METODY NUMERYCZNE
DLA INŻYNIERÓW

(notatki do wykładu)

Wrocław, maj 2018

Spis Treści

1. Wstęp

Niniejszy skrypt zawiera opis głównych zagadnień prezentowanych na wykładzie *Metody numeryczne dla inżynierów*, który jest przeznaczony dla studentów kierunku *Automatyka i Robotyka* na Wydziale Elektrycznym Politechniki Wrocławskiej.

Metody numeryczne są podstawowym narzędziem analitycznym w rękach współczesnego inżyniera i stąd też nietrudno znaleźć wyczerpującą literaturę na ten temat o różnym stopniu zaawansowania - niektóre propozycje podane są w końcowej części pracy. Każde jednak ujęcie tego tematu jest przeznaczone dla określonego czytelnika, o odpowiednim stopniu przygotowania i z myślą o specyficznym zastosowaniu prezentowanych metod. Głównym celem niniejszego opracowania jest prezentacja podstawowych metod numerycznych stosowanych w obliczeniach w elektrotechnice.

Zakłada się, że Czytelnik zna podstawowy kurs algebry i analizy matematycznej. Wymagana jest również podstawowa znajomość zasad tworzenia algorytmów obliczeniowych. Wykonanie prezentowanych przykładów obliczeniowych wymaga również elementarnej znajomości korzystania z komputerów.

Z wykładem związane są ćwiczenia laboratoryjne, w trakcie których są praktycznie ilustrowane zagadnienia przedstawiane na wykładzie. Podstawowym narzędziem programowym, stosowanym do opisu poszczególnych procedur obliczeniowych, jak i do obliczeń w laboratorium komputerowym jest MATLAB. Program ten jest stosowany tu zarówno do formułowania i sprawdzania prostych algorytmów numerycznych, jak i do rozwiązywania bardziej złożonych zagadnień z wykorzystaniem gotowych procedur.

Pakiet programowy MATLAB, jak wiele innych tego typu programów przeznaczonych do rozwiązywania zadań inżynierskich, zawiera sporą liczbę gotowych procedur numerycznych, które są dostępne w postaci pojedynczych instrukcji. Można zatem spytać, jaki jest cel dodatkowego wykładu na ten temat, skoro wystarczy się zapoznać z instrukcją obsługi odpowiedniego programu komputerowego. Jednak każdy użytkownik tego typu oprogramowania specjalistycznego natrafia na problemy związane z wyborem odpowiednich procedur (często do rozwiązania tego samego zadania można zastosować różne algorytmy), interpretacji błędów, dokładności wyników, rozwiązywania zagadnień niestandardowych, czy wreszcie rozumienia i interpretacji tekstu instrukcji. Ważna jest także umiejętność formułowania modeli matematycznych analizowanych zjawisk, które pozwalają określić poszukiwane parametry lub zależności między nimi. W takich przypadkach wymagana jest niekiedy pogłębiona znajomość zagadnień analizy numerycznej.

W praktyce inżynierskiej metody numeryczne są narzędziem służącym do formułowania i rozwiązywania praktycznych zagadnień obliczeniowych, a także do przekształcenia znanych modeli ciągłych do adekwatnych postaci dyskretnych. Z tego punktu widzenia metody numeryczne są tu traktowane jako wygodny i wydajny sposób rozwiązywania zadań inżynierskich. Przygotowanie i rozwiązanie takiego typu zadań wiąże się zazwyczaj z wykonaniem następujących działań:

- określenie modelu matematycznego analizowanego zjawiska lub opis stanu obserwowanego systemu;
- wybranie (opracowanie) odpowiedniej metody obliczeń numerycznych;
- analiza i weryfikacja poprawności przyjętego modelu oraz wykonanych obliczeń.

W niniejszym wykładzie będziemy się zajmować głównie drugim z wymienionych działań. Łączy się ono z podaniem sposobu (algorytmu) numerycznego rozwiązania postawionego zadania. W obliczeniach prowadzonych z zastosowaniem metod numerycznych należy się liczyć ze specyfiką stosowanych narzędzi. Liczby reprezentowane w komputerze są przedstawiane z ograniczoną dokładnością, która zależy od liczby bitów użytych do ich zapisu. Wynikające stąd błędy najczęściej nie mają znaczenia w dalszym wykorzystaniu wyników obliczeń. Niekiedy jednak wartość błędów powstających w poszczególnych etapach obliczeń jest tak duża, że kontynuowanie obliczeń staje się niemożliwe (przekroczenie zakresu) lub uzyskane wyniki zawierają niedopuszczalne błędy.

Można wyróżnić następujące cztery źródła błędów, które ograniczają dokładność końcowych wyników:

1. błędy w danych wejściowych
2. przybliżony model zjawiska
3. błędy aproksymacji modelu
4. błędy zaokrągleń

Błędy danych wejściowych leżą poza procesem obliczeń, jednak stosowanie odpowiednich procedur może prowadzić do redukcji ich wpływu na wynik (na przykład, wygładzanie danych pomiarowych). Problem ten łączy się zatem z drugim z wymienionych źródeł błędów. Należy jednak podkreślić, że błędy danych wejściowych, w ogólnym przypadku, są nieusuwalne.

Błędny lub przybliżony model analizowanego procesu wynika z uproszczeń przyjmowanych w trakcie formułowania modelu matematycznego zjawiska lub opisu stanu. Wynika to z potrzeby redukcji złożoności modelu, która jest przyjmowana w sposób świadomy lub z braku odpowiednich danych, tak że analizowany proces jest przedstawiany w sposób uproszczony.

Błąd metody jest związany z tym, że poprawny model jest aproksymowany za pomocą uproszczonych formuł, w których dodatkowo mogą być stosowane przybliżone dane raz parametry. Typowym przykładem tego typu podejścia jest aproksymowanie zależności różniczkowych za pomocą funkcji różnicowych.

Błędy zaokrągleń wynikają ze skończonej długości reprezentacji liczb w komputerze. Jeśli błędy te mają charakter przypadkowy a nie systematyczny, to sumaryczny błąd statystyczny nawet długiej serii obliczeń jest zazwyczaj mały. Systematyczne błędy zaokrągleń mogą jednak prowadzić do szybko rosnącej niedokładności obliczeń. Znanym źródłem takich błędów jest operacja odejmowania bliskich sobie liczb. Jeśli w algorytmie szeregowo powtarzane są takie działania, to szybko następuje niedopuszczalna kumulacja błędów. Typowy przykład jest związany z umieszczeniem takiej różnicy w mianowniku jakiegoś wyrażenia.

Poprawność i efektywność algorytmów obliczeniowych jest określana za pomocą różnych parametrów. Oto niektóre z nich.

Złożoność obliczeniowa algorytmu jest związana z liczbą operacji numerycznych, które prowadzą do uzyskania wyniku. Jest zrozumiałe, że spośród różnych algorytmów, które zapewniają poprawne rozwiązanie, należy wybierać te, które charakteryzują się małą złożonością obliczeniową. Jest to szczególnie istotne w układach sterowania, gdzie pełny cykl obliczeń numerycznych musi być wykonany w czasie określonym przez okres pomiędzy kolejnymi pomiarami wielkości wejściowych.

Uwarunkowanie zadania jest cechą metod numerycznych, która określa możliwość uzyskania poprawnych wyników przy stosowaniu dowolnych danych wejściowych z odpowiednio zdefiniowanego zbioru. Jeśli analizowany algorytm służy do rozwiązania zadania $y = w(x)$, to stopień uwarunkowania zadania można mierzyć za pomocą ilorazu $|w(x+\delta x) - x(x)|/|\delta x|$. Niekiedy używa się też terminu *czułość zadania*. Mówi się, zatem, że zadanie jest *dobrze uwarunkowane* lub *źle uwarunkowane*. W pierwszym przypadku zadanie jest stabilne względem danych wejściowych, co oznacza, że rozwiązanie w sposób ciągły zależy od dokładności danych wejściowych tak, że dla $|\delta x| \to 0$ jest $|\delta y| \to 0$. W przypadku złego uwarunkowania zadania, możliwość uzyskania poprawnego rozwiązania zależy od wartości danych wejściowych. Cecha ta jest wykorzystywana do odpowiedniej korekcji zadań źle uwarunkowanych, które nie mogą być inaczej rozwiązane.

W wielu przypadkach algorytm zastosowany do rozwiązania zadania dobrze uwarunkowanego (stabilnego) może być niestabilny. **Stabilność numeryczna algorytmu** odnosi się do możliwości uzyskania określonej dokładności obliczeń. Algorytm jest stabilny numerycznie, gdy zwiększając dokładność obliczeń można z dowolną dokładnością określić dowolne z istniejących rozwiązań.

2. Liniowe układy równań

2.1. Wprowadzenie

Zagadnienie rozwiązywania układów równań liniowych jest podstawowym problemem w metodach numerycznych. Metod rozwiązywania tego zagadnienia jest wiele, a wybór tej czy innej metody zależy od rodzaju zadania, oczekiwanej dokładności i środków technicznych będących w dyspozycji (szybkość procesora oraz objętość pamięci).

Załóżmy, że mamy układ trzech równań z trzema niewiadomymi:

$$\begin{aligned} 10x_1 - 7x_2 &= 6 \\ -3x_1 + 2x_2 + 6x_3 &= 4 \\ 5x_1 - x_2 + 5x_3 &= 3 \end{aligned} \tag{2.1}$$

Równanie to można zapisać w następującej postaci macierzowej:

$$\begin{bmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 4 \\ 3 \end{bmatrix} \tag{2.2}$$

Przechodząc do postaci ogólnej mamy:

$$\mathbf{Ax} = \mathbf{b} \tag{2.3}$$

gdzie: $\mathbf{A}$ - macierz kwadratowa ($n \times n$); w tym przypadku $n = 3$,

$\mathbf{x}$ - wektor niewiadomych ($n \times 1$),

$\mathbf{b}$ - wektor współczynników prawej strony ($n \times 1$).

Jeśli wyznacznik macierzy $\det(\mathbf{A}) \neq 0$, to rozwiązanie można przedstawić w następującej postaci

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \tag{2.4}$$

Można pokazać, że poszukiwanie rozwiązania równania (2.3) w postaci (2.4) prowadzi do algorytmu o dużej złożoności obliczeniowej, co jest związane z odwracaniem macierzy $\mathbf{A}$. Już zastosowanie reguł 'ręcznego' rozwiązywania układu równań (2.1) redukuje około n razy liczbę niezbędnych mnożeń potrzebnych do uzyskania wyniku. Poniżej przedstawimy niektóre najczęściej stosowane metody rozwiązywania równania (2.3).

2.2. Metoda eliminacji Gaussa

Powyższy przykład z układem trzech równań liniowych można rozwiązać stosując metodę, która jest zbliżona do tradycyjnej metody 'szkolnej'. Polega ona na kolejnej eliminacji zmiennych. Korzysta się przy tym z prostych działań, takich jak: mnożenie obu stron równania przez stałą wartość lub dodawanie równań stronami.

W rozważanym przypadku (2.1), zmienna x_1 może być wyeliminowana z drugiego równania przez odjęcie od niego równania pierwszego pomnożonego przez współczynnik $-3/10 = -0,3$. Podobnie można postąpić z trzecim równaniem: w tym przypadku pierwsze równanie przed odjęciem go od równania trzeciego należy pomnożyć przez współczynnik $5/10 = 0,5$. Po wykonaniu tych operacji otrzymamy następującą postać równania (2.1):

$$\begin{aligned} 10x_1 \quad & -7x_2 \quad && = \quad 6 \\ & -0.1x_2 \quad +6x_3 \quad = \quad 5.8 \\ & 2.5x_2 \quad +5x_3 \quad = \quad 0 \end{aligned} \tag{2.5}$$

które ma następującą formę macierzową:

$$\begin{bmatrix} 10 & -7 & 0 \\ 0 & -0.1 & 6 \\ 0 & 2.5 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5.8 \\ 0 \end{bmatrix} \tag{2.6}$$

Z ostatnich dwóch równań można najpierw określić x_3 przez eliminację zmiennej x_2 z trzeciego równania. Można to uzyskać przez dodanie drugiego równania po jego pomnożeniu przez współczynnik $-2,5/0,1 = 25$. Ostatecznie otrzymamy:

$$\begin{bmatrix} 10 & -7 & 0 \\ 0 & -0.1 & 6 \\ 0 & 0 & 155 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5.8 \\ 145 \end{bmatrix} \tag{2.7}$$

Zauważmy, że z ostatniego równania (ostatni wiersz) można już bezpośrednio określić zmienną x_3. Ten etap obliczeń nazywa się *etapem eliminacji zmiennych*. Poczynając teraz od ostatniego równania (ostatniego wiersza w zapisie macierzowym) można otrzymać kolejne rozwiązania. Jest to *postępowanie odwrotne*. Zatem, w celu uzyskania wartości wszystkich niewiadomych wykonujemy następujące działania:

$$x_3 = \frac{145}{155} = \frac{29}{31}$$

$$x_2 = \frac{1}{-0.1}\left(5.8 - 6\frac{29}{31}\right) = \frac{1}{-0.1}\left(\frac{179.8 - 174}{31}\right) = \frac{-58}{31}$$

$$x_1 = \frac{1}{10}\left(6 - 0\frac{29}{31} - 7\frac{5.8}{3.1}\right) = \frac{1}{10}\frac{18.6 - 40.6}{3.1} = \frac{-22}{31}$$

