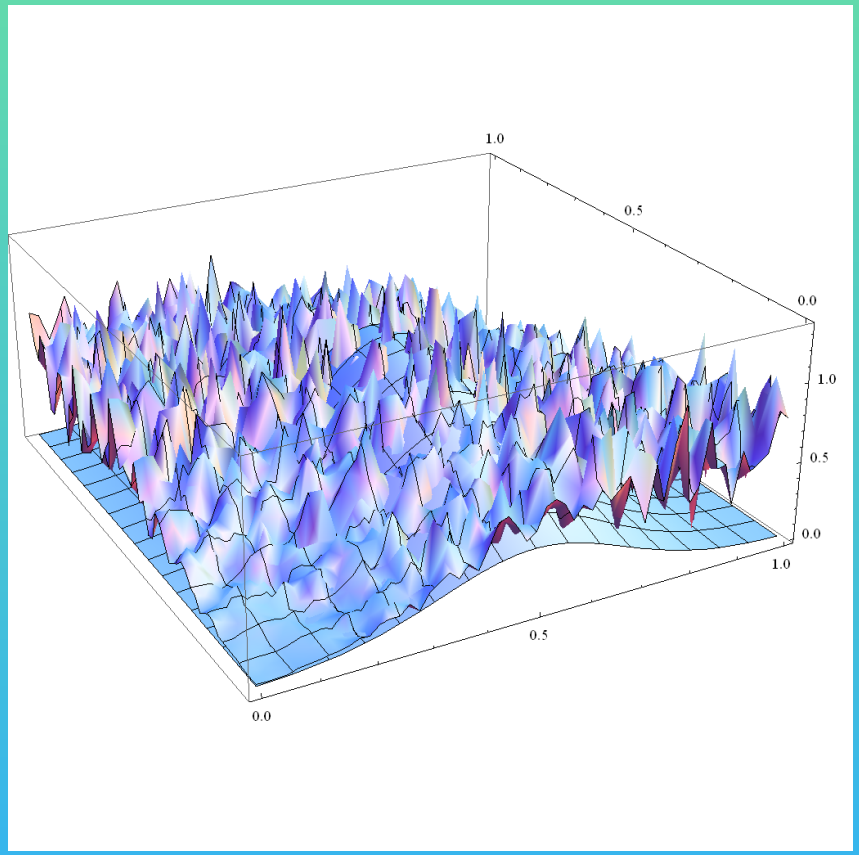


Funkcja g jest ciągła, ale z całą pewnością niełatwa do aproksymacji . . .

Złożenie funkcji g i funkcji $\phi_q(x) + \lambda\phi_q(y)$

Trudno sobie to wyobrazić, ale wyrażenie poniżej rzeczywiście odtwarza w przybliżeniu funkcję f:

$$\sum_{q=1}^{5} g\left(\phi_q(x) + \lambda\phi_q(y)\right) \simeq e^{-16(x-1/2)^2-4(y-1/2)^2}$$



Dlaczego to twierdzenie przeszło „bez echa”

1. stosuje się wyłącznie do reprezentacji funkcji ciągłych przez ciągłe
2. dodanie jakichkolwiek innych własności „psuje” twierdzenie: nie można np. przedstawić f. analitycznej n zmiennych przez analityczne jednej zmiennej, różniczkowalnej przez różniczkowalne itd.
3. jest sprzeczne z intuicją
4. nie ma prawie żadnych zastosowań
5. funkcje $\phi_q(t)$ są konstruowane w granicy pewnej procedury rekurencyjnej operującej liczbami wymiernymi i niewymiernymi
6. aczkolwiek f. g jest formalnie niezależna of ϕ_q jej konstrukcja odbywa się naprzemiennie z konstrukcją funkcji ϕ_q
7. proces ten jest słabo zbieżny, a sama funkcja g bardzo skomplikowana

Próby „zaprzęgnięcia” tw. Kołmogorowa do aproksymacji

1. H.L. Frisch et. al., *Approximate Representation of Functions of Several Variables in Terms of Functions of One Variable*, Phys. Rev. Lett., Vol. 63, Number 9, 1989.
2. Manuela Nees, *Aproximative versions of Kolmogorov's superposition theorem, proved constructively*, J. Comput. Appl. Math. 54 (1994) 239-250.
3. V. Kurkova, *Kolmogorov theorem and multilayer neural networks*, Neural Networks, 5 (1992) 501-506.
4. Manuela Nees, *Chebyshev approximation by discrete superpositions. Application to neural networks.*, Advances in Computational Mathematics **5** (1996) 137-151.

Jak widać z tytułów, próba zastosowania reprezentacji Kołmogorowa prowadzi prosto do pojęcia **sieci neuronowej**

Od Kołmogorowa do sieci neuronowej (1)

Zbieżność (M. Ness, 1996)

$$\left| f(x_1, \ldots, x_n) - \sum_{q=1}^{2n+1} \sum_{r=1}^{R} g_r\left(\sum \phi_q^r\right) \right| \leqslant \theta^R ||f||_\infty, \quad \theta > \frac{n}{n+1}$$

R - ilość iteracji w konstrukcji funkcji g_r, ϕ_q^r.
Frish et. al. dla $n = 2$ określa zbieżność jako „*amazingly quick*”, natomiast wynik powyżej nie budzi entuzjazmu.

Od Kołmogorowa do sieci neuronowej (1)

Zbieżność (M. Ness, 1996)

$$\left| f(x_1, \ldots, x_n) - \sum_{q=1}^{2n+1} \sum_{r=1}^{R} g_r\left(\sum \phi_q^r\right) \right| \leqslant \theta^R ||f||_\infty, \quad \theta > \frac{n}{n+1}$$

R - ilość iteracji w konstrukcji funkcji g_r, ϕ_q^r.
Frish et. al. dla $n = 2$ określa zbieżność jako „*amazingly quick*”, natomiast wynik powyżej nie budzi entuzjazmu.
Jeżeli jednak **zwiększymy** ilość funkcji ϕ_q to:

$$\left| f(x_1, \ldots, x_n) - \sum_{q=1}^{m} \sum_{r=1}^{R} g_r\left(\sum \phi_q^r\right) \right| \leqslant \theta^R ||f||_\infty, \quad \theta > \frac{n}{m-n}$$

Dla odpowiednio dużej liczby m możemy osiągnąć zadaną dokładność nawet w *jednym* kroku iteracji !

Od Kołmogorowa do sieci neuronowej (2)

Dalsze usprawnienie reprezentacji

Ciągle problem stanowią niewygodne funkcje ϕ_q, tzw. „devil's staircase". Czy można je zastąpić przez coś prostszego?

Funkcje **sigmoidalne**

Funkcję (nielinową) która w sposób monotoniczny przekształca $\mathbb{R} \to (0,1)$ nazywamy sigmoidalną np:

$$S(t) = \frac{1}{1+e^{-x}}, K(t) = (1 + \operatorname{tgh} x)/2$$

Propozycja użycia funkcji $S(t)$, $K(t)$ lub podobnej pojawiła się poprzez analizę budowy biologicznych neuronów.

Równoważność sieci neuronowej i funkcji n zmiennych

Reprezentacja w postaci **sieci neuronowej**

Czy reprezentacja:

$$f(x_1, \ldots, x_n) = \sum_k \alpha_k S\Big(\sum_{i=1}^{n} \lambda_i x_i\Big), \quad S(t) = 1/(1 + e^{-t}),$$

nazywana siecią neuronową o jednej warstwie ukrytej jest równoważna dekompozycji Kołmogorowa, t.j. dowolnej ciągłej funkcji n zmiennych, jeżeli k dopuszczamy dowolnie duże?

