Rozdział 6

Obliczanie całek. Kwadratury

W tym rozdziale zajmiemy się zadaniem obliczenia przybliżonego całek postaci:

$$\int_a^b f(t)\,dt,$$

dla funkcji f, czy ogólniej:

$$\int_a^b f(t)\rho(t)\,dt,$$

dla ρ danej wagi.

6.1 Funkcja octave'a quad()

Do obliczania przybliżonego całek jednowymiarowych tzn. typu

$$c = \int_a^b f(t)\,dt$$

służy funkcja octave'a:

quad ( )

Jej najprostsze wywołanie to:

c=**quad**(FCN, a , b) ,

gdzie a, b to początek i koniec odcinka, a FCN to wskaźnik (uchwyt) do funkcji octave'a postaci:

function y=f (x)

#komendy octave'a

y = ... ;

endfunction

Jeśli funkcja, której całkę chcemy obliczyć jest już wcześniej zdefiniowana, to uchwyt do niej zwraca operator octave'a: @, np. uchwyt do funkcji sin zwróci komenda: @sin.

Na początek kilka prostych przykładów całek, które sami potrafimy obliczyć analitycznie:

- całka z $\sin(x)$ na $[-1, 1]$

  quad(@sin , −1 , 1)

  Otrzymaliśmy zero - jak tego oczekiwaliśmy.

- $\int_0^{\pi} \sin(x)\, dx = 2$

  quad(@sin , 0 , **pi**)

  Otrzymaliśmy wynik zgodny z oczekiwaniami.

- całka z funkcji mało regularnej $\int_0^1 \sqrt{t}\, dt$:

  c=**quad**(@sqrt , 0 , 1)

  Sprawdźmy c−2/3, otrzymaliśmy $ans = 2.2204e-16$, czyli zero na poziomie błędu zaokrągleń.

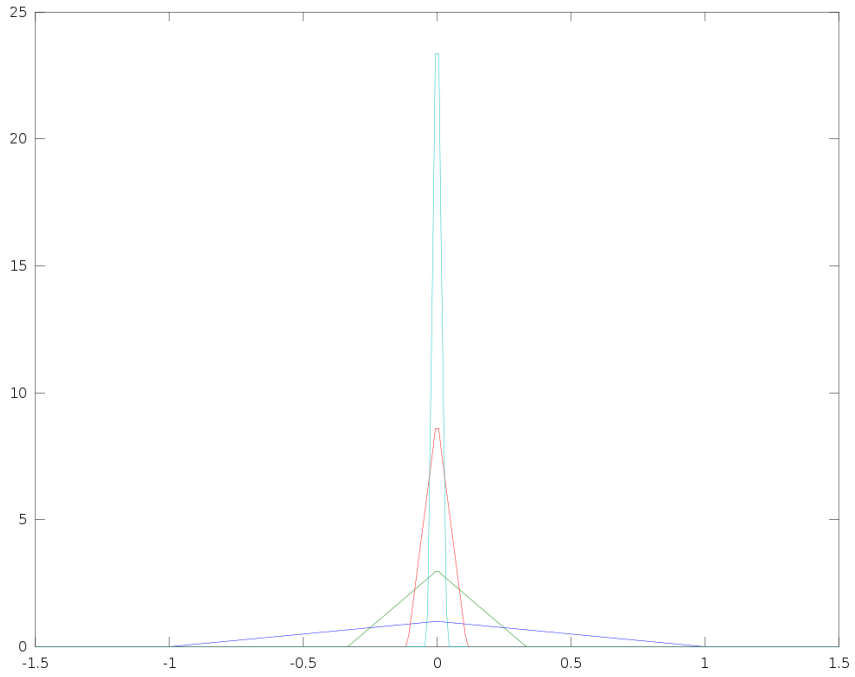
Czy funkcja **quad**() zawsze daje dobre wyniki? Rozpatrzmy prosty przykład całki z funkcji z parametrem

$$f_\epsilon(t) = \epsilon^{-1} f1(\frac{t}{\epsilon})$$

na $[-1, 1]$, por. rysunek 6.1, dla

$$f1(t) = \begin{cases} 1 - |t| & x \in (-1, 1) \\ 0 & \text{w przeciwnym przypadku} \end{cases}$$

Całka powinna być zawsze równa jeden. Zdefiniujemy najpierw funkcję $f1$:



