Pierwsze zadanie z kolokwium RPiS - sprawozdanie

Kacper Bajkiewicz

kwiecień 2020

1 Przygotowania do obliczania Φ(t)

Najpierw chciałbym udowodnić lemat, który pozwoli później uzależnić wartość funkcji Φ (t) od funkcji G(t). Poprzez $f(x)$ oznaczamy funkcje równa gestości standardowego rozkładu normalnego a Φ(t) dane jest wzorem: $\int_{-\infty}^{t} f(x)dx$. Pokażmy, że f(x) jest parzysta.

$$f(-x) = \frac{1}{\sqrt{2\pi}} exp(-\frac{(-x)^2}{2}) = \frac{1}{\sqrt{2\pi}} exp(-\frac{x^2}{2}) = f(x) \tag{1}$$

Pokażmy, że Φ(t) = 1 - Φ(-t).

$$1 - \Phi(-t) = \int_{-\infty}^{+\infty} f(x)dx - \int_{-\infty}^{-t} f(x)dx = \int_{-t}^{+\infty} f(x)dx = \int_{-\infty}^{-t} f(x)dx = \tag{2}$$

$$\int_{-t}^{-\infty} f(-x)dx = \int_{\infty}^{-t} f(-x)dx = \int_{-\infty}^{t} f(x)dx = \Phi(t) \tag{3}$$

2 Obliczanie G(t)

Niech G(t) dane bedzie wzorem: $\int_0^t exp(-\frac{(-x)^2}{2})$. Wiadomo, że całki tej nie da sie przedstawić za pomoca funkcji elementarnych, wobec czego, do odpowiedniego przybliżenia jej zastosujemy złożony wzór trapezów wsparty metoda Romberga.

2.1 Złożony wzór trapezów

Wiemy, że wartość całki po przedziale [a,b] można przybliżyć, tworzac swoisty trapez, którego podstawami beda odległości od punktów, przez które przebiega funkcja (dla x = a i x = b) do osi OX układu współrzednych, a którego wysokościami naturalnie stanie sie odcinek miedzy punktami a i b.
Wtedy pole takiego trapezu wyrazimy wzorem $P = \frac{(f(a)+f(b))(b-a)}{2}$.
Jasnym jest, że wykonujac taka operacje na dostatecznie małych podprzedziałach, koniec końców otrzymamy jakkolwiek satysfakcjonujace przybliżenie funkcji na całym przedziale. Niech:

$$\int_a^b f(x)dx \approx \frac{1}{2}\sum_{i=1}^{n}(x_i - x_{i-1})(f(x_i) - f(x_{i-1}) \tag{4}$$

Wtedy, zakładajac, że $x_i = a + (\frac{i(b-a)}{n})$, to $x_i - x_{i-1} = \frac{b-a}{n}$. Z czego dostajemy, że

$$\frac{1}{2}\sum_{i=1}^{n}(x_i - x_{i-1})(f(x_i) - f(x_{i-1}) = \frac{b-a}{2n}\sum_{i=1}^{n} f(x_i) - f(x_{i-1}) \tag{5}$$

Idac za równaniem wyżej można spotrzec, że równe jest to:

$$\frac{b-a}{n}\sum_{i=1}^{n}{}''f(x_i)), \tag{6}$$

a $''$ znaczy, że pierwszy i drugi wyraz tej sumy powinno podzielić sie przez 2 (bo je sumujemy tylko jednokrotnie w sumie (5)).

2.2 Metoda Romberga

Podzielmy nasz obszar całkowania na 2^n przedzialików równej długości (zakładajac, że $n \geq 0$). Wtedy pierwsze przybliżenie bedzie postaci:

$$R(n,0) = \frac{b-a}{2^n}\sum_{i=1}^{n}{}''f(a + i\frac{b-a}{2^n})) \tag{7}$$

Krokiem bedzie wtedy $i\frac{b-a}{2^n}$. Dla R(0,0) nasze równanie stanie sie postaci $\frac{(b-a)(f(a)+f(b))}{2}$.
R(k,0), gdzie k jest naturalne, można wyliczać rekurencyjnie, za pomoca R(k-1,0).
Wtedy:

$$R(n,0) = \frac{1}{2}R(n-1,0) + \frac{b-a}{2^n}\sum_{i=1}^{n}{}''f(a + (2i-1)\frac{b-a}{2^n})). \tag{8}$$