Równoważność sieci neuronowej i funkcji n zmiennych

Reprezentacja w postaci **sieci neuronowej**

Czy reprezentacja:

$$f(x_1, \ldots, x_n) = \sum_k \alpha_k S\Big(\sum_{i=1}^{n} \lambda_i x_i\Big), \quad S(t) = 1/(1 + e^{-t}),$$

nazywana siecią neuronową o jednej warstwie ukrytej jest równoważna dekompozycji Kołmogorowa, t.j. dowolnej ciągłej funkcji n zmiennych, jeżeli k dopuszczamy dowolnie duże?

Twierdzenie Cybenki

Równoważność sieci neuronowej i funkcji n zmiennych

Reprezentacja w postaci **sieci neuronowej**

Czy reprezentacja:

$$f(x_1, \ldots, x_n) = \sum_k \alpha_k S\Big(\sum_{i=1}^{n} \lambda_i x_i\Big), \quad S(t) = 1/(1 + e^{-t}),$$

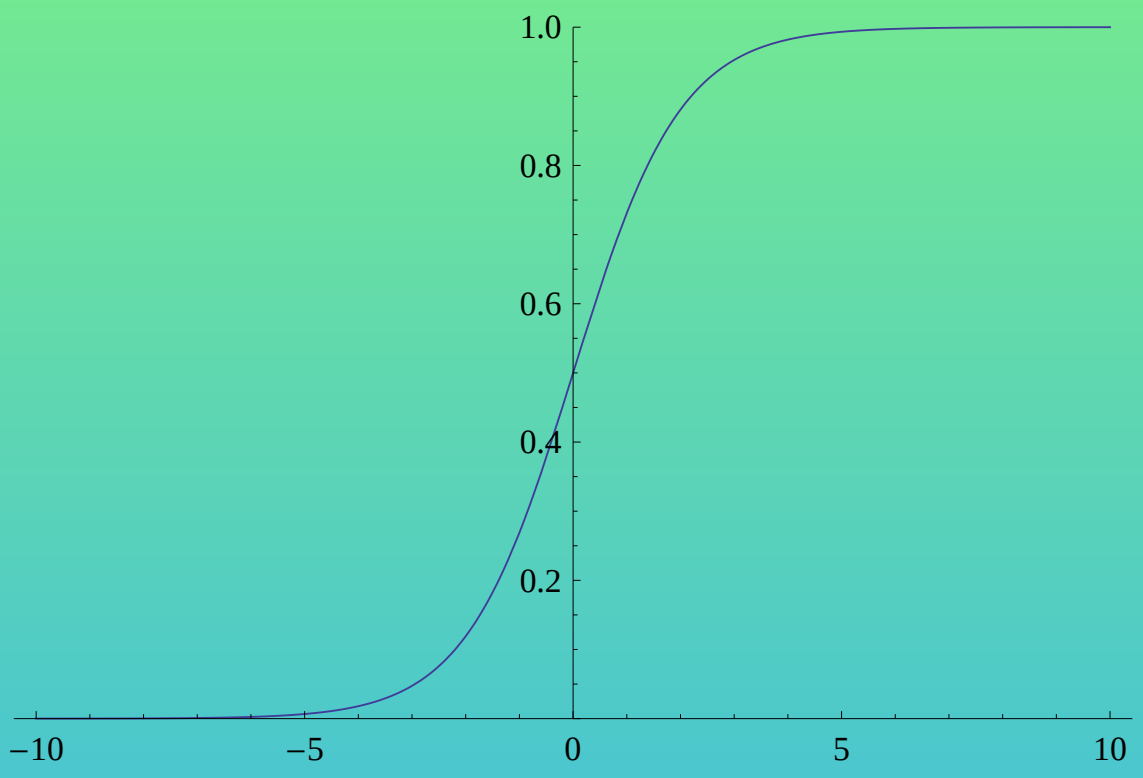
nazywana siecią neuronową o jednej warstwie ukrytej jest równoważna dekompozycji Kołmogorowa, t.j. dowolnej ciągłej funkcji n zmiennych, jeżeli k dopuszczamy dowolnie duże?

Twierdzenie Cybenki

Dla każdej ciągłej funkcji $f(x_1, \ldots, x_n)$ i stałej dodatniej ϵ istnieje skończona ilość stałych $\alpha_k, \lambda_i, \theta_i$ taka że:

$$\left| f(x_1, \ldots, x_n) - \sum_k \alpha_k S\Big(\sum_{i=1}^{n} \lambda_i x_i + \theta_i\Big) \right| < \epsilon$$