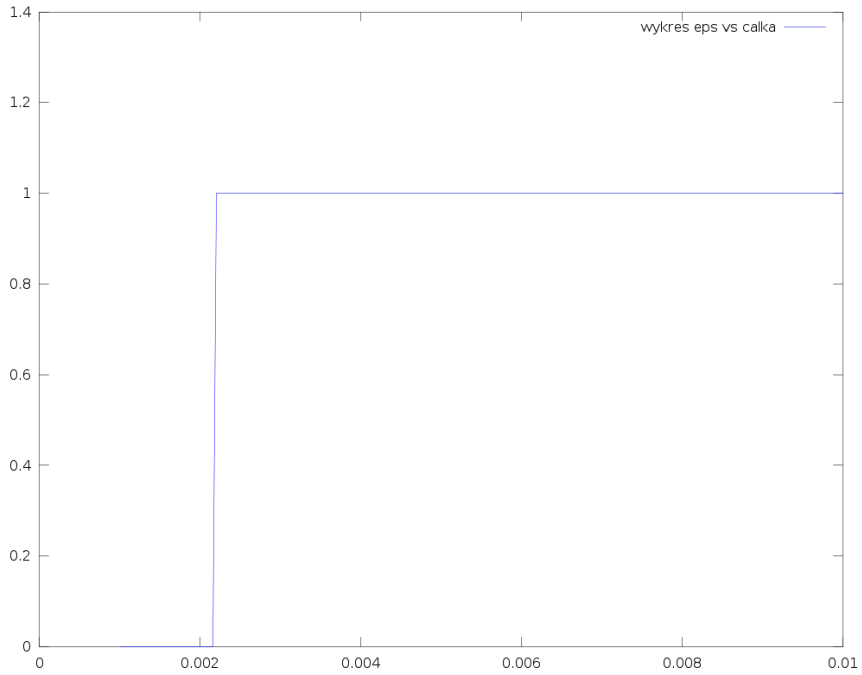
Rysunek 6.1: Wykresy funkcji *daszek*, tzn. f_ϵ dla kilku wartości parametru ϵ.

```
function y=f1(x)
 y=1-abs(x);
 y=y.*(y>0);
endfunction
c=quad(@f1 ,-1,1)
```

Otrzymaliśmy jeden. Powtórzmy obliczenia dla $\epsilon = 10^{-k}$ dla $k = 1, 3, 5, 7$:

```
M=6;
c=epsi=zeros(M+1,1);
epsi(1)=1;
for k=1:M,
   c(k)=quad(@(x) epsi(k)*f1(epsi(k)*x) ,-1,1);
   epsi(k+1)=epsi(k)*10;
endfor
```

Od pewnego momentu obliczone przybliżenia całek są równe zero zamiast jeden.



Rysunek 6.2: Wykres $\epsilon \mapsto \int_{-1}^{1} f_\epsilon(t)\, dt$ otrzymany za pomocą wywołania **quad**().

Wychwyćmy ten moment dokładniej. Powtórzmy obliczenia dla ϵ z $[1e-3, 1e-2]$.

```
epsi=linspace(100,1000,300);
M=length(epsi);
c=zeros(M,1);
for k=1:M,
  c(k)=quad(@(x) epsi(k)*f1(epsi(k)*x) ,-1,1);
endfor
plot(epsi,c)
```

Jest to dość gwałtowna zmiana, por. rysunek 6.2.

Dlaczego tak jest? Oczywiście funkcja octave'a **quad**(() może obliczyć przybliżenie całki korzystając z co najwyżej skończonej ilości wartości funkcji na odcinku całkowania. Nośnik funkcji f_ϵ jest bardzo mały dla $\epsilon \approx 1e-3$ względem odcinka całkowania $[-1, 1]$, stąd być może wszystkie obliczone wartości były równe zero.

Można się domyśleć, że funkcje octave'a działają na zasadzie *czarnej skrzynki*: podajemy parametry do funkcji, a dana funkcja, w tym przypadku **quad**() powinna zwrócić przybliżenie całki.

Spróbujmy obliczyć tę całkę dzieląc odcinek $[-1, 1]$ na małe pododcinki i całkując po nich tzn.

$$\int_a^b f(t)\,dt = \sum_{k=0}^{N-1} \int_{x_k}^{x_k+h} f(t)\,dt \qquad h = (b-a)/N; \quad x_k = a + k * h,$$

a każdą całkę $\int_{x_k}^{x_k+h} f(t)\,dt$ możemy obliczyć przy pomocy funkcji octave'a **quad**().

```
epsi=1000;
N=100;
h=2/N;
xk=-1;
g=@(x)  epsi*f1(epsi*x);
c=0;
for  k=0:(N-1),
   c+=quad(g  ,xk,xk+h);
   xk+=h;
endfor
abs(c-1)
```

Otrzymaliśmy błąd $abs(c-1)$ równy **ans** = 6.8505e−12, podczas gdy **abs**(**quad**(g ,−1,1)−1) zwraca błąd równy jeden.