Powyższe operacje można zapisać dla ogólnego przypadku. W tym celu rozpatrzmy ogólną postać równania (2.3), gdzie:

$$\mathbf{A}=\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix},\ \mathbf{b}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix},\ \mathbf{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2.8}$$

które można zapisać w postaci następującego układu równań

$$\begin{array}{ccccc} a_{11}x_1 & +a_{12}x_2 & +\dots & +a_{1n}x_n & =b_1 \\ a_{21}x_1 & +a_{22}x_2 & +\dots & +a_{2n}x_n & =b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ +a_{n1}x_1 & +a_{n2}x_2 & +\dots & +a_{nn}x_n & =b_n \end{array} \tag{2.9}$$

Stosując pierwszy krok eliminacji w odniesieniu do (2.9) otrzymamy układ równań, w których poczynając od drugiego z nich, wyeliminowana jest zmienna x_1:

$$\begin{array}{ccccc} a_{11}x_1 & +a_{12}x_2 & +\dots & +a_{1n}^{(2)}x_n & =b_1 \\ & +a_{22}^{(2)}x_2 & +\dots & +a_{2n}^{(2)}x_n & =b_2^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ & +a_{n2}^{(2)}x_2 & +\dots & +a_{nn}^{(2)}x_n & =b_n^{(2)} \end{array} \tag{2.10}$$

gdzie:

$$a_{22}^{(2)}=a_{22}-a_{12}\frac{a_{21}}{a_{11}},\ a_{23}^{(2)}=a_{23}-a_{13}\frac{a_{21}}{a_{11}},\ \dots,\ a_{2n}^{(2)}=a_{2n}-a_{1n}\frac{a_{21}}{a_{11}},$$

$$a_{32}^{(2)}=a_{32}-a_{12}\frac{a_{31}}{a_{11}},\ a_{33}^{(2)}=a_{33}-a_{13}\frac{a_{31}}{a_{11}},\ \dots,\ a_{3n}^{(2)}=a_{3n}-a_{1n}\frac{a_{31}}{a_{11}},\ \dots,$$

$$a_{n2}^{(2)}=a_{n2}-a_{12}\frac{a_{n1}}{a_{11}},\ a_{n3}^{(2)}=a_{n3}-a_{13}\frac{a_{n1}}{a_{11}},\ \dots,\ a_{nn}^{(2)}=a_{nn}-a_{1n}\frac{a_{n1}}{a_{11}}$$

oraz

$$b_2^{(2)}=b_2-b_1\frac{a_{21}}{a_{11}},\ b_3^{(2)}=b_3-b_1\frac{a_{31}}{a_{11}},\ \dots,\ b_n^{(2)}=b_n-b_1\frac{a_{n1}}{a_{11}}$$

W ostatnim kroku tej procedury układ równań ma następującą postać:

$$\begin{array}{ccccccc} a_{11}x_1 & +a_{12}x_2 & +\dots & +a_{1n}x_n & a_{1n}x_n & =b_1 \\ & +a_{22}^{(2)}x_2 & +\dots & +a_{2n}^{(2)}x_n & a_{2n}^{(2)}x_n & =b_2^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & +a_{n-1n-1}^{(n-1)}x_2 & +a_{n-1n}^{(n-1)}x_2 & b_{n-1}^{(n-1)} \\ & & & & +a_{nn}^{(n)}x_n & =b_n^{(n)} \end{array} \tag{2.11}$$

gdzie: $a_{nn}^{(n)} = a_{nn}^{(n-1)} - a_{n-1n}^{(n-1)} \frac{a_{nn-1}^{(n-1)}}{a_{n-1n-1}^{(n-1)}}$, $b_n^{(n)} = b_n^{(n-1)} - b_{n-1}^{(n-1)} \frac{a_{nn-1}^{(n-1)}}{a_{n-1n-1}^{(n-1)}}$.

W ten sposób, po wykonaniu procedury eliminacji zmiennych pierwotne równanie przekształca się do postaci z górną trójkątną macierzą $\mathbf{U}$:

$$\mathbf{Ux} = \mathbf{b} \tag{2.12}$$

gdzie:

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & u_{nn} \end{bmatrix}$$

Niewiadomą x_n wyznacza się z równania określonego przez ostatni wiersz:

$$x_n = \frac{b_n}{u_{nn}} \tag{2.13}$$

Dalej, znając niewiadome x_n, x_{n-1}, x_{k+1} z k-tego równania obliczamy:

$$x_k = \frac{b_k - \sum_{j=k+1}^{n} u_{kj} x_j}{u_{kk}} \tag{2.14}$$

przy czym uwzględniane są odpowiednio przekształcone współczynniki wektora $\mathbf{b}$.

Ostatecznie otrzymujemy następujący algorytm rozwiązywania układów równań liniowych metodą eliminacji Gaussa.

{*eliminacja zmiennych*}

for $i := 2$ **to** n **do**

  for $k := i$ **to** n **do**

  begin

$$a_{kl} := \begin{cases} a_{kl} - a_{i-1,l} \dfrac{a_{k,i-1}}{a_{i-1,i-1}} & \text{dla } l = i, i+1, \dots, n \\ 0 & \text{dla } l = 1, 2, \dots, i-1 \end{cases}$$

$$b_k := b_k - b_{i-1} \frac{a_{k,i-1}}{a_{i-1,i-1}}$$

  end;

end;

{*odwrotne podstawianie*}

$$x_n = \frac{b_n}{a_{nn}}$$

for $k := n-1$ **to** 1 **step** -1 **do** $x_k = \dfrac{b_k - \sum_{j=k+1}^{n} a_{kj} x_j}{a_{kk}}$

W powyższym algorytmie zakłada się, że w pierwszym etapie nie jest tworzona nowa macierz $\mathbf{U}$, natomiast tworzona macierz trójkątna jest zapisywana na miejscu macierzy $\mathbf{A}$.

Jak widać, w operacjach arytmetycznych ważną rolę odgrywają elementy leżące na przekątnej macierzy współczynników równania. Przez nie są dzielone odpowiednie równania w pierwszym etapie eliminacji zmiennych. Także w wyniku dzielenia uzyskuje się kolejne rozwiązania na etapie podstawiania zmiennych. Rozwiązanie staje się nieosiągalne, gdy któryś z tych elementów diagonalnych jest równy zero (wówczas macierz parametrów jest osobliwa). Również przy małych wartościach elementów diagonalnych można spodziewać się dużych błędów (gdyż występuje dzielenie przez małą liczbę, która - z racji reprezentacji dyskretnej - może być przedstawiona niedokładnie. Aby tego uniknąć stosuje się modyfikację metody, która polega na tak zwanym *częściowym wyborze elementu wiodącego*. W tym celu, przed eliminacją kolejnej zmiennej (etap wprzód), spośród równań pozostających do rozpatrzenia (poniżej danego wiersza) wybiera się to, które ma w redukowanej kolumnie (w pierwszej niezerowej) największą wartość i zamienia się go z danym równaniem. Odpowiedni algorytm zostanie pokazany w następnym rozdziale.

Optymalne metody rozwiązywania układów równań liniowych powinny przewidywać takie uporządkowanie równania, aby macierz $\mathbf{A}$ była diagonalnie dominującą. Oznacza to, że moduły elementów na przekątnej są nie mniejsze od sumy modułów pozostałych elementów w tym samym wierszu (wówczas jest to macierz diagonalnie dominująca kolumnowo), co można zapisać następująco

$$|a_{ii}| \geq \sum_{\substack{k=1 \\ k \neq i}}^{n} |a_{ki}|, \; i = 1, 2, ..., n$$

2.3. Metoda rozkładu LU

Załóżmy, że kwadratowa macierz współczynników równania $\mathbf{A}$ zostanie przedstawiona w postaci iloczynu dwóch macierzy trójkątnych:

$$\mathbf{A} = \mathbf{LU} \tag{2.15}$$

gdzie:

$$\mathbf{L}=\begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & 0 & \\ \vdots & \vdots & \dots & \vdots & \vdots \\ l_{n-1,1} & l_{n-1,2} & \dots & 1 & \\ l_{n,1} & l_{n,2} & \dots & l_{n,n-1} & 1 \end{bmatrix},\ \mathbf{U}=\begin{bmatrix} u_{11} & u_{12} & \dots & u_{1,n-1} & u_{1n} \\ & u_{22} & \dots & u_{2,n-1} & u_{2n} \\ \vdots & \vdots & \dots & \vdots & \vdots \\ & 0 & \dots & u_{n-1,n-1} & u_{n-1,n} \\ & & \dots & & u_{n,n} \end{bmatrix}$$

Załóżmy, że znane są macierze $\mathbf{L}$, $\mathbf{U}$ dla danej macierzy $\mathbf{A}$. Wówczas równanie (2.15) można zapisać w następującej formie

$$\mathbf{LUx}=\mathbf{b} \tag{2.16}$$

Wektor $\mathbf{x}$ można określić w dwóch etapach, rozwiązując kolejno następujące równania

$$\mathbf{Lz}=\mathbf{b} \tag{2.17}$$

$$\mathbf{Ux}=\mathbf{z} \tag{2.18}$$

Ze względu na trójkątną strukturę macierzy $\mathbf{L}$ oraz $\mathbf{U}$, równania (2.17)- (2.18) można rozwiązać bezpośrednio przez odwrotne podstawianie, jak w metodzie Gaussa. Wymaga to wykonania n^2 operacji mnożenia i dzielenia, a więc tyle, ile potrzeba na pomnożenia macierzy przez wektor. Dużą oszczędność uzyskuje się wówczas, gdy równanie (2.16) trzeba rozwiązać dla różnych wartości wektora $\mathbf{b}$. Należy zauważyć, że macierze $\mathbf{L}$ oraz $\mathbf{U}$ mogą być zapisane w jednej macierzy $\mathbf{P}=[\mathbf{L}\backslash\mathbf{U}]$, gdyż elementy diagonalne macierzy $\mathbf{L}$ są zawsze równe 1, więc nie muszą być pamiętane.

Wektor $\mathbf{x}$ można określić za pomocą następującego algorytmu

for $i:=1$ **to** n **do** $z_i:=b_i-\sum_{m=1}^{i-1}p_{im}z_m$ {rozwiązanie równania (2.17)}

for $i:=n$ **to** 1 **step** -1 **do** $x_j:=\left(z_j-\sum_{m=j+1}^{n}p_{jm}x_m\right)/p_{jj}$ {rozwiązanie równania (2.18) }

Algorytm rozkładu $\mathbf{LU}$ można łatwo wyznaczyć na podstawie związku (2.15). Na przykład, dla $n=3$ macierz $\mathbf{A}$ wyraża się w następujący sposób za pomocą współczynników macierzy $\mathbf{L}$ oraz $\mathbf{U}$:

$$\mathbf{A}=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}=\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ l_{21}u_{11} & l_{21}u_{12}+u_{22} & l_{21}u_{13}+u_{23} \\ l_{31}u_{11} & l_{31}u_{12}+l_{32}u_{22} & l_{31}u_{13}+l_{32}u_{23}+u_{33} \end{bmatrix}$$

Z powyższego przedstawienia można określić sposób obliczania elementów macierzy $\mathbf{L}$ oraz $\mathbf{U}$:

1. $u_{11} = a_{11}$

   $l_{21} = a_{21} / u_{11}$

   $l_{31} = a_{31} / u_{11}$

2. $u_{12} = a_{12}$

   $u_{22} = a_{22} - l_{21} u_{12}$

   $l_{32} = (a_{32} - l_{31} u_{12}) / u_{22}$

3. $u_{13} = a_{13}$

   $u_{23} = a_{23} - l_{21} u_{13}$

   $u_{33} = a_{33} - l_{31} u_{13} - l_{32} u_{23}$

Widać, że w każdym z trzech kroków ($n = 3$) najpierw są obliczane elementy macierzy **U**, a następnie elementy macierzy **L** w danej kolumnie. Dla ogólnego przypadku można to zapisać w postaci następującego algorytmu

{warunki początkowe - inicjalizacja macierzy: $\mathbf{L} = \mathbf{1}$ $(n \times n)$, $\mathbf{U} = \mathbf{0}$ $(n \times n)$}

for $k := 1$ **to** n **do**

begin

for $i := k$ **to** n **do** $u_{ki} := a_{ki} - \sum_{m=1}^{k-1} l_{km} u_{mi}$;

for $j := k+1$ **to** n **do** $l_{jk} := \left(a_{jk} - \sum_{m=1}^{k-1} l_{jm} u_{mk} \right) / u_{kk}$;

end;

Algorytm ten jest nazywany algorytmem Gaussa-Banachiewicza [8].

Podobnie jak w przypadku algorytmu Gaussa, dla poprawienia skuteczności i dokładności algorytmu **LU** można stosować wybór maksymalnego elementu głównego w kolumnie. W tym celu należy porównać ze sobą wyrazy k-tej kolumny macierzy **A** leżące na i poniżej głównej przekątnej ($k \le j \le n$):

$$p_j := a_{jk} - \sum_{m=1}^{k-1} l_{jm} u_{mk} \quad k \le j \le n \tag{2.19}$$

i wybrać spośród nich największy co do modułu. Odpowiadający mu wiersz należy przestawić z rozpatrywanym k-tym wierszem macierzy **A**. Procedura ta nie prowadzi do znacznego skomplikowania algorytmu, gdyż wyrażenie (2.18) jest fragmentem głównego algorytmu i tak musi być obliczone.

Załóżmy, że elementy macierzy **L** oraz **U** będą zapisane na odpowiednich miejscach macierzy **A** (macierz ta nie zostanie zachowana), a elementy wektora $\mathbf{d} = \{d_i\}$ określają numery wierszy macierzy **A** zgodnie z przestawieniem wynikającym z wyboru maksymalnego elementu głównego. Przeprowadzone rozważania prowadzą wówczas do następującego algorytmu rozkładu **LU** z wyborem maksymalnego elementu głównego w kolumnie.