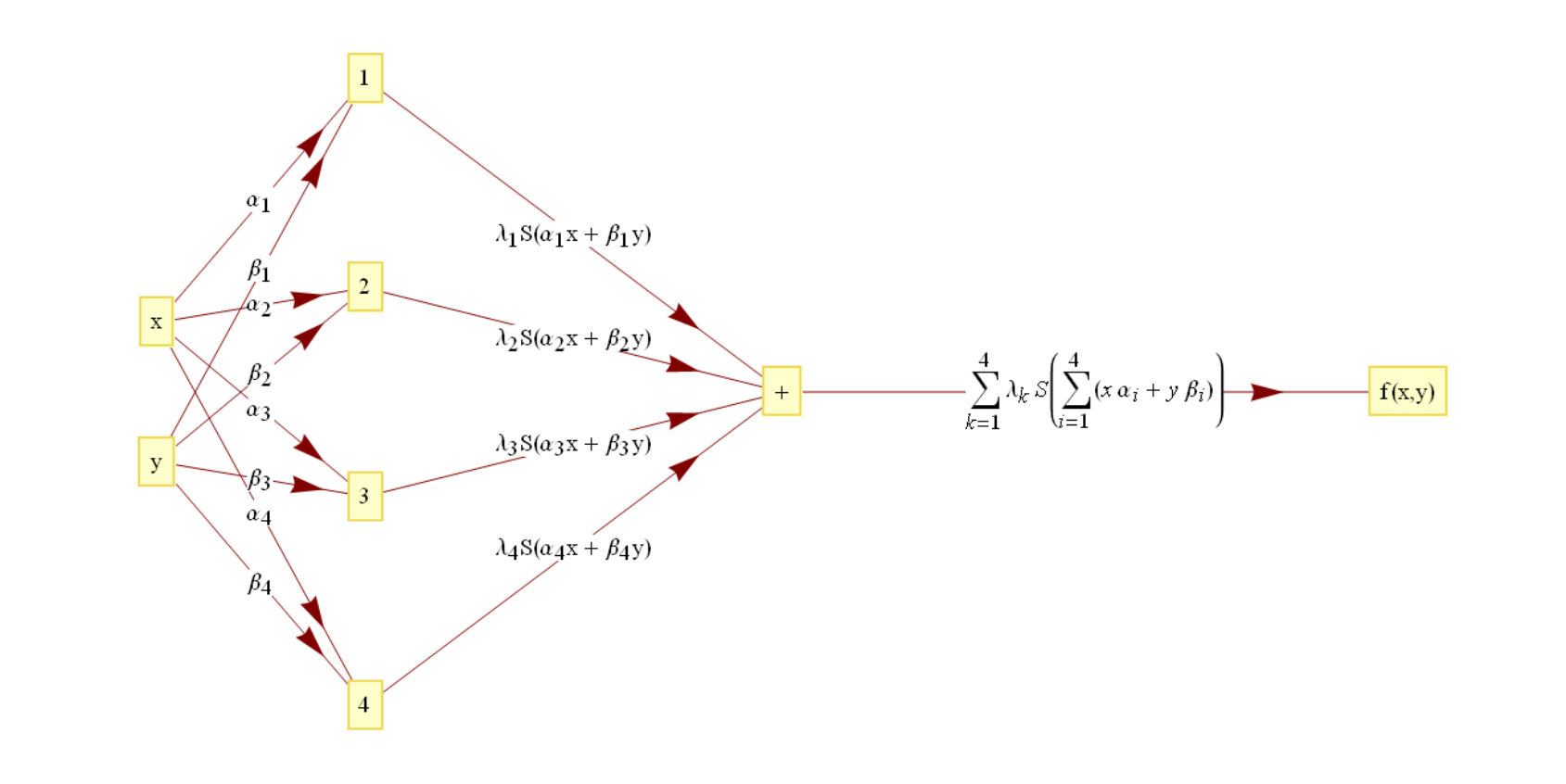
Funkcja sigmoidalna



$$S(t) = \frac{1}{1 + e^{-t}} = \frac{1}{2}\left(\operatorname{tgh}\frac{t}{2} + 1\right)$$

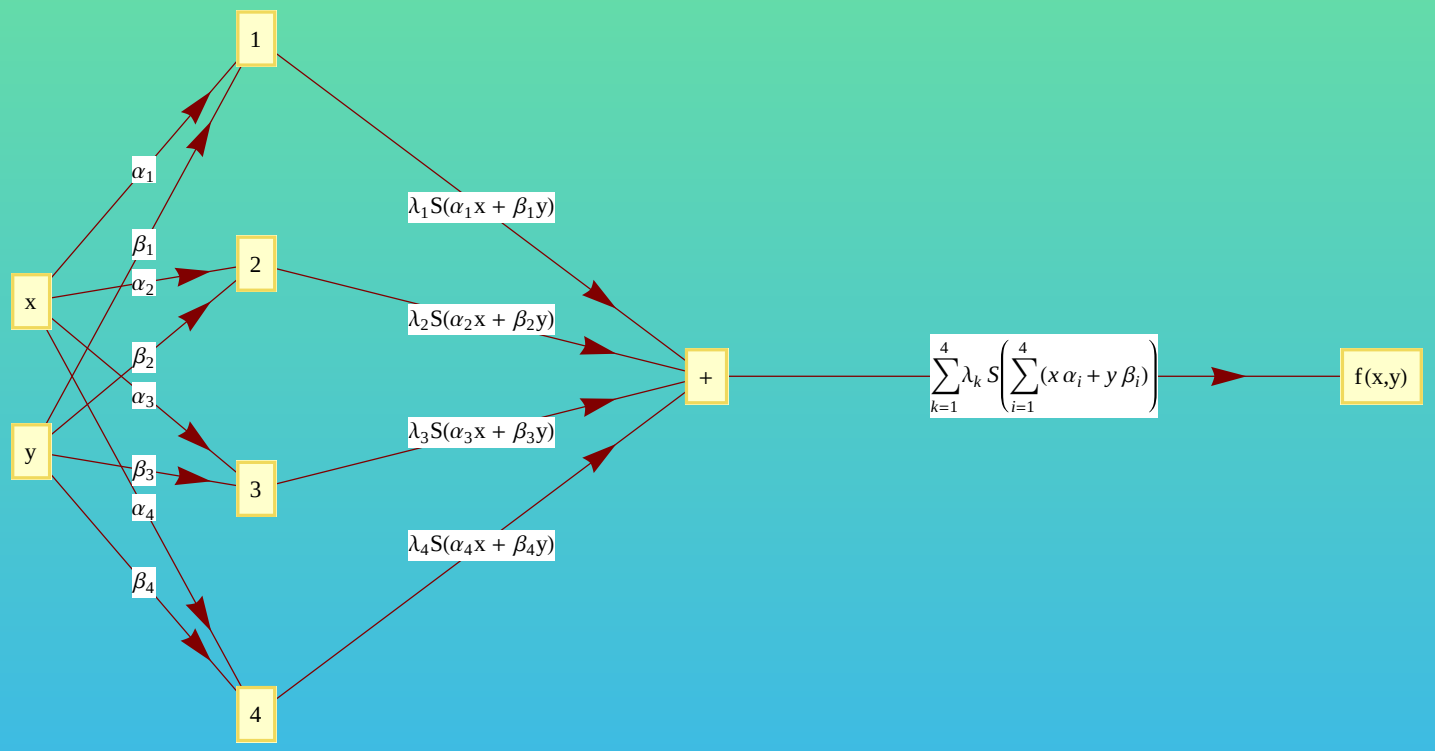
Przykład prostej małej sieci neuronowej

$$f(x,y) = \sum_{k=1}^{4} \lambda_k S\left(\sum_{i=1}^{4} \alpha_i x + \beta_i y\right)$$