Oczywiście znając nośnik funkcji możemy policzyć całkę po jej nośniku. Otrzymamy:

```
epsi=1000;
g=@(x)  epsi*f1(epsi*x);
c=quad(g  ,-1e-3,1e-3)
abs(c-1)
```

Błąd jest równy **ans** = 1.1102e−16, czyli jest na poziomie błędu zaokrągleń.

Również dla funkcji o dużej zmienności, np. silnie oscylujących, funkcja octave'a **quad**(() może zwrócić zły wynik.

Spróbujmy scałkować np. $\sin(999 * x)$ na odcinku $[0, \pi]$ i porównajmy z dokładnym wynikiem $\int_0^\pi \sin(999 * x)dt = 1e - 3 * (1 - \cos(999 * \pi))$.

```
a=999;
g=@(x)  sin(a*x);
c=quad(g  ,0,pi)
abs(c-(1/a)*(1-cos(a*pi)))
```

Tu funkcja **quad**() wydrukowała na ekranie ostrzeżenie:

ABNORMAL RETURN FROM DQAGP

Warto wspomnieć, że funkcja **quad**() oblicza również całki po odcinkach nieograniczonych. Za wartości a, b w wywołaniu **quad**(f,a,b) możemy przyjąć inf lub odpowiednio −inf, np. możemy scałkować funkcję $\exp(\frac{x^2}{2})$ po całej prostej rzeczywistej, czy jakiejś półprostej:

```
g=@(x) exp(-0.5*x*x);
c1=quad(g ,-inf ,0)
c2=quad(g,0 , inf)
c3=quad(g,-inf , inf)
c3-c1-c2
```

Czy wynik jest zgodny z oczekiwaniami?

6.2 Kwadratury złożone

Oczywiście w octave'ie możemy zaimplementować najprostsze kwadratury złożone, np. najprostszą kwadraturę złożoną:

$$\int_a^b f(t)\,dtP_N(f) \approx \frac{b-a}{N}\sum_{k=1}^{N} f(a+k*h)$$

Implementacja w octave'ie tej kwadratury to:

```
function c=prostakw(f,a,b,N=300)
 #kwadratura prostokatow prawostronna
 #Input: a,b, konce przedzialu calkowania
 #N- ilosc punktow - domyslnie 100
 #f- wskaznik do funkcji -
 #      zwracajacej wektor wartosci dla wektora danych
 #Output: c - przyblizenie calki
 h=(b-a)/N;
 x=(1:N)*h;
 y=f(x);
 c=h*sum(y);
endfunction
```

Przetestujmy tę kwadraturę. Na początku na jednomianach:

```
prostakw(@(x) x,0,1)-0.5
prostakw(@(x) x.*x,0,1)-1/3
```

Wynik jest poprawny. Jeżeli zwiększymy N, to być może uzyskamy lepszy wynik:

```
prostakw(@(x) x,0,1,1000)-0.5
prostakw(@(x) x.*x,0,1,1000)-1/3
```

Błąd jest na poziomie 10^{-4}.

Dla funkcji klasy $C^1([a,b])$ błąd można oszacować przez

$$|\int_a^b f(t)\,dt - P_N f| \leq \|f'\|_{\infty,[a,b]} h = O(N^{-1}),$$

przy czym dla funkcji $f(x) = x$ w tym oszacowaniu widzimy równość.

Można pokazać, że dla dostatecznie gładkiej funkcji zachodzi:

$$\int_a^b f(t)\,dt - P_N f = -f'(\xi)h$$

dla pewnego punktu $\xi \in (a,b)$, a dla bardziej regularnych funkcji

$$\int_a^b f(t)\,dt - P_N f = Ch + O(h^2)$$

dla pewnej ujemnej stałej C.

N	h	E_N	$E_N/E_{N/2}$
10	1.00e-01	5.17e-02	0.00000
20	5.00e-02	2.54e-02	2.03279
40	2.50e-02	1.26e-02	2.01653
80	1.25e-02	6.28e-03	2.00830
160	6.25e-03	3.13e-03	2.00416
320	3.13e-03	1.56e-03	2.00208
640	1.56e-03	7.82e-04	2.00104
1280	7.81e-04	3.91e-04	2.00052

Tablica 6.1: badanie rzędu kwadratury $P_N f$ dla całki z funkcji $f(x) = x^2$ na $[0,1]$. Błąd $E_N = |\int_0^1 f - P_N f|$.