Kolejne przybliżenia dla m $\neq$ 0 otrzymamy zaś ze wzoru:

$$R(n,m) = R(n,m-1) + \frac{1}{4^m - 1}(R(n,m-1) - R(n-1,m-1)) \tag{9}$$

Na podstawie powyższych odkryć udało mi sie napisać algorytm, który oblicza metoda Romberga przybliżenia całki na przedziale. Algorytm zwraca wynik, kiedy kolejne wyniki beda różniły sie o wartość mniejsza niż bład podany w argumencie wywołania funkcji *rombergapprox*.

```
def romberg_approx(measur_err, function, a, b):
    def calculate_romberg(n, m):
        if m != 0:
            return romberg_tab[n][m-1] + ((romberg_tab[n][m-1] - romberg_tab[n-1][m-1]) / ((4 ** m) - 1))
        else:
            temp_sum = 0
            for i in range(1, 2**(n-1)+1):
                temp_sum += function(a + (2*i - 1)*h)
            return (romberg_tab[n-1][0] / 2) + h * temp_sum
    romberg_tab = []
    h = b - a
    romberg_tab.append([h*(function(a) + function(b)) / 2]) #dodajemy pierwszy wyraz R(0,0)
    k = 1
    while True:
        h = h / 2
        romberg_tab.append([]) #dodajemy kolejny wiersz naszemu Rombergowi R(k, 0)
        for i in range(k+1):
            romberg_tab[k].append(calculate_romberg(k, i))
            if i == 0:
                if abs(romberg_tab[k][0] - romberg_tab[k-1][k-2]) < measur_err:
                    return romberg_tab[k][0]
            elif abs(romberg_tab[k][i] - romberg_tab[k][i-1]) < measur_err:
                return romberg_tab[k][i]
        k += 1
```

3 No i co z tym G(t)?

Teraz komputer wyrecza nas w policzeniu G(t). Ponadto, bardzo prosto można uzależnić wynik funkcji $\Phi(t)$ od G(t) właśnie. Gdy uda sie już znaleźć zwiazek miedzy nimi, łatwo bedzie dostać wynik funkcji $\Phi(t)$. Wyprowadźmy wiec te zależność:

$$\Phi(t) = \int_{-\infty}^{0} f(x)dx + \int_{0}^{t} f(x)dx = \Phi(0) + \frac{1}{\sqrt{2\pi}}G(t) = \frac{1}{2} + \frac{1}{\sqrt{2\pi}}G(t). \tag{10}$$

Żeby uczciwie skończyć, należy jeszcze wesprzeć wyprowadzenie powyżej o udowodnienie malutkiego lemaciku, że $\Phi(0) = \frac{1}{2}$

$$\Phi(0) = 1 - \Phi(-0) = 1 - \Phi(0) \tag{11}$$

Czyli, z prostych rachunków.

$$2\Phi(0) = 1 \rightarrow \Phi(0) = \frac{1}{2}. \tag{12}$$

Posługujac sie wiec po raz kolejny kodem pythonowym, tak wygladać bedzie program, który obliczy wartość funkcji $\Phi(t)$

```
def calculate_phit(t):
    def G(x):
        return exp(-(x*x)/2)
    return 1/2 + romberg_approx(0.0001, G, 0, t) / sqrt(2 * pi)
```

4 Słowem zakończenia

Na poczatku chce bardzo przeprosić, że tekst sprawozdania zajał 3.5 strony, ale to mój pierwszy dokument w LaTeXu i po prostu nie udało mi sie zmienić rozmiaru marginesów bez wysłania obrazka w kosmos.
Jeśli chodzi o źródła, to korzystałem z podrecznika Davida Cincaida o tytule Analiza Numeryczna i stron na polskiej i angielskiej Wikipedii (tych o metodzie Romberga).

Kacper Bajkiewicz