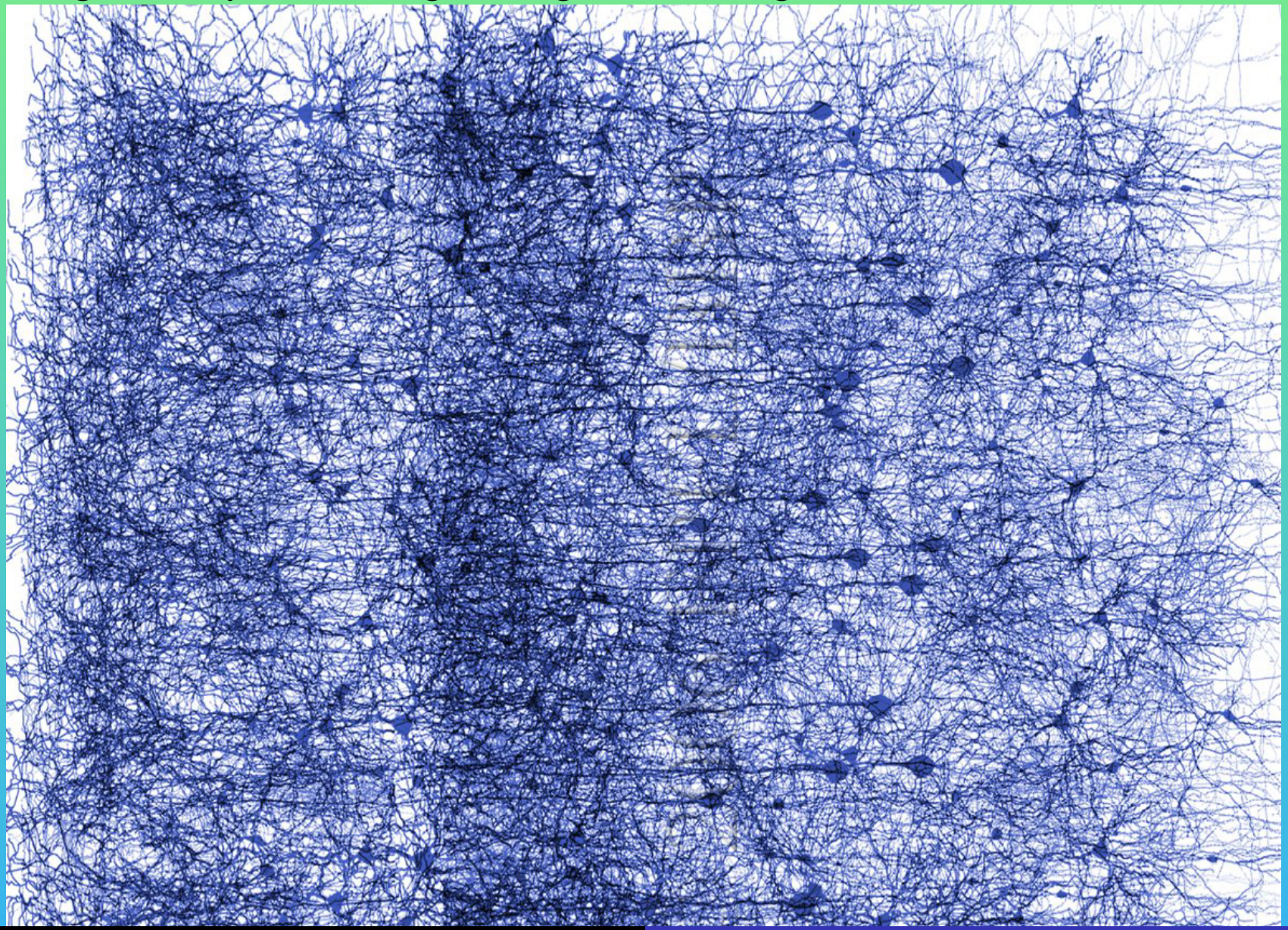
Przykład prostej małej sieci neuronowej

$$f(x,y) = \sum_{k=1}^{4} \lambda_k S\left(\sum_{i=1}^{4} \alpha_i x + \beta_i y\right)$$



Przykład dużej sieci neuronowej: Blue Brain Project

Fragment symulowanego fragmentu mózgu człowieka



Zastosowanie sieci neuronowych w praktyce

Słowniczek pojęć

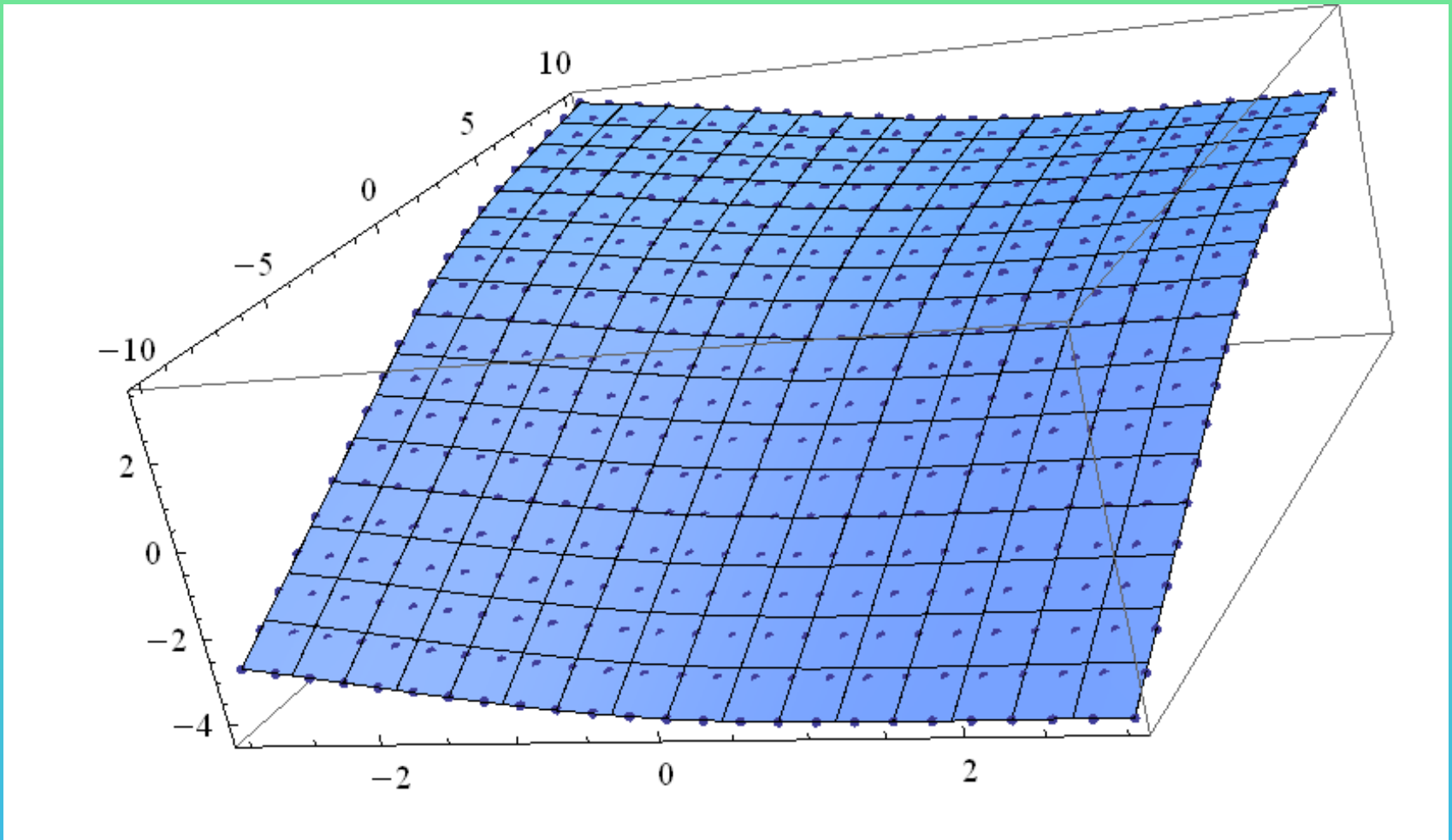
- **sieć neuronowa**: wieloparametrowa funkcja będąca superpozycją f. sigmoidalnych
- **uczenie sieci neuronowej**: dopasowanie najlepszych parametrów dopasowania **s. n.** do zbioru danych, w tym metodą najmniejszych kwadratów
- **wartwa ukryta** funkcje sigmoidalne wraz z parametrami
- **perceptron** sieć neuronowa zwracająca funkcję skalarną, jak przedstawiono na poprzednim slajdzie
- **sieć wielowarstowowa** - zamiast sumować wartości k funkcji sigmoidalnych, traktujemy je jako k zmiennych wejściowych dla kolejnej sieci neutronowej

Wyniki eksperymentów numerycznych dla całki F-D

1. dopasowanie jednowarstwowej sieci nie działa nawet dla setek neuronów (parametrów swobodnych modelu)
2. dla sieci 2-warstwowej do uzyskania dokładności na poziomie 0.5% w zakresie parametru $\ln\alpha = -6\ldots6$ (relatywistyczność), $\eta = -10\ldots10$ (degenracja) wystarczą 4 neurony w pierwszej warstwie i 3 w drugiej. Oznacza to że mamy do dyspozycji 28 dowolnych parametrów dopasowania.
3. wyjście (x2) poza zakres w którym fitujemy powoduje błędy na poziomie poniżej 50%
4. sieć neutronowa posiada bardzo dużo lokalnych minimów, nie ma pewności że fit jest najlepszy; pomimo tego dopasowanie jest „optycznie" zawsze bardzo dobre
5. szybko i skutecznie działają algorytmy szukające minimów lokalnych np. Levenberg-Marquadt z losową generacją danych startowych i selekcją najlepszych rozwiązań
6. standardowe metody „uczenia" typu „backpropagation" okazały się bardzo powolne i dające mizerne rezultaty
7. w **Mathematice 7.0**, „uczymy" sieć neuronową funkcją **FindFit**; bardzo skuteczne *(ale powolne!)* jest dodanie opcji **Method→NMinimize, NormFunction→Norm[#,Infinity]&** która włącza algorytmy globalnej optymalizacji