Przetestujmy teraz tę kwadraturę dla ustalonej analitycznej funkcji f i podwajanych wartości N, tzn. obliczymy $P_N f$ dla $N = N_0, 2 * N_0, \ldots$ i następnie policzymy:

$$\frac{E_k}{E_{k+1}} = \frac{C * h + O(h^2)}{C * h/2 + O(h^2)} \approx 2,$$

dla $E_k = |\int_a^b f(t)\,dt - P_{2^k N_0} f|$.

N	h	E_N	$E_N/E_{N/2}$
10	3.14e-01	1.65e-02	
20	1.57e-01	4.11e-03	4.00495
40	7.85e-02	1.03e-03	4.00123
80	3.93e-02	2.57e-04	4.00031
160	1.96e-02	6.43e-05	4.00008
320	9.82e-03	1.61e-05	4.00002
640	4.91e-03	4.02e-06	4.00000
1280	2.45e-03	1.00e-06	4.00000

Tablica 6.2: badanie rzędu kwadratury $P_N f$ dla całki z funkcji $f(x) = \sin(x)$ na $[0, \pi]$. Błąd $E_N = |\int_0^\pi f - P_N f|$.

```
N=10; M=8;
#c=2; #znana wartosc calki
e=0;
for k=1:M,
 kw=prostakw(f,a,b,N);
 ep=e;
 e=abs(c-kw);
 printf("[%d] e=%g ep/e=%6.5f  \n",N,e,ep/e);
 N*=2;
endfor
```

Wyniki dla funkcji $f(x) = x^2$ i odcinka $[0, 1]$ są zawarte w Tabeli 6.1, a dla funkcji $\sin(x)$ i całki z odcinka $[0, \pi]$ w Tabeli 6.2. Dlaczego dla $\sin(x)$ wyniki wskazują na błąd $E_N \approx O(h^2)$?

N	h	E_N	$E_N/E_{N/2}$
10	1.00e-01	4.17e-02	
20	5.00e-02	2.09e-02	1.99085
40	2.50e-02	1.05e-02	1.99544
80	1.25e-02	5.25e-03	1.99772
160	6.25e-03	2.63e-03	1.99886
320	3.13e-03	1.31e-03	1.99943
640	1.56e-03	6.57e-04	1.99972
1280	7.81e-04	3.29e-04	1.99986

Tablica 6.3: badanie rzędu kwadratury $P_N f$ dla całki z funkcji $f(x) = \sin(x)$ na $[0, 1]$. Błąd $E_N = |\int_0^1 f - P_N f|$.

Zmieńmy odcinek na $[0,1]$ i wyniki będą zgodne z oczekiwanymi, por. Tabela 6.3.

6.3 Normy całkowe

Zastanówmy się, jak w sposób przybliżony wyznaczyć normę typu L^p na odcinku $[a,b]$ dla zadanej funkcji $f(x)$ zdefiniowanej w octave'ie jako

```
function y=f(x)
```

Spróbujmy zdefiniować funkcję octave'a, która korzystając z **quad**() wyznaczy przybliżenie normy $\|f\|_{L^p(a,b)} = \left(\int_a^b |f(t)|^p\,dt\right)^{1/p}$. Jako parametry tej funkcji chcemy podawać wskaźnik do funkcji typu y=f(x), p, oraz końce odcinka $[a,b]$. Żeby wywołać funkcję **quad**() musimy podać funkcję octave'a jednoargumentową $x \mapsto |f(x)|^p$ jako pierwszy argument. Trzeba ją odpowiednio zdefiniować wewnątrz naszej funkcji. Najwygodniejszym sposobem jest użycie mechanizmu funkcji anonimowej, por Rozdział 3.2.1:

```
fp=@(x) (abs(f(x))^p;
```

oczywiście p musi być wcześniej zdefiniowane. Tak więc można by napisać tę funkcję następująco:

```
function n=Lpnorm(f,a,b,p=2)
  fp=@(x) abs(f(x))^p;
  n=quad(fp,a,b);
  n=n^(1/p);
endfunction
```

Sprawdźmy dla $p=2$ i prostych funkcji np.:

```
Lpnorm(@(x) 1,0,1)-1
Lpnorm(@(x) x,0,1)-1/sqrt(3)
Lpnorm(@cos,0,pi)-sqrt(pi/2)
```

Wyniki są poprawne.

Mając zaimplementowaną taką funkcję, możemy np. przeprowadzić badanie rzędów zbieżności interpolacji splajnowej w normie L^2, zamiast w normie maksimum, tak jak w Rozdziale 5.4.