{*warunki początkowe*}

$err := 0;$

for $i := 1$ **to** n **do** $d_j := 0;$

{*główny algorytm*}

for $k := 1$ **to** n **do**

begin

{*wybór elementu głównego*}

$b := 0;$

for $j := k$ **to** n **do**

begin

$$a_{jk} := a_{jk} - \sum_{m=1}^{k-1} a_{jm} a_{mk};$$

if $|a_{jk}| > b$ **then**

begin $b := |a_{jk}|$**;** $w := j$ **end;**

end;

if $b = 0$ **then begin** $err := 1$**; halt end;** {*brak rozwiązania*}

{*przestawienie wierszy*}

if $w > k$ **then**

begin

for $j := 1$ **to** n **do**

begin $b := a_{kj}$**;** $a_{kj} := a_{wj}$**;** $a_{wj} := b$ **end;**

$s := d_k$**;** $d_k := d_w$**;** $d_w := s$

end;

{obliczenie u_{ki}}

for $i := k$ **to** n **do** $a_{ki} := a_{ki} - \sum_{m=1}^{k-1} a_{km} a_{mi};$

{obliczenie l_{jk}}

for $j := k+1$ **to** n **do** $a_{jk} := a_{jk} / a_{kk};$

end;

Jeśli wynik tego algorytmu jest stosowany łącznie z algorytmem rozwiązywania równania (2.16), to wektor **b** należy uszeregować zgodnie z indeksami zawartymi w wektorze przestawień **d**:

$b_i = b_{d_i}$**,** $i = 1,2,...,n$**,**

gdzie: wektor $\mathbf{b} = \{b_i\}$ może być bezpośrednio użyty w algorytmie (2.17).

Algorytm rozwiązywania układu równań liniowych może być stosowany do *odwracania macierzy*. Zauważmy, że

$$\mathbf{A}\mathbf{A}^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{2.20}$$

zatem

$$\mathbf{A}^{-1} = \mathbf{A}^{-1}\left[\mathbf{1}_{(1)} \quad \mathbf{1}_{(2)} \quad \cdots \quad \mathbf{1}_{(n)}\right] = \left[\mathbf{a}_{(1)}^{(-1)} \quad \mathbf{a}_{(2)}^{(-1)} \quad \cdots \quad \mathbf{a}_{(n)}^{(-1)}\right] \tag{2.21}$$

gdzie

$\mathbf{1}_{(i)}$ jest wektorem kolumnowym $(n \times 1)$ z jedynką na i-tej pozycji i zerami w pozostałych miejscach,

$\mathbf{a}_{(i)}^{(-1)}$ jest i-tą kolumną poszukiwanej macierzy $\mathbf{A}^{-1}$.

Można zauważyć, że $\mathbf{a}_{(i)}^{(-1)}$ jest rozwiązaniem równania

$$\mathbf{A}\mathbf{a}_{(i)}^{(-1)} = \mathbf{1}_{(i)} \tag{2.22}$$

zatem w celu obliczenia macierzy $\mathbf{A}^{-1}$ należy rozwiązać n równań typu (2.22). W przedstawionych metodach wymaga to tylko jednokrotnego rozkładu macierzy $\mathbf{A}$ (na macierz trójkątna lub na macierze LU). Złożoność obliczeniowa takiego algorytmu jest z grubsza równa trzykrotnej złożoności rozwiązania pojedynczego układu równań liniowych.

2.4. Iteracyjne metody rozwiązywania układu równań liniowych

Przedstawione powyżej metody eliminacji nie uwzględniają różnych właściwości macierzy współczynników, które w metodach iteracyjnych mogą prowadzić do uproszczenia obliczeń, co jest szczególnie ważne w zadaniach o dużych rozmiarach. Ma to miejsce, na przykład, w przypadku macierzy o silnie dominującej przekątnej, gdy wiele elementów leżących poza przekątną ma małą wartość lub są to elementy zerowe. Można w takim przypadku założyć, że wszystkie elementy leżące na przekątnej macierzy współczynników równania są różne od zera. W taki przypadku równanie (2.3) można zapisać w następującej postaci:

$$x_i = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j\right), \; i = 1,2,..,n \tag{2.23}$$

Przy zadanych wartościach początkowych poszukiwanych niewiadomych zdefiniowanych przez wektor **x**, kolejne przybliżenia można uzyskać zgodnie z algorytmem iteracyjnym. Metody iteracyjne sprowadzają się do poszukiwania rozwiązania układu równań o postaci

$$f_i(x_1, x_2, ..., x_n) = 0,\ i = 1,2,...,n \tag{2.24}$$

który jest równoważny (2.3).

Ogólny schemat iteracyjnego rozwiązywania układu n równań można zapisać następującą zależnością

$$x_i^{j+1} = x_i^j + \lambda^j \upsilon_i^j,\ i = 1,2,...,n \tag{2.25}$$

gdzie j jest numerem kroku iteracji,

λ^j jest wielkością kroku iteracji,

υ_i^j parametrem określającym 'kierunek' iteracji,

przy założonych początkowych wartościach x_i^0, $i = 1,2,...,n$.

W przypadku układu równań liniowych, odpowiednie metody iteracyjne są tworzone na podstawie przedstawienia równania (2.3) w następującej postaci

$$\mathbf{x} = \mathbf{C}\mathbf{x} + \mathbf{d} \tag{2.26}$$

skąd kolejne przybliżenia rozwiązania są określane zgodnie z równaniem

$$\mathbf{x}^{k+1} = \mathbf{C}\mathbf{x}^k + \mathbf{d} \tag{2.27}$$

Zgodnie z tym algorytmem, równanie (2.23) można zapisać w następującej formie iteracyjnej:

$$x_i^{k+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j^k \right),\ i = 1,2,...,n \tag{2.28}$$

Zależność ta jest znana jako iteracyjna metoda Jakobiego rozwiązywania równań liniowych.

Poszczególne metody różnią się sposobem wyboru kroku iteracji λ oraz parametru υ. Omówimy poniżej pewną modyfikację metody Jakobiego, znaną jako *metoda Gaussa-Seidla*.

W metodzie Gaussa-Seidla kolejne przybliżenie rozwiązania równania (2.3) określa się zgodnie z następującym podstawieniem

$$\begin{aligned}
a_{11}x_1^{k+1} + \ & a_{12}x_2^k + \ a_{13}x_3^k + \ \cdots + \ a_{1n}x_n^k = \ b_1 \\
a_{21}x_1^{k+1} + \ & a_{22}x_2^{k+1} + \ a_{23}x_3^k + \ \cdots + \ a_{2n}x_n^k = \ b_2 \\
\cdots \ & \cdots \ \cdots \ \cdots \ \cdots \ \cdots \\
a_{n1}x_1^{k+1} + \ & a_{n2}x_2^{k+1} + \ a_{n3}x_3^{k+1} + \ \cdots + \ a_{nn}x_n^{k+1} = \ b_n
\end{aligned} \tag{2.29}$$

co można zapisać w następującej formie macierzowej

$$\mathbf{A}_1\mathbf{x}^{k+1} + \mathbf{A}_2\mathbf{x}^k = \mathbf{b} \tag{2.30}$$

gdzie

$$\mathbf{A}_1 = \begin{bmatrix} a_{11} & & & & \\ a_{21} & a_{22} & & 0 & \\ \vdots & \vdots & \dots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & \\ a_{n,1} & a_{n,2} & \dots & a_{n,n-1} & a_{nn} \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} a_{12} & \dots & a_{1,n-1} & a_{1n} \\ & \dots & a_{2,n-1} & a_{2n} \\ \vdots & \dots & \vdots & \vdots \\ 0 & \dots & & a_{n-1,n} \\ & \dots & & \end{bmatrix}$$

Algorytm iteracyjnego poszukiwania rozwiązania wynika bezpośrednio z (2.30). W następujących po sobie krokach określane jest przybliżenie kolejnej zmiennej po uwzględnieniu uzyskanych przybliżeń poprzednich zmiennych:

$$\begin{aligned} x_1^{k+1} &= -a_{12}x_2^k/a_{11} - a_{13}x_3^k/a_{11} - \cdots - a_{1n}x_n^k/a_{11} + b_1/a_{11} \\ x_2^{k+1} &= -a_{21}x_1^{k+1}/a_{22} - a_{23}x_3^k/a_{22} - \cdots - a_{2n}x_n^k/a_{22} + b_2/a_{22} \\ &\cdots \qquad\qquad \cdots \qquad\qquad \cdots \quad \cdots \qquad\qquad \cdots \qquad \cdots \\ x_n^{k+1} &= -a_{n1}x_1^{k+1}/a_{nn} - a_{n2}x_2^{k+1}/a_{nn} - \cdots - a_{n,n-1}x_{n-1}^{k+1}/a_{nn} + b_n/a_{nn} \end{aligned} \tag{2.31}$$

co może być zapisane w następującej ogólnej postaci

$$x_i^{k+1} = \frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1}\frac{a_{ij}}{a_{ii}}x_j^{k+1} - \sum_{j=i+1}^{n}\frac{a_{ij}}{a_{ii}}x_j^k \tag{2.32}$$

Warunki zbieżności procesu iteracyjnego związanego z algorytmem Gaussa-Seidla mogą być określone na podstawie badania równania uzyskanego z (2.30)

$$\mathbf{x}^{k+1} = -\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{x}^k + \mathbf{A}_1^{-1}\mathbf{b} \tag{2.33}$$

Można pokazać (patrz rozdział dotyczący iteracyjnego rozwiązywania układów równań nieliniowych), że warunek zbieżności procesu określonego przez (2.33) jest określony przez wartości własne macierzy $-\mathbf{A}_1^{-1}\mathbf{A}_2$. Dostatecznym i wystarczającym warunkiem zbieżności metody jest to aby moduły wszystkich wartości własnych tej macierzy były mniejsze od jedności. Jest to równoważne następującemu warunkowi odnoszącemu się do współczynników macierzy $\mathbf{A}$

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \quad i = 1,2,...,n \tag{2.34}$$

co oznacza, że rozwiązanie iteracyjne jest możliwe, jeśli moduły elementów diagonalnych są większe od sumy modułów wszystkich pozostałych elementów w wierszu macierzy. Większość zagadnień spotykanych w technice spełnia ten warunek. Niekiedy należy wcześniej odpowiednio przekształcić wyjściowy układ równań.

Ostatecznie, metoda Gaussa-Seidla iteracyjnego rozwiązywania układów równań liniowych przybiera formę następującego algorytmu.

1. Uporządkować wyjściowy układ n równań tak, aby w macierzy współczynników **A** największe co do modułu elementy znalazły się na przekątnej, co jest określone następującym warunkiem
   $|a_{ii}| > |a_{ij}|_{i \neq j}$, $i = 1,2,...,n$, $j = 1,2,...,n$
2. Przyjąć warunki początkowe
   $\{x\}_0 = \{x_2^0 \quad x_3^0 \quad \cdots \quad x_n^0\}$
3. Powtarzać proces iteracyjny (2.33) dla $k = 1,2,\cdots$ aż spełniony zostanie warunek
   $\max_{i=1,2,...,n} \left| x_i^{k+1} - x_i^k \right| < \varepsilon$
   gdzie ε - założona dokładność obliczeń.

Metody iteracyjne stosowane są zazwyczaj do rozwiązywania dużych układów równań, w których wiele współczynników ma wartość zerową (są to tak zwane równania z macierzami rzadkimi). Wówczas można oczekiwać mniejszej złożoności obliczeniowej takiego podejścia niż stosowanie metod skończonych. Metody iteracyjne są także stosowane do poprawiania (zwiększania dokładności) wyników uzyskanych w rezultacie stosowania metod skończonych.

3. Rozwiązywanie równań nieliniowych

3.1. Zagadnienia jednowymiarowe

Załóżmy, że dana jest funkcja $f(x)$ rzeczywistego argumentu x. Celem naszych działań jest określenie rozwiązania następującego równania

$$f(x) = 0 \tag{3.1}$$

to znaczy, określenie wartości zmiennej x, dla których spełniona jest zależność (3.1).

Należy zauważyć, że w ogólnym przypadku zadanie to nie jest proste, gdyż ze względu na nieliniowość funkcji $f(x)$ nie jest nawet wiadomo ilu rozwiązań można oczekiwać. Nie ma ogólnych, jednoznacznych metod rozwiązywania takich zadań. Znane są natomiast metody przybliżone, które opierają się na poszukiwaniu rozwiązań w drodze kolejnych iteracyjnych przybliżeń.

Metoda prostej iteracji

Zapiszmy równanie (3.1) w następującej postaci

$$x = g(x) \tag{3.2}$$

Iteracyjne rozwiązanie równania (3.2) polega na wykonaniu następujących działań

$$x^{k+1} = g(x^k) \tag{3.3}$$

przy warunkach początkowych: $x^0 = x_0$.

Powstaje oczywiście pytanie, czy ciąg wartości x^k uzyskany w wyniku stosowania procedury (3.3) prowadzi do rozwiązania, to znaczy, czy metoda jest stabilna. Dowodzi się, że warunek zbieżności można zapisać następująco. Dla dowolnie wybranej zmiennej ξ zachodzi nierówność

$$\left|g(x) - g(\xi)\right| \le K\left|x - \xi\right| \tag{3.4}$$

gdzie $K < 1$.