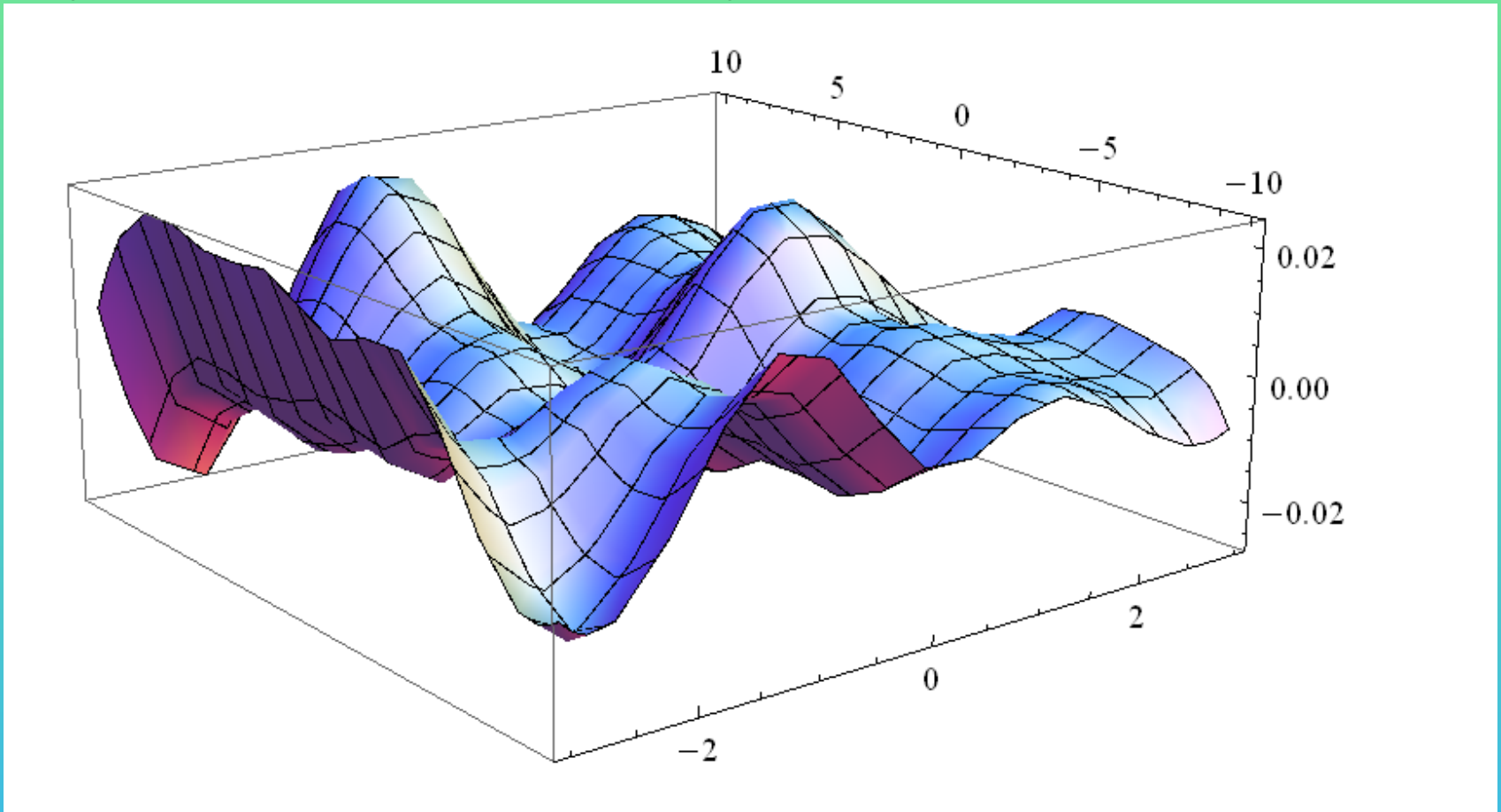
Przykładowy fit całki Fermiego-Diraca siecią neuronową

Dopasowanie „zbioru uczącego” i sieci neuronowej (23 parametry)



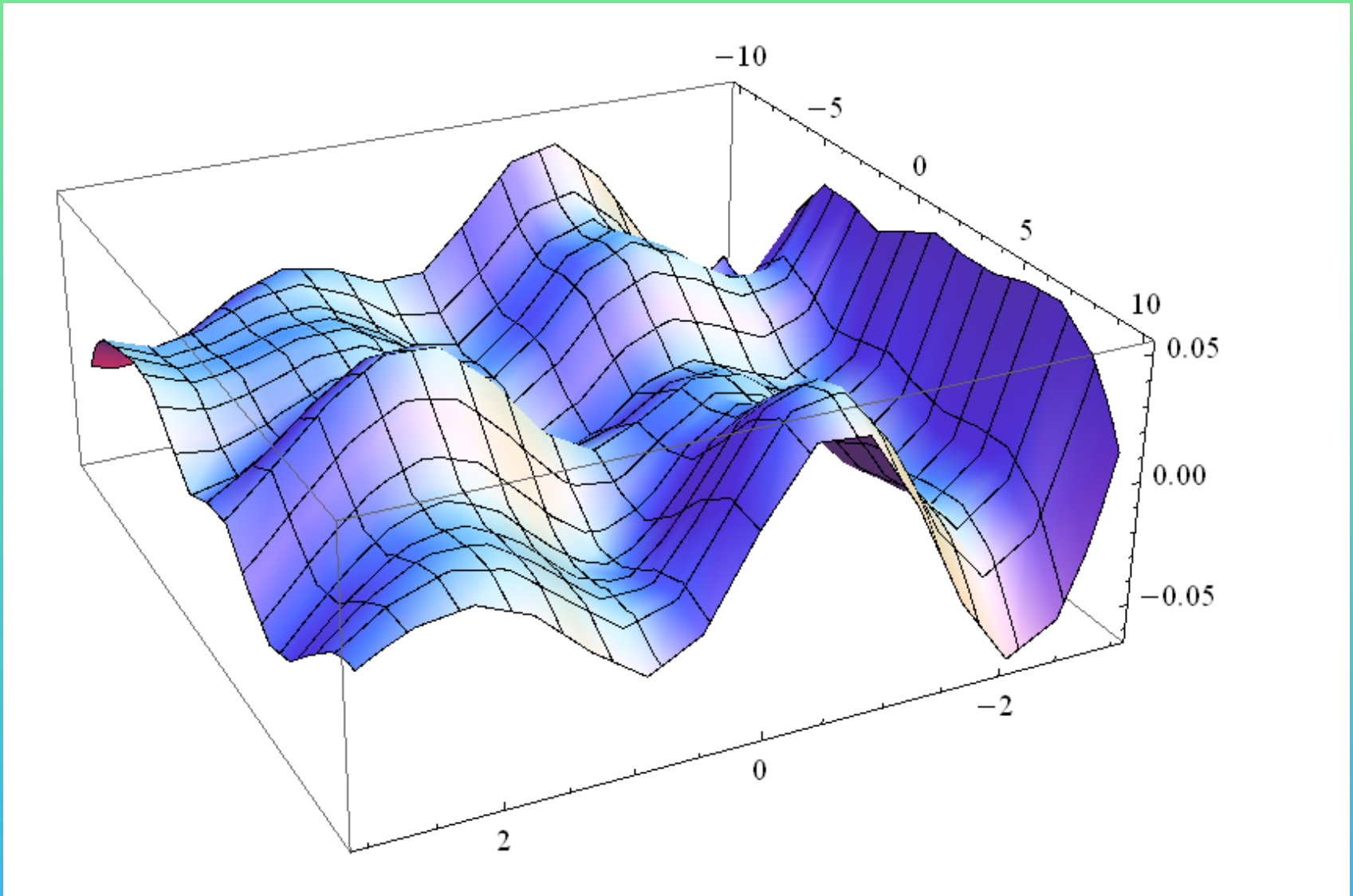
Przykładowy fit całki Fermiego-Diraca siecią neuronową

Błąd absolutny logarytmu dziesiętnego



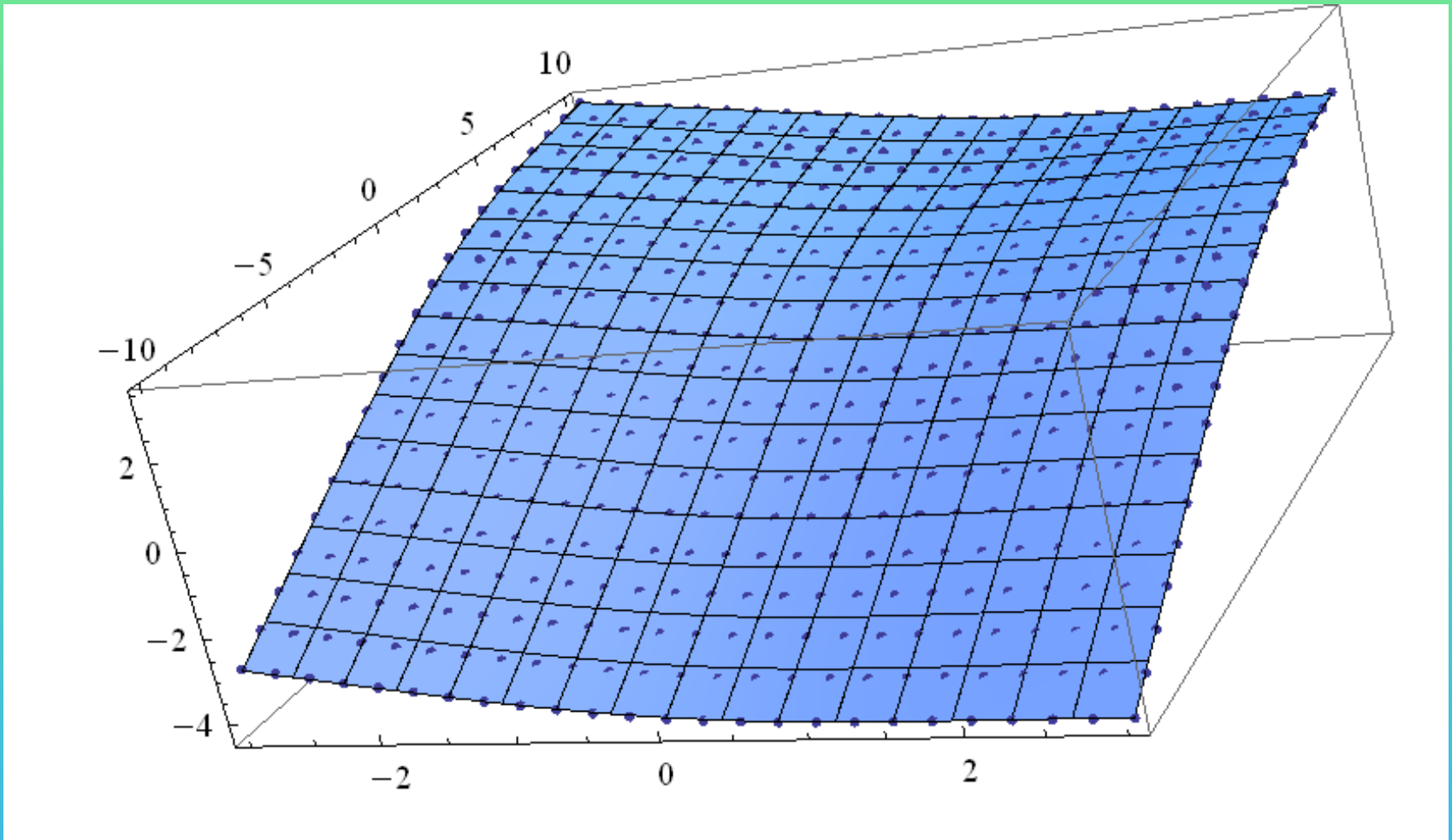
Przykładowy fit całki Fermiego-Diraca siecią neuronową

Błąd względny



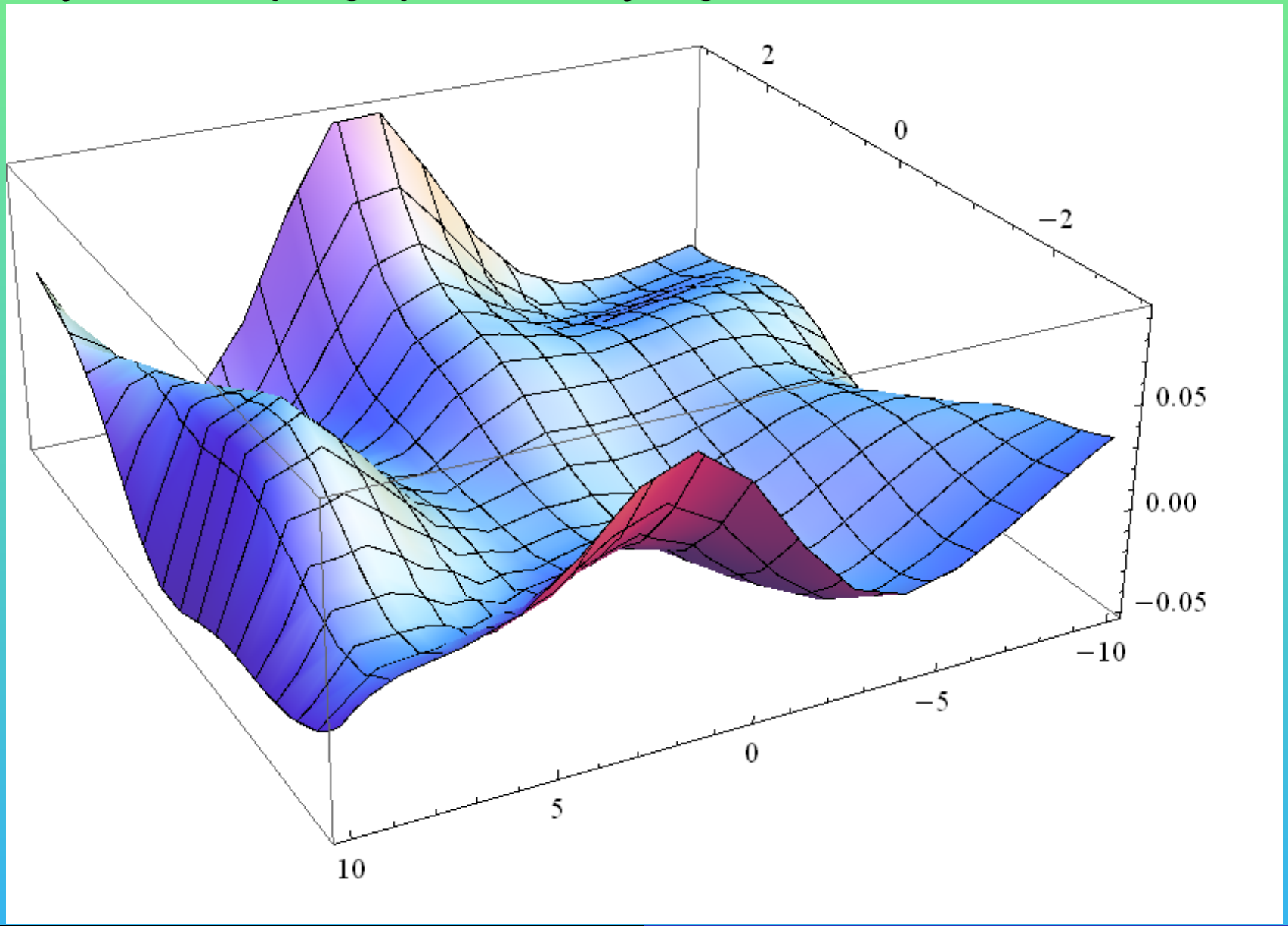
Przykładowy fit całki Fermiego-Diraca funkcją wymierną