Jeśli warunek (3.4) jest spełniony, to algorytm (3.3) nazywa się odwzorowaniem zawężającym, które prowadzi do rozwiązania. Warunek ten w wielu przypadkach nie jest spełniony i różne metody iteracyjnego rozwiązywania równania (3.1) biorą się stąd, żeby tak wyrazić równanie (3.1) w formie (3.2), aby poszerzyć obszar zbieżności rozwiązania i przyśpieszyć proces tego rozwiązania. W ogólnym przypadku odwzorowanie (3.3) można zapisać następująco

$$x^{k+1} = \Phi(x^k) \tag{3.5}$$

przy czym, funkcja Φ, znana jako funkcja iteracyjna, jest tak dobrana, że jeśli x' jest rozwiązaniem równania (3.1), to $\Phi(x') = x'$.

Metoda połowienia

Metoda połowienia (*metoda bisekcji*) wywodzi się z obserwacji, że jeśli na granicach przedziału $[a,b]$ funkcja $f(x)$ ma różne znaki, to wewnątrz przedziału znajduje się przynajmniej jedno miejsce zerowe tej funkcji. Z kolei strategia poszukiwania kolejnego, bliższego rozwiązania polega na wskazaniu w tym celu punktu, leżącego w środku tego właśnie przedziału. W ten sposób otrzymujemy następujący algorytm.

```
{warunki początkowe}
x := a ; y := b ;
fx := f(x); fy := f(y);        { fx oraz fy powinny mieć różne znaki }
{pętla iteracyjna}
while abs(x − y) > ε do
 begin
       {połowienie}
       z := (x + y) / 2 ;
       fz := f(z);
       if sign( fz) = sign( fx) then
         begin
               p := x ; x := z ; z := p ;
         end;
        else
         begin
               p := y ; y := z ; z := p ;
         end;
end;
```

Można zauważyć, że w przypadku cyfrowej reprezentacji liczb, w każdej iteracji połowienia dokładność rozwiązania wzrasta o jeden bit. Algorytm jest zatem zbieżny dosyć wolno, chociaż przy poprawnym wyborze początkowego przedziału, zawsze prowadzi do rozwiązania. Jest on często stosowany jako procedura, która prowadzi do rozwiązania w skrajnych sytuacjach, gdy zawodzą inne metody.

Metoda Newtona

Znaczne przyspieszenie procesu iteracyjnego można uzyskać, jeśli odpowiednio dobierze się funkcję iteracyjną Φ w (3.5). W tym celu można zastąpić nieliniową funkcję $f(x)$ w pobliżu rozwiązania (to jest w pobliżu zera) za pomocą jej rozwinięcia w szereg Taylora

$$f(\xi) = 0 = f(x^0) + f'(x^0)(\xi - x^0) + f''(x^0)\frac{(\xi - x^0)^2}{2!} + \\ \ldots + f^{(k)}(x^0 + \theta)\frac{(\xi - x^0)^k}{k!} + \theta(\xi - x^0) \tag{3.6}$$

Pozostawiając tylko dwa pierwsze wyrazy rozwinięcia (przybliżenie liniowe) otrzymujemy

$$0 \approx f(x^0) + f'(x^0)(\xi - x^0) \tag{3.7}$$

oraz

$$\xi \approx x^0 - \frac{f(x^0)}{f'(x^0)}, \text{ jeśli } f'(x) \neq 0,$$

co w ogólności prowadzi do następującej procedury iteracyjnej

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}, \tag{3.8}$$

która jest znana jako metoda Newtona rozwiązywania równań nieliniowych.

Można pokazać, że Metoda Newtona dla pierwiastków jednokrotnych ma przynajmniej zbieżność kwadratową, co odnosi się do stopnia przybliżenia do rozwiązania w kolejnych iteracjach.

Metoda siecznych

Jeśli w metodzie Newtona zastąpić różniczkowanie funkcji za pomocą wyrażenia różnicowego, to otrzymamy przybliżenie metody Newtona, które ze względu na interpretację graficzną jest znane jako metoda siecznych. Przybliżone różniczkowanie funkcji $f(x)$ może być określone następująco

$$f'(x^k) \approx \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}} \tag{3.9}$$

co, po podstawieniu do (3.9), prowadzi do następującego algorytmu

$$x^{k+1} = x^k - \frac{f(x^k)\left(x^k - x^{k-1}\right)}{f(x^k) - f(x^{k-1})} \tag{3.10}$$

jeśli tylko $f(x^k) - f(x^{k-1}) \neq 0$.

Metoda siecznych jest w wielu przypadkach wygodniejsza do stosowania (szczególnie w tych przypadkach, gdy nie ma możliwości określenia pochodnej funkcji $f(x)$), jednak jest ona słabiej zbieżna.

Zauważmy, że powyższe metody mogą być stosowane jedynie wówczas, gdy spełniony jest warunek o różnej od zera wartości mianownika odpowiedniego wyrażenia (3.8) lub (3.10). Poprawnie sformułowany algorytm powinien uwzględniać to i w przypadku, gdy wartość ta jest odpowiednio mała, powinna być proponowana inna wersja algorytmu.

Metody wielokrokowe: algorytm Aitkena

Dane jest równanie nieliniowe o postaci

$$f(x) = 0 \tag{3.11}$$

Metoda prostej iteracji poszukiwania wartości x, dla której spełnione jest równanie (3.11) polega na przekształceniu go do postaci

$$x = g(x) \tag{3.12}$$

dla której można sformułować następującą regułę iteracyjną

$$x^{k+1} = g(x^k) \tag{3.13}$$

z warunkami początkowymi: $x^0 = x_0$.

Algorytm (3.13) prowadzi do rozwiązania, gdy proces iteracyjny jest zbieżny. Zbieżność jest zapewniona, gdy spełniony jest następujący warunek. Dla dowolnie wybranej zmiennej ξ zachodzi nierówność

$$\left|g(x) - g(\xi)\right| \leq K\left|x - \xi\right| \tag{3.14}$$

gdzie $K < 1$.

Aby rozszerzyć obszar zbieżności i przyspieszyć zbieżność procesu iteracyjnego można stosować jego korekcję według metody Aitkena. Jej idea polega na zastąpieniu problemu rozwiązania równania (3.11) przez zagadnienie poszukiwania zer funkcji, utworzonej z kolejnych wyników prostej iteracji:

$$h\left(x^k\right) = x^k - x^{k-1} = 0 \tag{3.15}$$

gdzie zmienne x^k oblicza się według (3.13).

Problem sprowadza się zatem do określenia sposobu korekcji metody prostej iteracji w celu uzyskania rozwiązania procesu (3.15). Ponieważ funkcja $h(x^k)$ jest dostępna w postaci numerycznej, więc rozwiązania (3.15) można poszukiwać za pomocą metody siecznych:

$$x_p^{k+1} = x^k - \frac{h\left(x^k\right)}{\frac{\Delta h\left(x^k\right)}{\Delta\left(x^k\right)}} = x^k - \frac{x^k - x^{k-1}}{\frac{\left(x^{k+1} - x^k\right) - \left(x^k - x^{k-1}\right)}{\left(x^k - x^{k-1}\right)}}, \tag{3.16}$$

przy czym:

$$\Delta h\left(x^k\right)=\left(x^{k+1}-x^k\right)-\left(x^k-x^{k-1}\right)=h\left(x^{k+1}\right)-h\left(x^k\right),$$
$$\Delta\left(x^k\right)=\left(x^k-x^{k-1}\right)=h\left(x^k\right).$$

Korekcja jest zatem dokonywana na podstawie trzech kolejnych wartości x^{k-1}, x^k, oraz x^{k+1}, przybliżenia, uzyskanych według metody prostej iteracji zgodnie z następującą regułą:

$$x_p^{k+1}=x^k-\frac{\left(x^{k+1}-x^k\right)^2}{x^{k+2}-2x^{k+1}+x^k} \tag{3.17}$$

Wynik tej korekcji przyjmuje się w charakterze kolejnego przybliżenia rozwiązania: $x^{k+1}=x_p^{k+1}$, po czym następują znów dwa kroki procedury (3.13) do kolejnej korekcji (3.17). W ten sposób uzyskuje się algorytm o następującej postaci.

1. Przyjąć warunki początkowe
   $x^0=x_0,\ \ k=0$ - numer kroku iteracji
2. Wykonać dwa kroki prostej iteracji
   $y^k=g(x^k),\ \ z^k=g(y^k)$
3. Skorygować wynik:
   $$\Delta^k=\frac{\left(y^k-x^k\right)^2}{z^k-2y^k+x^k}$$
   $$x^{k+1}=x^k-\Delta^k$$
4. Jeśli $abs(\Delta^k)>eps$, $k=k+1$, przejdź do 2

3.2. Rozwiązywanie układów równań nieliniowych

Układ równań nieliniowych może być w ogólnym przypadku zapisany następująco

$$f(\mathbf{x})=\begin{bmatrix} f_1(x_1,x_2,...,x_n)\\ f_2(x_1,x_2,...,x_n)\\ ...\\ f_n(x_1,x_2,...,x_n)\end{bmatrix}=0 \tag{3.18}$$

Rozwiązanie tego układu równań oznacza określenie wektora $\mathbf{x}=\begin{bmatrix}x_1 & x_2 & ... & x_n\end{bmatrix}^T$, dla którego spełnione jest równanie (3.18).

Metoda Newtona-Raphsona

Metody rozwiązywania tego zagadnienia powstają przez odpowiednie rozszerzenie metod rozwiązywania pojedynczych równań. W szczególności, równanie (3.7) dla przypadku wielowymiarowego ma następującą postać

$$0\approx f(\mathbf{x}^0)+f'(\mathbf{x}^0)(\boldsymbol{\xi}-\mathbf{x}^0) \tag{3.19}$$

gdzie wektor ξ przedstawia współrzędne punktu, w którym spełniony jest warunek (3.18).

Macierz określająca pochodną $f'(\mathbf{x}^0)$ jest nazywana Jakobianem (macierzą Jakobiego)

$$\mathbf{J}\left(f(\mathbf{x})\right) = f'(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{3.20}$$

Analogicznie do (3.8), rozwinięcie (3.24) prowadzi do następującej iteracyjnej procedury rozwiązywania układu równań (3.18)

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{J}^{-1}\left(f(\mathbf{x}^k)\right) f(\mathbf{x}^k) \tag{3.21}$$

jeśli $\det\left[\mathbf{J}\left(f(\mathbf{x}^k)\right)\right] \neq 0$, przy czym

$$\mathbf{J}\left(f(\mathbf{x}^k)\right) = \mathbf{J}\left(f(\mathbf{x})\right)\Big|_{\mathbf{x}=\mathbf{x}^k}$$

Algorytm (3.21) jest znany jako metoda Newtona-Raphsona iteracyjnego rozwiązywania układu równań nieliniowych. W programach komputerowych wzór (3.21) jest realizowany przez następujący algorytm

- oblicz $f(\mathbf{x}^k)$,
- oblicz $\mathbf{J}\left(f(\mathbf{x}^k)\right) = f'(\mathbf{x}^k)$,
- rozwiąż układ równań liniowych $\mathbf{J}\left(f(\mathbf{x}^k)\right)\mathbf{z}^k = f(\mathbf{x}^k)$
- podstaw $\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{z}^k$

W charakterze oceny zbieżności procesu iteracyjnego można przyjąć normę wektora $\mathbf{z}^k$ odniesioną do normy wektora $\mathbf{x}^k$

$$\frac{\left\|\mathbf{z}^k\right\|}{\left\|\mathbf{x}^k\right\|} < \varepsilon \tag{3.22}$$

Ze względu na ograniczoną dokładność obliczania funkcji $f(\mathbf{x}^k)$ oraz Jakobianu $\mathbf{J}\left(f(\mathbf{x}^k)\right)$, dokładność całego algorytmu jest ograniczona. Objawia się to tym, że począwszy od pewnej wartości minimalnej, norma wektora $\mathbf{z}^k$ zacznie narastać. Jest to sygnał, że należy skończyć obliczenia. Wynika stąd następujące kryterium zakończenia obliczeń

$$\left\|\mathbf{z}^{k+1}\right\| > \rho\left\|\mathbf{z}^k\right\| \tag{3.23}$$

gdzie ρ jest rzędu jedności.

Metoda siecznych

Również metoda siecznych może być rozszerzona na przypadek wielowymiarowy. Łatwo zauważyć, że równanie (3.10) można uogólnić następująco

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \Delta\mathbf{X}^k \left(\Delta\mathbf{F}^k\right)^{-1} f(\mathbf{x}^k) \tag{3.24}$$

przez analogię do rozwinięcia (3.19)

$$0 \approx f(\mathbf{x}^k) + \Delta\mathbf{F}^k \left(\Delta\mathbf{X}^k\right)^{-1} \left(\xi - \mathbf{x}^k\right) \tag{3.25}$$

gdzie

$\Delta\mathbf{X}^k$, $\Delta\mathbf{F}^k$ są macierzami $n \times n$ o kolumnach, odpowiednio:

$\Delta\mathbf{x}_j^k = \mathbf{x}^j - \mathbf{x}^k$ oraz $\Delta\mathbf{f}_j^k = f(\mathbf{x}^j) - f(\mathbf{x}^k)$, $j = k-n, k-n-1, ..., k-1$.

Równania (3.22), (3.24) mają sens wtedy, gdy macierze $\Delta\mathbf{X}^k$, $\Delta\mathbf{F}^k$ są nieosobliwe. Jednakże zbieżność ciągu $\mathbf{x}^k$ wymaga silnej nieosobliwości wszystkich macierzy $\Delta\mathbf{X}^k$, co oznacza, że moduł wyznacznika tej macierzy powinien być dostatecznie duży.