Dopasowanie „zbioru uczącego” i funkcji wymiernej (23 parametry)



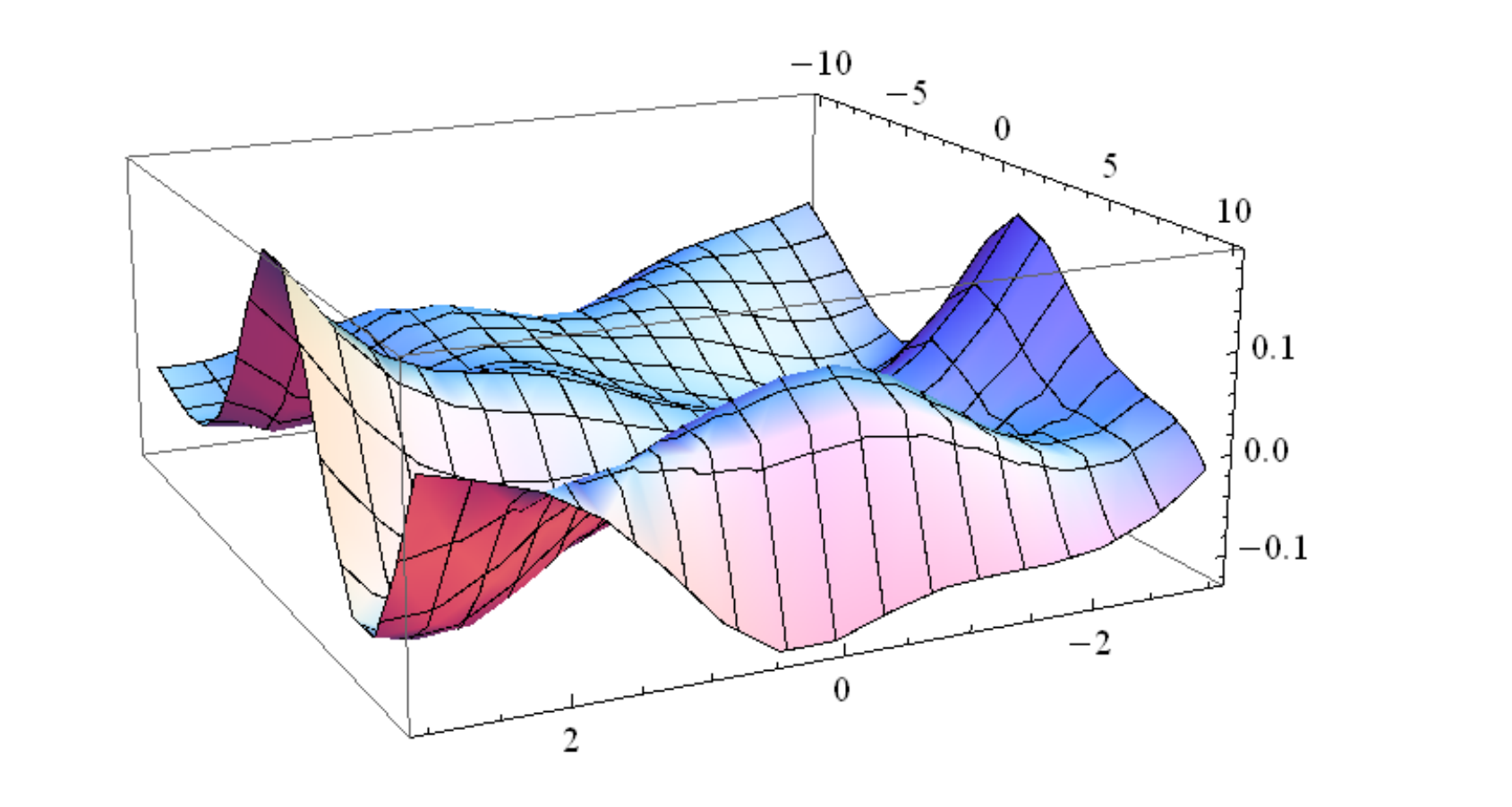
Przykładowy fit całki Fermiego-Diraca funkcją wymierną

Błąd absolutny logarytmu dziesiętnego



Przykładowy fit całki Fermiego-Diraca funkcją wymierną

Błąd względny



Uzasadniony entuzjazm?

Wątpliwości co do znaczenia sieci neuronowych

1. Czy dopasowanie funkcji o 23 parametrach jest sukcesem? *Praktyka w fizyce doświadczalnej i teoretycznej mówi, że w takiej sytuacji możemy dopasować cokolwiek do każdych danych.*

	REL. ERR.	ABS. ERR.	$\int (f - \hat{f})^2$	x2
Neural	5%	0.02	0.0007	50-1500%
Rational	10%	0.05	0.05	10^4%
Polynomial	20%	0.05	0.09	10^9 %

2. przedłużanie poza zakres stosowalności sieci neuronowej nie jest katastrofalne w skutkach, ale nie może równać się z np. szeregami asymptotycznymi
3. trudno odtwarzać funkcje okresowe

Czy istnieje lepsza metoda aproksymacji „dowolnych" funkcji wielu zmiennych?

UWAGA! UWAGA! UWAGA! UWAGA! UWAGA! UWAGA!

Rozważania w tej części mają charakter spekulacyjny!

Nie jest to jednak rzecz całkowicie nieznana. Istnieja podobne techniki okreslane jako **Symbolic Regression** oraz **Gene Expression Programing**.

Nieliniowośc i składanie funkcji dwóch zmiennych

Twierdzenie A. N. Gorbana

Każda ciągła funkcja wielu zmiennych może być aproksymowana z dowolną dokładnością za pomocą dodawania, mnożenia przez liczbę i składania **jednej** *dowolnie wybranej* ciągłej **nieliniowej** funkcji jednej zmiennej.

Jeżeli za tę nieliniową funkcję weźmiemy e^x, to wspomniany wcześniej „dociekliwy student" zatryumfuje!

Obserwacje: istotne elementy układanki

1. pierwszym istotnym elementem jest jakakolwiek nieliniowość
2. drugim istotnym elementem jest funkcja dwóch zmiennych $+(x, y)$ (dodawanie)
3. składanie funkcji
4. mnożenie przez liczbę (w tym liczbę przestępną)

Proponowany kierunek poszukiwań

Hipoteza

Każdą funkcję wielu zmiennych można aproksymować [przedstawić?] jako złożenie pewnej skończonej [konkretnej?] liczby nieliniowych funkcji **dwóch** zmiennych.

Szkic ewentualnego dowodu

Bierzemy **cztery** funkcje dwóch zmiennych:

$$A(x,y) = e^x + \ln|y| - 1, P(x,y) = x + y, M(x,y) = x - y, I(x,y) = 1.$$

Poprzez ich składanie możemy uzyskać: eksponentę, mnożenie, dzielenie, mnożenie przez -1, odejmowanie, potęgowanie, dowolne stałe całkowite i wymierne (w tym 0) oraz mnożenie przez dowolną stałą wymierną. Brakuje *mnożenia przez liczbę przestępną*, np. nie uzyskamy nigdy $f(x,y) = \pi$! Poza tym jednym odtwarzamy wszystkie założenia twierdzenia Gorbana; nieliniowość pochodzi od e^x.

Przykłady

Reverse Polish Notation (RPN)

$yzA \equiv A(y,z)$,
$xxxxxBBBB \equiv B(B(B(B(x,x),x),x),x) = x+x+x+x+x = 5x$

$$A(x,y) = e^x + \ln|y| - 1, P(x,y) = x+y, M(x,y) = x-y, I(x,y) = 1$$

- $1 = XXI = XYI = YYI = YXI$
- $0 = 1 - 1 = XYIYYIM$
- $3 = 1 + 1 + 1 = XXIXXIXXIPP$
- $e^x = A(x,1) + 1 = XXIXAXXIP$
- $\ln x = A(0,x) = XYIYYIMXA$
- $\frac{3}{2} = e^{\ln 3 - \ln 2} =$
  $XXIXYIYYIMXXIXXIXXIPPAXYIYYIMXXIXXIPPAMAXXIP$

Dobór funkcji i optymalna sekwencja ich składania

Załóżmy, że hipoteza o aproksymacji składaniem nieliniowych funkcji 2 zmiennych jest prawdziwa. Ewentualne zastosowanie będzie wymagało rozwiązania dwóch problemów stojących na drodze:

Dobór i ilość funkcji

Problem I: Ile i jakie funkcje nieliniowe dwóch zmiennych pozwolą na skuteczną, szybkozbieżną aproksymację ?

Optymalna sekwencja składania

Problem II: Jak w zadanym zbiorze funkcji 2 zmiennych znaleźć optymalną sekwencję aproksymującą?

Dobór i ilość funkcji

Dobór i ilość funkcji

Problem I: Ile i jakie funkcje nieliniowe dwóch zmiennych pozwolą na skuteczną, szybkozbieżną aproksymację ?

Rozwiązanie (?)

Problem na dzień dzisiejszy wygląda na nierozwiązywalny. Przyjmuję więc roboczo, że funkcje $A(x, y), B, \ldots$ będą utworzone z „typowych" nieliniowych funkcji elementarnych i specjalnych. `Może się jednak okazać, że nie mogą to być ,,byle jakie'' funkcje ale funkcje o pewnych ściśle określonych własnościach, lub nawet konkretne funkcje`

Optymalna sekwencja składania funkcji

Optymalna sekwencja składania

Problem jest wręcz idealnie stworzony aby rozwiązać go algorytmami genetycznymi ponieważ:

1. sekwencja zmiennych i funkcji w Odwrotnej Notacji Polskiej (RPN) wygląda mniej więcej tak:

   XXIXYIYYIMXXIXXIXXIPPAXYIYYIMXXIXXIPPAMAXXIP

   gdzie zapis $XYA \equiv A(X, Y)$. Taki ciąg może podlegać mutacjom.

2. poza typowymi w algorytmach genetycznych mutacjami losowymi i krzyżowaniem, w sekwencjach jak powyżej występują dodatkowo mutacje zmiany fazy odczytu (zaczynamy np. od drugiej pozycji) i tzw. mutacje nonsensowne (kończymy zbyt wcześnie lub wydłużamy sekwencje), czyli dokładnie tak jak w translacji DNA

3. występuje „śmieciowy" DNA np: w ciągu PMXXAPM... „translacja" może zacząć się najwcześniej od XXAPM..., oraz redundancja np: $1 = XXI = XYI = YYI$

Algorytm selekcji

- można użyć metody najmniejszych kwadratów w pewnym obszarze do selekcji
- porządane byłoby jednak włączenie dodatkowych kryteriów, jak np. poprawne lub przynajmniej słabo patologiczne zachowanie się w ∞
- podany schemat miał na celu jak najdokładniejsze odtworzenie mechanizmów mutacji DNA i jego translacji w białka; w szczególności stałe liczbowe tworzone są poprzez składanie funkcji co spowalnia ich optymalizację
- celowe może okazać się dopuszczenie stałych parametrów które będą dopasowywane zwykłymi algorytmami minimalizacji

W pewnych problemach, np. kinetyce, porządane byłoby także dopuszczenie dynamicznej zmiany nie tylko funkcji, ale również ilości zmiennych.

KONIEC

Stanisław Ulam

Ask not what mathematics can do for biology, but what biology can do for mathematics.

Literatura (1)

1. G.G. Lorentz „Approximation of functions", Sect. 11, *Representation of Functions of Several Variables by Functions of One Variable*, 1. The Theorem of Kolmogorov
2. H.L. Frisch et. al., „Approximate Representation of Functions of Several Variables in Terms of Functions of One Variable", Phys. Rev. Lett., Vol. 63, Number 9, 1989.
3. S. Ja. Khavinson, „On interpolation properties of exterior functions in Kolmogorov superpositions", Mathematical Notes, Vol. 55, No. 4, 1994.
4. Manuela Nees, „Chebyshev approximation by discrete superposiotions. Application to neural networks.", Advances in Computational Mathematics **5** (1996) 137-151.
5. A. N. Gorban, „Approximation of Continuous Functions of Several Variables by an Arbitrary Nonlinear Continuous Function of One Variable, Linear Functions and Their Superpositions", Appl. Math. Lett. Vol. 11, No. 3, 45-49, 1998.

Literatura (2)

6. T. Yamakawa and V. Kreinovich, „Why Fundamental Physics Equations Are of Second Order", International Journal of Theoretical Physics, Vol. 38, No. 6, 1763-1769 (1999).
7. G. Li, C. Rosenthal, H. Rabitz, „High Dimensional Model Representations", The Journal of Physical Chemistry A, Vo. 105, Number 33, 7765-7777 (2001)
8. W. Arnold, seria prac w j. rosyjskim, „On the representability of functions of two variables in the form $\chi(\phi(x) + \psi(y))$". Uspehy Math. Nauk 1957, 12:2, 119-121
   „On the functions of three variables." Doklady AN USSR, 1957, 114:4, 679-681.
   „On the representation of continuous functions of three variables by the superpositions of continuous functions of two variables." Matem. Sbornik, 1959, 48:1, 3-74 and 1962, 56:3, 392.
   http://www.pdmi.ras.ru/~arnsem/Arnold/arn-papers.html
9. Cybenko, G.V. (1989). Approximation by Superpositions of a Sigmoidal function, Mathematics of Control, Signals and Systems, vol. 2 no. 4 pp. 303-314. [PDF]