Z równania (3.22) widać, że w każdym kroku metody siecznych dla przypadku wielowymiarowego wymagana jest znajomość $n+1$ wartości wektora $\mathbf{x}$ oraz tyluż wartości funkcji $f(\mathbf{x})$. Algorytm iteracyjny składa się z następujących kroków

- warunki początkowe: założyć wartości wektorów: $\mathbf{x}^{-n}$, $\mathbf{x}^{-n+1}$, ..., $\mathbf{x}^0$ oraz przyjąć numer kroku iteracji $k = 0$
- obliczyć macierze $\Delta\mathbf{X}^k$, $\Delta\mathbf{F}^k$
- rozwiązać układ równań liniowych $\Delta\mathbf{F}^k\mathbf{z}^k = f(\mathbf{x}^k)$
- obliczyć nową wartość wektora $\mathbf{x}^{k+1} = \mathbf{x}^k - \Delta\mathbf{X}^k\mathbf{z}^k$

Należy zauważyć, że ograniczenia warunkujące stosowanie metody siecznych mogą uniemożliwiać wykonanie kolejnych kroków procesu iteracyjnego. Trzeba zatem stosować odpowiednie rozwiązania (inne metody pomocnicze), pozwalające uniknąć zatrzymania obliczeń.

4. Interpolacja

4.1. Wprowadzenie

Zadanie interpolacji odnosi się do działań zmierzających do przedstawienia funkcji w postaci ciągłej, gdy znana jest ona w postaci dyskretnej. Jest to zatem zdanie odwrotne do dyskretyzacji lub próbkowania wielkości ciągłej.

Załóżmy, że dla danego zbioru zmiennych niezależnych z przedziału $< a;b >$: $x_1, x_2, ..., x_{n+1}$ znane są przyporządkowane im wartości funkcji: $y_1, y_2, ..., y_{n+1}$. Zależność ta jest zazwyczaj przedstawiana w postaci tabelarycznej:

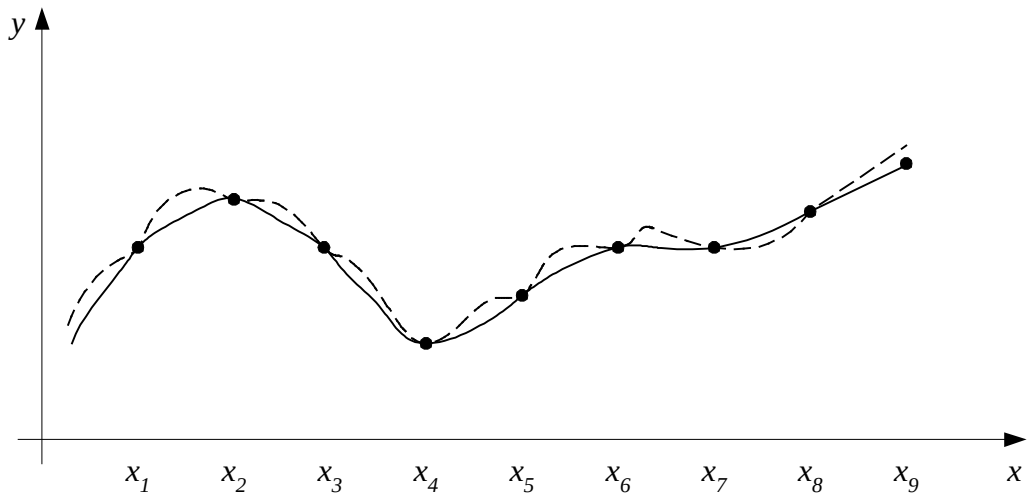
$y_1 = f(x_1)$,

$y_2 = f(x_2)$,

...

$y_{n+1} = f(x_{n+1})$.

Zadaniem interpolacji jest wyznaczenie przybliżonych wartości funkcji dla wartości zmiennych niezależnych z przedziału $< a;b >$, lecz nie będących punktami ze zbioru $x_1, x_2, ..., x_{n+1}$. Jest to bardzo ogólne sformułowanie zadania i łatwo zauważyć, że istnieje nieskończenie wiele sposobów jego rozwiązania, jeśli nie jest zadany sposób przeprowadzenia funkcji interpolacyjnej przez zadane punkty (rys. 4.1).



Rys. 4.1. Zasada interpolacji funkcji dyskretnej

Najczęściej poszukuje się funkcji interpolacyjnej o ściśle określonej postaci, tak, aby zachowywała się ona w określony sposób. Są to często wielomiany algebraiczne lub trygonometryczne.

Podstawowym celem interpolacji jest określenie wartości funkcji danej w postaci stabelaryzowanej dla zmiennej x mieszczącej się pomiędzy danymi zawartymi w tablicy. Można w ten sposób zapamiętać w komputerze zależność określoną na

podstawie pomiaru. Typowymi przykładami zastosowania interpolacji jest obliczanie całek oraz pochodnych funkcji dyskretnych w czasie. W takim przypadku, w celu poprawienia dokładności obliczenia całki można skorzystać z wartości funkcji aproksymującej określonej dla dowolnych wartości argumentu.

4.2. Wielomian interpolacyjny Newtona

Załóżmy, że dana jest funkcja $f(x)$ w postaci tablicy, w której punktom $x_1, x_2, ..., x_n$, zwanym węzłami interpolacji, przyporządkowane są wartości $f(x_1), f(x_2), ..., f(x_n)$. Zakłada się, że $x_i \neq x_j$ dla $i \neq j$. Funkcja interpolacyjna może być określona w postaci wielomianu:

$$P(x) = b_1 + b_2 x + b_3 x^2 + ... + b_{n+1} x^n \tag{4.1}$$

Jeśli funkcja dyskretna $f(x)$ dana jest w dwóch punktach (n = 2), to funkcja interpolacyjna w postaci (4.1) redukuje się do prostej (n+1 = 2). Podobnie, przez trzy punkty (n = 3) można jednoznacznie poprowadzić parabolę, określoną przez wielomian drugiego stopnia (n+1 = 3). Można dowieść, że w ogólnym przypadku, dla n+1 punktów węzłowych (x_i, y_i), istnieje tylko jeden wielomian $P(x)$ spełniający warunek [1], [13]:

$$P(x_i) = y_i, \; i=1, 2, ..., n+1 \tag{4.2}$$

W przypadku wzoru interpolacyjnego Newtona, poszukiwany wielomian interpolacyjny jest zapisywany w postaci:

$$\begin{aligned} P(x) &= a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2) + ... + a_{n+1}(x - x_1)(x - x_2)...(x - x_n) \\ &= b_1 + b_2 x + b_3 x^2 + \cdots + b_{n+1} x^n \end{aligned} \tag{4.3}$$

Korzystając z właściwości (4.2), powyższy zapis pozwala napisać następujący układ równań:

$$\begin{aligned} P(x_1) &= y_1 &&= a_1 \\ P(x_2) &= y_2 &&= a_1 + a_2(x_2 - x_1) \\ &\vdots \\ P(x_{n+1}) &= y_{n+1} &&= a_1 + a_2(x_{n+1} - x_1) + ... + a_{n+1}(x_{n+1} - x_1)(x_{n+1} - x_2)...(x_{n+1} - x_n) \end{aligned} \tag{4.4}$$

co można zapisać w następującej postaci macierzowej:

$$\Delta\mathbf{X} \cdot \mathbf{a} = \mathbf{y} \tag{4.5}$$

gdzie:

$$\Delta\mathbf{X} = \begin{bmatrix} 1 & & & & \\ 1 & x_2 - x_1 & & 0 & \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n+1} - x_1 & x_{n+1} - x_2 & \cdots & x_{n+1} - x_{n+1} \end{bmatrix}, \; \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n+1} \end{bmatrix}, \; \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n+1} \end{bmatrix}$$

Jak widać, macierz $\Delta\mathbf{X}$ ma dogodną trójkątną postać, co pozwala bezpośrednio podać rozwiązanie równania (4.5), wykonując podobne działania, jak w procedurze odwrotnego podstawiania algorytmu Gaussa:

$$\begin{aligned} a_1 &= y_1 \\ a_2 &= \frac{y_2 - a_1}{x_2 - x_1} \\ a_3 &= \frac{y_3 - a_1 - a_2(x_3 - x_1)}{(x_3 - x_1)(x_3 - x_2)} \\ &\ldots \end{aligned} \tag{4.6}$$

Przykład. Interpolowana funkcja dana jest dla: $x_1 = 1$, $x_2 = 2$, $x_3 = 4$, przy czym wartości funkcji przyjmują wartości: $f(x_1) = 1$, $f(x_2) = 4$, $f(x_3) = 0$. Określić funkcję interpolacyjną.

Poszukiwana funkcja ma postać jak w (4.3), przy czym współczynniki są obliczane zgodnie z (4.6):

$a_1 = f(x_1) = 1$,

$$a_2 = \frac{f(x_2) - a_1}{x_2 - x_1} = \frac{4-1}{2-1} = 3,$$

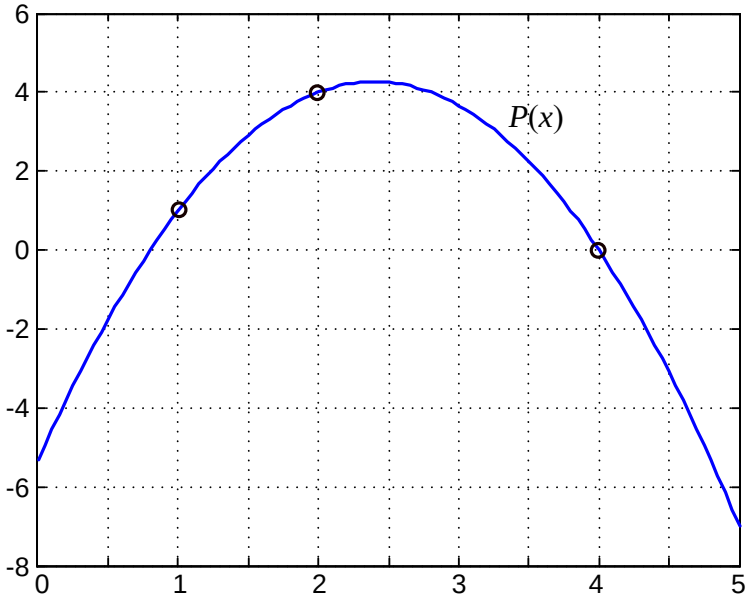
$$a_3 = \frac{f(x_3) - a_1 - a_2(x_3 - x_1)}{(x_3 - x_1)(x_3 - x_2)} = \frac{0-1-3(4-1)}{(4-1)(4-2)} = \frac{-5}{3}.$$

Na podstawie (4.3), współczynniki funkcji interpolacyjnej (4.1) przyjmą następujące wartości: $b_1 = a_1 - a_2x_1 + a_3x_1x_2 = -16/3$, $b_2 = a_2 - a_3(x_1 + x_2) = 8$, $b_3 = a_3 = -5/3$.

Zatem, funkcja interpolacyjna ma następującą postać:

$$P(x) = \frac{-1}{3}\left(16 - 24x + 5x^2\right).$$

Na rys. 4.2 pokazany jest przebieg uzyskanej funkcji interpolacyjnej z zaznaczonymi wartościami danej funkcji dyskretnej.



Rys. 4.2. Przebieg funkcji interpolacyjnej

Powyższe zależności wygodnie jest zapisać wprowadzając pojęcie ilorazów różnicowych. Oznaczmy i – tą różnicę

$$h_i = x_{i+1} - x_i \tag{4.7}$$

Wyrażenia

$$\Delta_i = \frac{f(x_{i+1}) - f(x_i)}{h_i} = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{4.8}$$

nazywa się ilorazami różnicowymi pierwszego rzędu. Odpowiednio także:

$$\Delta_i^{(2)} = \frac{\Delta_{i+1} - \Delta_i}{h_{i+1} + h_i} = \frac{(x_{i+1} - x_i)(f(x_{i+2}) - f(x_{i+1})) - (x_{i+2} - x_{i+1})(f(x_{i+1}) - f(x_i))}{(x_{i+2} - x_{i+1})(x_{i+1} - x_i)(x_{i+2} - x_i)} \tag{4.9}$$

- iloraz drugiego rzędu

$$\Delta_i^{(3)} = \frac{\Delta_{i+1}^{(2)} - \Delta_i^{(2)}}{h_{i+2} + h_{i+1} + h_i} = \frac{\Delta_{i+1}^{(2)} - \Delta_i^{(2)}}{x_{i+3} - x_i} \quad \text{- iloraz trzeciego rzędu} \tag{4.10}$$

i ogólnie:

$$\Delta_i^{(k)} = \frac{\Delta_{i+1}^{(k-1)} - \Delta_i^{(k-1)}}{x_{i+k} - x_i} \quad \text{dla } k = 1, 2, ..., n-1,\ i = 1, 2, ..., n-k \tag{4.11}$$

Dla zbioru n par $(x_i, f(x_i))$ można utworzyć tablicę ilorazów różnicowych, zwaną schematem ilorazów różnicowych (patrz Tablica).

Wielomian interpolacyjny Newtona ma następującą postać:

$$P(x) = f(x_1) + \Delta_1(x - x_1) + \Delta_1^{(2)}(x - x_1)(x - x_2) + ... + \Delta_1^{(n-1)}(x - x_1)(x - x_2)...(x - x_{n-1}) \tag{4.12}$$

Widać, że w postaci wielomianu $P(x)$ występują współczynniki z górnej przekątnej schematu ilorazów różnicowych. Można sprawdzić, że

$P(x_i) = f(x_i)$ dla $i = 1, 2, ..., n$.

Algorytm interpolacyjny Newtona sprowadza się więc do obliczenia ilorazów różnicowych oraz określenia wartości wielomianu dla konkretnej wartości zmiennej x. Ważne znaczenie ma przypadek, gdy wszystkie punkty stałe (węzły) są jednakowo od siebie oddalone. Wówczas mamy: $h = h_i = x_{i+1} - x_i = const$ oraz

x_i	$f(x_i)$	Ilorazy różnicowe				
		I rząd	II rząd	III rząd	IV rząd	V rząd
x_1	$f(x_1)$	Δ_1	$\Delta_1^{(2)}$	$\Delta_1^{(3)}$	$\Delta_1^{(4)}$	$\Delta_1^{(5)}$
x_2	$f(x_2)$	Δ_2	$\Delta_2^{(2)}$	$\Delta_2^{(3)}$	$\Delta_2^{(4)}$	
x_3	$f(x_3)$	Δ_3	$\Delta_3^{(2)}$	$\Delta_3^{(3)}$		
x_4	$f(x_4)$	Δ_4	$\Delta_4^{(2)}$			
x_5	$f(x_5)$	Δ_5				
x_6	$f(x_6)$					

$$\Delta_i = \frac{f(x_{i+1}) - f(x_i)}{h},\ \Delta_i^{(2)} = \frac{\Delta_{i+1} - \Delta_i}{2h} = \frac{f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)}{2h^2},$$

$$\Delta_i^{(k)} = \frac{\Delta_{i+1}^{(k-1)} - \Delta_i^{(k-1)}}{kh} \qquad \text{dla } k = 1,2,...,n-1,\ i = 1,2,...,n-k$$

dzięki czemu upraszcza się reprezentacja funkcji interpolacyjnej (4.12), gdyż:

$$x_2 = x_1 + h,\ x_k = x_1 + (k-1)h,\ k = 1,2,...,n \tag{4.13}$$

Wprowadzając nową zmienną

$$q = x - x_1,$$

a zatem: $q - h = x - x_2 = x - x_1 - h,\ q - (n-2)h = x - x_{n-1} = x - x_1 - (n-2)h$

otrzymamy z (4.6)

$$P(q) = f(x_1) + q\Delta_1 + q(q-h)\Delta_1^{(2)} + ... + q(q-h)...\big(q-(n-2)h\big)\Delta_1^{(n-1)} \tag{4.14}$$

Nie ma więc potrzeby obliczania współczynników wielomianu (4.3).

Przykład. Określić ogólną postać wielomianu interpolacyjnego Newtona dla funkcji dyskretnej reprezentowanej przez trzy kolejne równooddalone punkty.

W tym przypadku wielomian interpolacyjny (4.14) jest ograniczony do trzech wyrazów:
$P(q) = f(x_1) + q\Delta_1 + q(q-h)\Delta_1^{(2)}$,
co, po podstawieniu odpowiednich wartości, przyjmuje następującą postać:

$$P(q) = f(x_1) + q\frac{f(x_2) - f(x_1)}{h} + q(q-h)\frac{f(x_3) - 2f(x_2) + f(x_1)}{2h^2}$$

Podstawiając $d = q/h$ (d jest w ten sposób względną odległością od początku przedziału), otrzymamy:

$$P(d) = f(x_1) + d\big(f(x_2) - f(x_1)\big) + d(d-1)\frac{f(x_3) - 2f(x_2) + f(x_1)}{2}$$

Po uporządkowaniu otrzymamy:

$$P(d) = \frac{1}{2}\Big(2f(x_1) - d\big(3f(x_1) - 4f(x_2) + f(x_3)\big) + d^2\big(f(x_1) - 2f(x_2) + f(x_3)\big)\Big)$$

W ten sposób, na przykład, wartość funkcji w środku drugiego ($d = 1{,}5$) odcinka można oszacować następująco:

$$\begin{aligned} P(1.5) &= \frac{1}{2}\left(2f(x_1) - \frac{3}{2}\big(3f(x_1) - 4f(x_2) + f(x_3)\big) + \frac{9}{4}\big(f(x_1) - 2f(x_2) + f(x_3)\big)\right) \\ &= \frac{1}{8}\big(3f(x_3) + 6f(x_2) - f(x_1)\big) \end{aligned}$$

4.3. Numeryczne różniczkowanie funkcji dyskretnej

Funkcję interpolującą można wykorzystać do określenia algorytmu numerycznego różniczkowania funkcji dyskretnej. Odpowiednią formułę uzyskuje się przez różniczkowanie funkcji interpolującej:

$$D(x) = \frac{d}{dx}P(x)$$

Na przykład, dla aproksymacji 3-punktowej (jak w Przykładzie 4.1), otrzymamy:

$$\begin{aligned} \frac{d}{dd}P(d) &= \frac{1}{h}\left(f(x_2) - f(x_1) - \frac{f(x_3) - 2f(x_2) + f(x_1)}{2}\right) + 2d\,\frac{f(x_3) - 2f(x_2) + f(x_1)}{2h} \\ &= \frac{-f(x_3) + 4f(x_2) - 3f(x_1)}{2h} + \frac{d\big(f(x_3) - 2f(x_2) + f(x_1)\big)}{h} \end{aligned}$$

Dla różniczkowania na końcu przedziału ($d = 2$) otrzymamy:

$$D(2) = \frac{3f(x_3) - 4f(x_2) + f(x_1)}{2h}$$

Podobnie, w środku przedziału: $D(1) = \dfrac{f(x_3) - f(x_1)}{2h}$.

5. Aproksymacja

5.1. Wprowadzenie

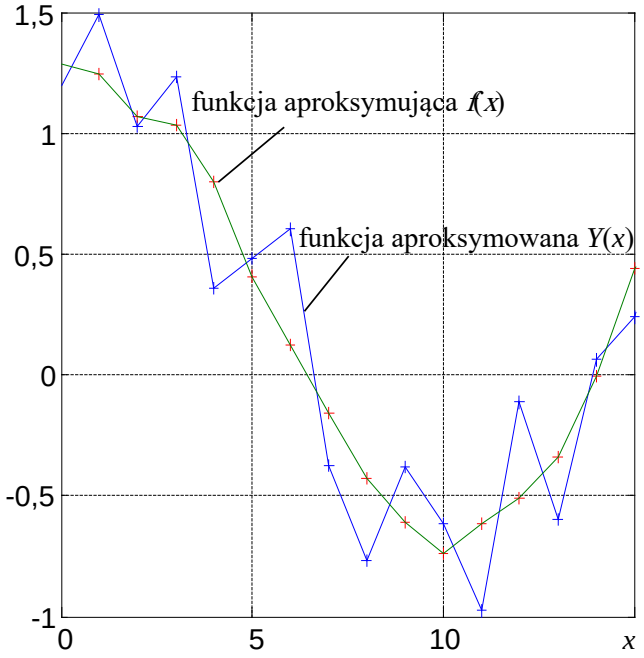
Zadanie aproksymacji polega na przybliżeniu funkcji $Y(x)$ za pomocą innej funkcji $f(x)$, która odnosi się do tego samego obszaru. W praktycznych zastosowaniach inżynierskich $Y(x)$ jest najczęściej funkcją dyskretną $Y(x) = Y(x_i) = y_i$, $i = 0,1,...,M-1$, reprezentującą dane pomiarowe znane dla M różnych wartości zmiennej niezależnej, a $f(x)$ przedstawia model (wzorzec) obserwowanego procesu. Zależność $Y(x)$ jest nazywana *funkcją aproksymowaną*, natomiast $f(x)$ jest *funkcją aproksymującą*. Zakłada się, że dostępne próbki y_i obarczone są błędami, zatem aproksymacja ma na celu najlepsze przybliżenie danych za pomocą funkcji aproksymującej zgodnie z przyjętym kryterium.

Funkcję $f(x)$ wygodnie jest przedstawić w następującej postaci:

$$f(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + ... + a_{N-1}\varphi_{N-1}(x) \tag{5.1}$$

gdzie $\varphi_i(x)$, $i = 0,1,...,N-1$ są funkcjami bazowymi, natomiast a_i, $i = 0,1,...,N-1$ przedstawiają poszukiwane parametry funkcji. Można zauważyć, że funkcja (5.1) jest liniowa względem nieznanych parametrów

Problem ten ilustruje rys. 5.1.



Rys. 5.1. Ilustracja zasady aproksymacji funkcji

Parametry funkcji aproksymującej $f(x)$ są określane na podstawie przyjętego kryterium. Na przykład, żąda się, aby spełnione było minimum kwadratowej normy

różnicy obu funkcji: $\min\left(\left\|Y(x)-f(x)\right\|\right)$. W przypadku funkcji dyskretnej warunek ten jest równoważny kryterium najmniejszych kwadratów:

$$S_2=\left\|Y(x)-f(x)\right\|=\sum_{i=0}^{M-1}\left(Y(x_i)-f(x_i)\right)^2 \tag{5.2}$$

Inne kryterium aproksymacji jest sformułowane za pomocą zależności:

$$S_1=\sup_{x\in\langle a,b\rangle}\left|f(x)-Y(x)\right| \tag{5.3}$$

co oznacza, że poszukiwana funkcja $f(x)$ powinna dać najmniejsze maksimum różnicy pomiędzy daną funkcją i jej aproksymacją. Jest ono znane jako kryterium aproksymacji jednostajnej.

Najszersze zastosowanie praktyczne znalazła aproksymacja według metody najmniejszych kwadratów (MNK). Jest to związane z istnieniem bardzo efektywnych obliczeniowo algorytmów, które wywodzą się z kryterium minimalizacji funkcji (5.2).

5.2. Aproksymacja średniokwadratowa

Załóżmy, że znana jest funkcja $Y(x_i)=y_i$ na zbiorze dyskretnym $x_i, i=0,1,..M-1$ w przedziale $<a,b>$. Chcemy, aby wartości funkcji y_i były przybliżone przez funkcję $f(x)$ o postaci (5.1) w tym samym przedziale.

Jeśli odwołać się do interpretacji pomiarowej, to y_i, $i=0,1,..M-1$ przedstawia zbiór M pomiarów, w stosunku do których zakładamy, że spełnione są następujące zależności:

$$\begin{array}{rccccc} y_0 = & a_0\varphi_0(x_0) & +a_1\varphi_1(x_0) & \cdots & a_{N-1}\varphi_{N-1}(x_0) & +v_0 \\ y_1 = & a_0\varphi_0(x_1) & +a_1\varphi_1(x_1) & \cdots & a_{N-1}\varphi_{N-1}(x_1) & +v_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{M-1} = & a_0\varphi_0(x_{M-1}) & +a_1\varphi_1(x_{M-1}) & \cdots & a_{N-1}\varphi_{N-1}(x_{M-1}) & +v_{M-1} \end{array} \tag{5.4}$$

gdzie wielkości v_i, $i=0,1,..,M-1$ przedstawiają odchyłki (błędy) pomiędzy postulowaną wartością funkcji aproksymującej $f(x_i)$, a dyskretnymi wartościami funkcji aproksymowanej y_i (zmierzonymi wartościami), N jest liczbą składników funkcji aproksymującej (a zatem, liczbą nieznanych współczynników a_i, i=0,1,..,N-1).

Dalsze rozważania wygodnie jest prowadzić, korzystając z zapisu macierzowego. Układ równań (5.4) przyjmie następującą formę:

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{M-1} \end{bmatrix} = \begin{bmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_{N-1}(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_{N-1}(x_1) \\ \vdots & \vdots & \vdots & \vdots \\ \varphi_0(x_{M-1}) & \varphi_1(x_{M-1}) & \cdots & \varphi_{N-1}(x_{M-1}) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \end{bmatrix} + \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{M-1} \end{bmatrix} \tag{5.5}$$

co w ogólnym zapisie wygląda następująco:

$$\mathbf{y} = \mathbf{Ha} + \mathbf{v} \tag{5.6}$$

gdzie:

$\mathbf{y} = [y_0\ y_1 \ldots y_{M-1}]^{\mathrm{T}}$ - wektor reprezentujący aproksymowaną funkcję dyskretną,

$\mathbf{H} = [\mathbf{h}(0)\quad \mathbf{h}(1)\ \ldots\ \mathbf{h}(M\text{-}1)]^{\mathrm{T}}$ - macierz modelu określonego przez funkcje bazowe:

$$\mathbf{h}(i) = [\varphi_0(x_i)\quad \varphi_1(x_i)\quad \cdots\quad \varphi_{N-1}(x_i)],\ i = 0,1,..M-1,$$

$\mathbf{v} = [v_0\ v_1 \ldots v_{M-1}]^{\mathrm{T}}$ - wektor błędów pomiarowych,

$\mathbf{a} = [a_0\quad a_1\quad \cdots\quad a_{N-1}]^{\mathrm{T}}$ - wektor estymowanych parametrów.

Można zauważyć, że odchyłki pomiędzy danymi wartościami aproksymowanej funkcji $Y(x_i) = y_i$, a wartościami postulowanej funkcji aproksymującej $f(x_i)$ można określić następująco:

$$v_{\mathrm{i}} = F(x_i)\text{-}f(x_i) = y_i - \mathbf{h}(i)\cdot\mathbf{a},\ i = 0,..,M\text{-}1 \tag{5.7}$$

Funkcja kryterialna $S_2 = S_2(\mathbf{a})$ (5.2) jest określona, jako suma kwadratów odchyłek (błędów) (5.7), co można zapisać w następującej postaci:

$$S_2(\mathbf{a}) = \sum_{i=0}^{M-1} v_i^2 = \sum_{i=0}^{M-1}\left(F(x_i)\text{-}f(x_i)\right)^2 = \mathbf{v}^T\mathbf{v}, \tag{5.8}$$

przy czym, na podstawie (5.6):

$$\mathbf{v} = \mathbf{y} - \mathbf{Ha}$$

Funkcja (5.8) osiąga minimum, gdy jej pochodne względem parametrów, określonych przez wektor współczynników $\mathbf{a}$, przyjmują zerowe wartości:

$$\begin{aligned}
&\frac{\partial S_2(\mathbf{a})}{\partial a_0} = \frac{\partial}{\partial a_0}\sum_{i=0}^{M-1}\left(F(x_i) - f(x_i)\right)^2 = 0\\
&\frac{\partial S_2(\mathbf{a})}{\partial a_1} = \frac{\partial}{\partial a_1}\sum_{i=0}^{M-1}\left(F(x_i) - f(x_i)\right)^2 = 0\\
&\qquad\qquad \text{fi}\cdot\\
&\frac{\partial S_2(\mathbf{a})}{\partial a_{N-1}} = \frac{\partial}{\partial a_{N-1}}\sum_{i=0}^{M-1}\left(F(x_i) - f(x_i)\right)^2 = 0
\end{aligned} \tag{5.9}$$

co można zapisać w postaci wektorowej:

$$\frac{\partial S_2(\mathbf{a})}{\partial \mathbf{a}} = \frac{\partial}{\partial \mathbf{a}}\left((\mathbf{y} - \mathbf{Ha})^T(\mathbf{y} - \mathbf{Ha})\right) = 2\mathbf{H}^{\mathrm{T}}\mathbf{Ha}\text{-}2\mathbf{H}^{\mathrm{T}}\mathbf{y} = 0 \tag{5.10}$$

Korzysta się tu z zależności:

$$\begin{aligned}
(\mathbf{y} - \mathbf{Ha})^T(\mathbf{y} - \mathbf{Ha}) &= \left(\mathbf{y}^T - \mathbf{a}^T\mathbf{H}^T\right)(\mathbf{y} - \mathbf{Ha}) = \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{Ha} - \mathbf{a}^T\mathbf{H}^T\mathbf{y} + \mathbf{a}^T\mathbf{H}^T\mathbf{Ha}\\
&= \mathbf{y}^T\mathbf{y} - 2\mathbf{a}^T\mathbf{H}^T\mathbf{y} + \mathbf{a}^T\mathbf{H}^T\mathbf{Ha}
\end{aligned} \tag{5.11}$$

Ostatnia równość wynika z faktu, że wielkość $S_2(\mathbf{a})$ jest skalarem, a więc także:

$\mathbf{y}^T\mathbf{Ha} = \mathbf{a}^T\mathbf{H}^T\mathbf{y} = p$,

a więc:

$$\frac{\partial}{\partial \mathbf{a}}\left(\mathbf{y}^T\mathbf{Ha} - \mathbf{a}^T\mathbf{H}^T\mathbf{y}\right) = 2\frac{\partial}{\partial \mathbf{a}}\left(\mathbf{a}^T\mathbf{H}^T\mathbf{y}\right) = 2\mathbf{H}^T\mathbf{y}$$

p – wielkość skalarna.

Podobnie, różniczkując ostatni składnik w (5.11), otrzymamy:

$$\frac{\partial}{\partial \mathbf{a}}\left(\mathbf{a}^T\mathbf{H}^T\mathbf{Ha}\right) = \mathbf{H}^T\mathbf{Ha} + \mathbf{a}^T\mathbf{H}^T\mathbf{H} = 2\mathbf{H}^T\mathbf{Ha}.$$

Zatem, z (5.10) uzyskuje się następującą równość:

$$\mathbf{H}^T\mathbf{Ha} = \mathbf{H}^T\mathbf{y} \tag{5.12}$$

Zauważmy, że macierz $\mathbf{H}^T\mathbf{H}$ jest kwadratowa o wymiarze N, wyrażenie z prawej strony (5.12) jest wektorem o długości N, a N-elementowy wektor $\mathbf{a}$ zawiera poszukiwane współczynniki aproksymującej funkcji (5.1). Równanie (5.12) przedstawia zatem klasyczny liniowy układ równań z N niewiadomymi. Można go rozwiązać jedną ze znanych metod.

Równanie w postaci (5.12) jest nazywane równaniem normalnym. Formalnie, dla warunku $M \geq N$, jego rozwiązanie można zapisać w postaci:

$$\mathbf{a} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{y} \tag{5.13}$$

gdzie: macierz prostokątna $\mathbf{H}^+ = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T$ jest nazywana macierzą pseudo-odwrotną (macierzą Moore'a-Penrose'a). W przypadku, gdy $M = N$, macierz pseudo-odwrotna $\mathbf{H}^+$ w (5.13) odpowiada macierzy odwrotnej $\mathbf{H}^{-1}$ (jeśli macierz $\mathbf{H}$ jest nieosobliwa).

Niektóre właściwości macierzy pseudo-odwrotnej:

$\mathbf{H}^+\mathbf{H} = \left(\mathbf{H}^+\mathbf{H}\right)^T$, $\quad \mathbf{HH}^+ = \left(\mathbf{HH}^+\right)^T$,

$\mathbf{H}^+\mathbf{HH}^+ = \mathbf{H}^+$, $\quad \mathbf{HH}^+\mathbf{H} = \mathbf{H}$.

Przykład

Dane są cztery punkty na płaszczyźnie (x,y): (-2, 20), (1, 2), (2, 7), (3, 12) – rys. 5.2. Określić parabolę, która najlepiej, w sensie kryterium najmniejszych kwadratów, aproksymuje podaną funkcję dyskretną.

Funkcja aproksymująca jest określona w postaci wielomianu drugiego stopnia:

$f(x) = ax^2 + bx + c$

Podstawiając kolejne punkty do powyższego równania, otrzymamy następujący nadokreślony układ równań:

$4a - 2b + c = 20$

$a + 2b + c = 2$

$4a + 2b + c = 7$

$9a + 3b + c = 12$

którego postać macierzowa jest następująca:

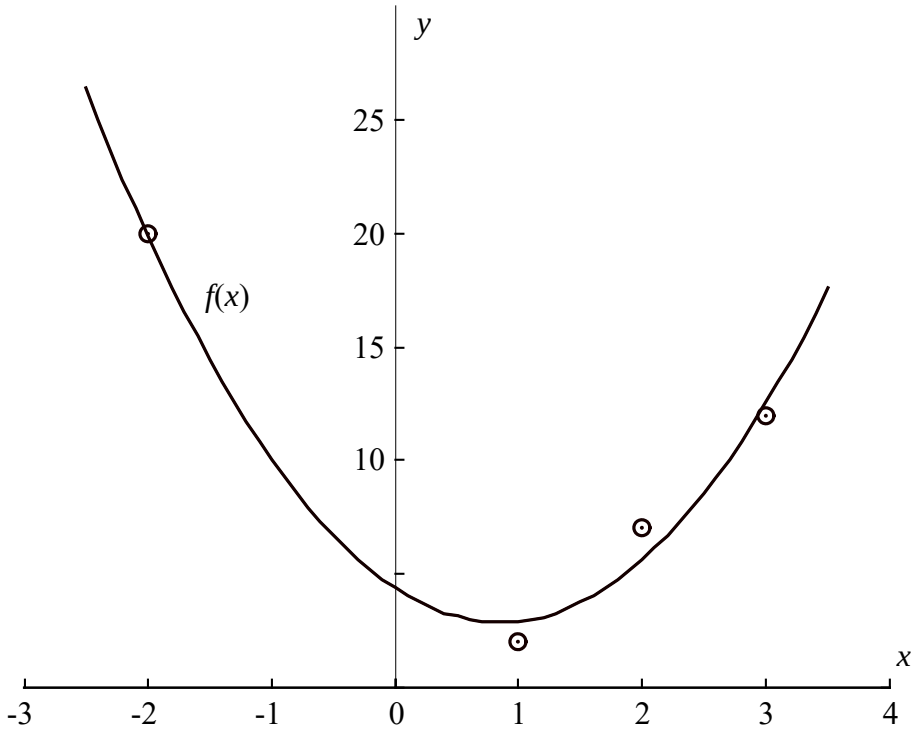
$$\begin{bmatrix} 4 & -2 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 20 \\ 2 \\ 7 \\ 12 \end{bmatrix}$$

Stosując zależność (5.13) otrzymamy:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \left(\begin{bmatrix} 4 & 1 & 4 & 9 \\ -2 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & -2 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 4 & 1 & 4 & 9 \\ -2 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 20 \\ 2 \\ 7 \\ 12 \end{bmatrix}$$

$$= \frac{1}{1448} \begin{bmatrix} 92 & -196 & -68 & 172 \\ -376 & 140 & 152 & 84 \\ 324 & 1104 & 516 & -496 \end{bmatrix} \begin{bmatrix} 20 \\ 2 \\ 7 \\ 12 \end{bmatrix} = \frac{1}{362} \begin{bmatrix} 759 \\ -1292 \\ 1587 \end{bmatrix} = \begin{bmatrix} 2{,}097 \\ -3{,}569 \\ 4{,}384 \end{bmatrix}$$

Przebieg uzyskanej funkcji aproksymującej jest pokazany na rys. 5.2.



Rys. 5.2. Aproksymacja funkcji za pomocą paraboli

W ogólnym przypadku, niektóre pomiary reprezentowane przez aproksymowaną funkcję mogą być bardziej wiarygodne od innych. Wówczas ich wpływ na przebieg obliczanej funkcji aproksymującej powinien być większy. Można to uwzględnić przez wprowadzenie współczynników wagowych w_i do funkcji kryterialnej (5.8):

$$S_2(\mathbf{a}) = \sum_{i=0}^{M-1} w_i v_i^2 = \sum_{i=0}^{M-1} w(x_i)\left(F(x_i) - f(x_i)\right)^2 = \mathbf{v}^T \mathbf{W} \mathbf{v}\,, \tag{5.14}$$

gdzie: $\mathbf{W}$ jest wagową macierzą kwadratową diagonalną o wymiarach $M \times M$; na jej przekątnej leżą współczynniki $w(x_i)$, których wartości są zwykle normalizowane:

$0 \le w(x_i) \le 1$. Przy braku informacji o wspomnianej wiarygodności pomiarów, współczynniki wagowe przyjmują wartość 1 ($\mathbf{W}$ jest wówczas macierzą jednostkową).

Uwzględnienie macierzy wagowej w (5.14) prowadzi do następującej postaci równania (5.12):

$$\mathbf{H}^{\mathrm{T}}\mathbf{WHa} = \mathbf{H}^{\mathrm{T}}\mathbf{Wy} \tag{5.15}$$

oraz, odpowiednio:

$$\mathbf{a} = \left(\mathbf{H}^{\mathrm{T}}\mathbf{WH}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{Wy} \tag{5.16}$$

Metoda najmniejszych kwadratów, zgodnie z którą współczynniki funkcji aproksymującej są określane według (5.13) lub (5.16), jest bardzo użytecznym i praktycznym narzędziem w wielu zastosowaniach. Niektóre z nich są rozpatrywane poniżej.

5.3. Filtr wygładzający

Problem wygładzania danych pomiarowych polega na przybliżeniu obserwowanych (mierzonych) parametrów procesu za pomocą przyjętych zależności, które stanowią model tego procesu. Odchylenia od modelu są traktowane jako zakłócenia. Aby uzyskać poprawne wygładzanie danych pomiarowych, przyjęty model powinien być zbliżony do przebiegu obserwowanego procesu (funkcja aproksymująca powinna mieć dostatecznie dużo stopni swobody), a jednocześnie nie powinien on być ściśle dopasowany do danych empirycznych (zakłócenia nie powinny być odzwierciedlone w modelu).

Załóżmy, że funkcja $Y(x)$ przedstawia proces, który jest obserwowany poprzez próbkowanie określonego parametru i dostępnych jest M ostatnich jego próbek: $Y(x_i) = y_i$, $i = 0,1,...,M-1$. Proces ten jest reprezentowany (modelowany) za pomocą funkcji aproksymującej w postaci wielomianu stopnia $N-1 < M$:

$$f(x_i) = a_0 + a_1 x_i + ... + a_{N-1} x_i^{N-1} \tag{5.17}$$

Dla uproszczenia załóżmy, że próbkowanie odbywa się ze stałym okresem T. Ponieważ dostępne są próbki w oknie pomiarowym o szerokości M, więc aproksymacja jest równoważna filtracji nierekursywnej w tym właśnie oknie. Odpowiedni filtr jest określony następującym równaniem:

$$g_{x_m}(k) = \sum_{i=0}^{M-1} h(i)\, y_{k-M+i+1} = \mathbf{hy}(k) \tag{5.18}$$

gdzie x_m oznacza wartość zmiennej niezależnej względem której określona jest k-ta próbka odpowiedzi filtru, $0 \le x_m \le (M-1)T$; $\mathbf{h} = \begin{bmatrix} h(0) & h(1) & ... & h(M-1) \end{bmatrix}$ - wektor współczynników filtru; $\mathbf{y}(k) = \begin{bmatrix} y_{k-M+1} & y_{k-M+2} & ... & y_k \end{bmatrix}^T$ - wektor zawierający ostatnie M próbek sygnału wejściowego.

Współczynniki filtru (5.18) należy tak dobrać, aby funkcja $g_{x_m}(k)$ aproksymowała przebieg dyskretny określony przez wektor $\mathbf{y}(k)$ w punkcie x_m, licząc od początku przedziału, zgodnie z modelem $f(x)$ (5.17) według kryterium najmniejszych kwadratów.

Zgodnie z przedstawionym opisem funkcja $f(x)$ aproksymuje mierzony dyskretny przebieg według następującej relacji:

$$f(x_i)\big|_{x_i=iT} = y_i + v_i \qquad \text{- w punktach odpowiadających kolejnym próbkom.} \tag{5.19}$$

Zakłada się zatem, że funkcja będzie aproksymowana tylko w węzłach odpowiadających punktom próbkowania.

Stosowne zależności dla metody najmniejszych kwadratów można napisać przy założeniu, że punkt odpowiadający zmiennej x_m znajduje się w początku układu współrzędnych ($x_m = 0$). Wówczas dla jednego zbioru M próbek otrzymamy:

$$\begin{aligned}
&a_0 + a_1(-x_m) + a_2(-x_m)^2 + ... + a_{N-1}(-x_m)^{N-1} = y_0 \\
&a_0 + a_1(-x_m + T) + a_2(-x_m + T)^2 + ... + a_{N-1}(-x_m + T)^{N-1} = y_1 \\
&\quad ... \\
&a_0 + a_1(-x_m + (m-1)T) + a_2(-x_m + (m-1)T)^2 + ... + a_{N-1}(-x_m + (m-1)T)^{N-1} = y_{m-1} \\
&a_0 + a_1(0) + a_2(0)^2 + ... + a_{N-1}(0)^{N-1} = y_m \\
&\quad ... \\
&a_0 + a_1\big((M-1)T - x_m\big) + a_2\big((M-1)T - x_m\big)^2 + ... + a_{N-1}\big((M-1)T - x_m\big)^{N-1} = y_{M-1}
\end{aligned} \tag{5.20}$$

W punkcie odniesienia równanie modelu ma zatem następującą postać: $a_0 = y_m$. Jest to wynikiem odpowiedniego przesunięcia osi czasu, jednak takie założenie jest pomocne dla uproszczenia kolejnych kroków procedury syntezy filtru.

Równania (5.20) można zapisać w postaci macierzowej:

$$\mathbf{Aa} = \mathbf{y} \tag{5.21}$$

gdzie, jeśli dla uproszczenia przyjąć $T = 1$:

$$\mathbf{A} = \begin{bmatrix} 1 & -m & (-m)^2 & ... & (-m)^{N-1} \\ 1 & (1-m) & (1-m)^2 & ... & (1-m)^{N-1} \\ .. & ... & ... & ... & ... \\ 1 & 0 & 0 & ... & 0 \\ .. & ... & ... & ... & ... \\ 1 & (N-m-1) & (N-m-1)^2 & ... & (N-m-1)^{N-1} \end{bmatrix}, \quad \mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \\ \vdots \\ y_{M-1} \end{bmatrix}$$

Zgodnie z przyjętymi założeniami, aproksymacja odbywa się w odniesieniu do m-tej próbki w oknie pomiarowym ($m = 1...M$), co oznacza, że z lewej strony tej próbki znajduje się $n_L = m - 1$ próbek, a z prawej: $n_P = M - m - 1$ próbek. Wielkość n_P określa opóźnienie odpowiedzi algorytmu na sygnał wejściowy i jest nazywane opóźnieniem grupowym [12].

Wektor poszukiwanych współczynników wielomianu jest określony następująco:

$$\mathbf{a} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{y}\,. \tag{5.22}$$

Macierz $\mathbf{A}$ nie zależy od pomiarów, zatem część powyższego równania może być określona przed rozpoczęciem obliczeń. Łatwo zauważyć, że [11]:

$$\left\{a_{ij}\right\} = \left\{\mathbf{A}^T\mathbf{A}\right\} = \sum_{k=n_L}^{n_P} A_{ki}A_{kj} = \sum_{k=n_L}^{n_P} k^{i+j}\,,\ i,j = 0,1,...,N-1\,. \tag{5.23}$$

Wracając do problemu syntezy filtru (5.18) można zauważyć, że sygnał wyjściowy $g_{x_m}(k) = g_m(k)$ jest estymatą próbki $y(m) \approx a_0$ (co wynika ze struktury równania (5.6)). Zatem, równanie (5.22) przedstawia filtr (5.18), jeśli obliczać w nim tylko współczynnik a_0. Kolejne współczynniki filtru są utworzone przez pierwszy wiersz macierzy $\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$. Można je określić zgodnie z następującymi zależnościami:

$$h(0) = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{v}(0)\,,\ h(1) = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{v}(1)\,,\ h(M-1) = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{v}(M-1) \tag{5.24}$$

gdzie: $\mathbf{v}(0) = \begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}^T$, $\mathbf{v}(1) = \begin{bmatrix}0 & 1 & \cdots & 0\end{bmatrix}^T$, $\mathbf{v}(M-1) = \begin{bmatrix}0 & \cdots & 0 & 1\end{bmatrix}^T$.

Obliczanie współczynników filtru (5.24) można uprościć, jeśli zauważyć, że [11]:

$$h(l) = \left\{\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{v}(l)\right\}_0 = \sum_{i=0}^{N-1}\left\{\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\right\}_{0i} l^i \tag{5.25}$$

Utworzony w ten sposób filtr nierekursywny (5.18) aproksymuje zbiór M kolejnych danych pomiarowych, uzyskanych w równych odstępach czasu, według aproksymującej funkcji wykładniczej (5.17).

5.4. Filtr różniczkujący

Filtr służący do określania pierwszej pochodnej funkcji dyskretnej może być łatwo utworzony na podstawie przedstawionego powyżej algorytmu. Można zauważyć, że pochodna funkcji aproksymującej (5.17) ma następującą postać:

$$\frac{df(x_i)}{dx} = a_1 + 2a_2x_i + ... + (N-1)a_{N-1}x_i^{N-2} \tag{5.26}$$

Współczynniki poszukiwanego filtru różniczkującego:

$$d_{x_m}(k) = \sum_{i=0}^{M-1} d(i)\,y_{k-M+i+1} = \mathbf{d}\mathbf{y}(k) \tag{5.27}$$

są zatem określone przez zbiór współczynników a_1 funkcji (5.26). Można je obliczyć zgodnie z (5.25), przy czym, należy wziąć drugi wiersz macierzy $\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$ (oznaczony indeksem 1):

$$d(l) = \left\{\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{v}(l)\right\}_1 = \sum_{i=0}^{N-1}\left\{\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\right\}_{1i} l^i \tag{5.28}$$

W celu uzyskania większej dokładności filtracji (mniejsza wariancja wyników), w charakterze punktu odniesienia należy brać punkt, leżący najbliżej środka okna pomiarowego.

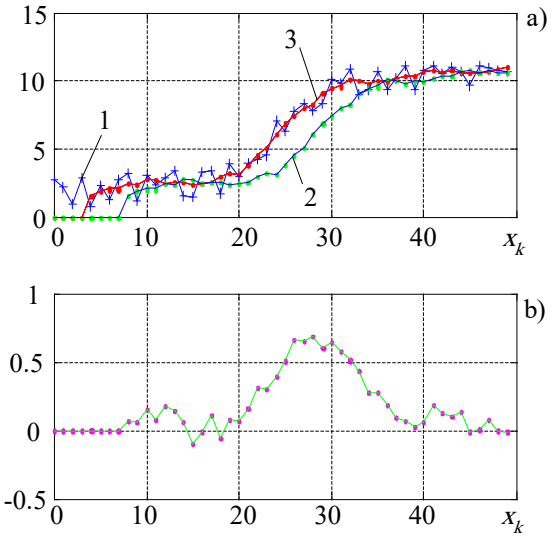
Należy zauważyć, że zarówno filtr wygładzający, jak i różniczkujący, pozwalają aproksymować wejściową funkcję dyskretną (lub jej pochodną) tylko w punktach, odpowiadających momentom próbkowania. Jeśli funkcja aproksymująca powinna być określona dla dowolnych wartości argumentu, to należy stosować postać (5.17), a w związku z tym, powinny być wyznaczone wszystkie współczynniki funkcji a_i, $i = 0,1,...,N-1$.

5.5. Przykład obliczeniowy

Rozważmy problem wygładzania danych pomiarowych i określania pochodnej funkcji reprezentowanej tymi danymi za pomocą odpowiednich filtrów rekursywnych. Zarejestrowany przebieg jest przedstawiony na rys. 5.3a, krzywa 1. Próbkowanie odbywa się ze stałą częstotliwością.

Do wygładzania tego przebiegu zastosowano filtr (5.18), który został zaprojektowany na podstawie funkcji aproksymacyjnej (5.17) 2-go rzędu. Założono, że w oknie przetwarzania filtru znajduje się $M = 9$ próbek. Filtracja jest prowadzona w odniesieniu do środkowej próbki w oknie pomiarowym, a więc $m = 5$.

Po zastosowaniu przedstawionej powyżej procedury uzyskuje się następującą funkcję impulsową filtru wygładzającego i różniczkującego:



Rys. 5.3.

$$\mathbf{h} = \frac{1}{231}\left[-21 \quad 14 \quad 39 \quad 54 \quad 59 \quad 54 \quad 39 \quad 14 \quad -21\right]$$

$$\mathbf{d} = \frac{1}{60}\left[-4 \quad -3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4\right]$$

W wyniku filtracji (5.18) oraz (5.27) otrzymuje się przebiegi wyjściowe (rys. 5.3): wygładzonych danych (krzywa 2) oraz estymaty pochodnej (rys. 5.3b). W celu lepszego porównania przebiegu oryginalnego z wygładzonym (aproksymowanym),

ten ostatni został przesunięty o liczbę próbek stanowiących opóźnienie grupowe (4 próbki, krzywa 3).

Na zakończenie kilka uwag praktycznych dotyczących problemu aproksymacji.

- Przy wyborze funkcji bazowych należy się kierować podobieństwem pomiędzy aproksymowaną funkcją i jej reprezentacją 'wygładzoną'; w charakterze funkcji bazowych najczęściej wybiera się szereg potęgowy lub trygonometryczny.
- Liczba wyrazów w funkcji aproksymacyjnej decyduje o dokładności odwzorowania oryginalnego przebiegu; niski rząd tej funkcji powoduje zgrubne przybliżenie, dzięki czemu efekt filtracji jest bardziej wyrazisty.
- Kwestia ta łączy się również z szerokością okna przetwarzania (liczba M jednocześnie branych pod uwagę próbek funkcji oryginalnej): im szersze jest okno pomiarowe, tym lepsze jest wygładzanie danych. Przy szerokim oknie pomiarowym można także stosować funkcje aproksymujące wyższego rzędu z zachowaniem wierności odtwarzania. Niestety, zwiększenie długości okna pomiarowego prowadzi do zwiększenia opóźnienia grupowego, co ma istotne znaczenie wówczas, gdy aproksymacja odbywa się bezpośrednio w czasie pomiarów. Związana z tym zwłoka czasowa sprawia, że informacja o stanie nadzorowanego procesu jest dostępna z określonym opóźnieniem.

W przypadku filtracji sygnałów, model użyty do projektowania filtrów wygodnie jest tworzyć na bazie funkcji trygonometrycznych sinus/kosinus. Można wówczas łatwo uzyskać procedury do pomiaru amplitudy i fazy mierzonych sygnałów lub ich harmonicznych [12].

5.6. Metoda Najmniejszych Kwadratów z wykorzystaniem rozkładu macierzy według wartości szczególnych – SVD

Dowolna macierz $\mathbf{A}$ $(m \times n)$ może być przedstawiona w następującej postaci:

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T \tag{5.29}$$

gdzie:

$\mathbf{U}$ - macierz $(m \times m)$, której kolumny spełniają następującą zależność:

$\sum_{i=1}^{m} u_{ik} u_{ij} = 1$, $1 \le k \le n$, $1 \le j \le n$, to znaczy, są wzajemnie ortogonalne;

$\mathbf{W} = \begin{bmatrix} \mathbf{\Sigma} & 0 \\ 0 & 0 \end{bmatrix}$ - macierz $(m \times n)$ wartości szczególnych, przy czym:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \cdots & \\ & & & \sigma_r \end{bmatrix},\ \sigma_1 \ge \sigma_2 \ge \text{fi} \cdot \ge \sigma_r > 0,\ r \le \min(m,n) \text{ – rząd macierzy } \mathbf{A}^{*}; \tag{5.30}$$

$\mathbf{V}$ - macierz kwadratowa $(n \times n)$, której kolumny spełniają następującą zależność:

[*] Rząd macierzy r jest największą liczbą niezależnych wierszy (lub kolumn) macierzy.

$\sum_{i=1}^{n} v_{ik} v_{ij} = 1$, $1 \le k \le n$, $1 \le j \le n$, a więc są również wzajemnie ortogonalne.

Z powyższego opisu wynika, że:

$$\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{1} \tag{5.31}$$

Kolumny macierzy $\mathbf{U}$ są wektorami własnymi macierzy kwadratowej $\mathbf{A}\mathbf{A}^T$, natomiast kolumny macierzy $\mathbf{V}$ są wektorami własnymi macierzy $\mathbf{A}^T\mathbf{A}$. Wartości szczególne rozkładu (5.29) –a więc elementy diagonalne macierzy $\mathbf{\Sigma}$ - są natomiast pierwiastkami kwadratowymi wartości własnych macierzy $\mathbf{A}\mathbf{A}^T$ lub $\mathbf{A}^T\mathbf{A}$. Zależność (5.29) jest nazywana rozkładem macierzy $\mathbf{A}$ według wartości szczególnych (ang. SVD – *Singular Value Decomposition*). Wynika stąd sposób obliczania macierzy rozkładu (5.29).

Wektor własny $\mathbf{x}$ macierzy $\mathbf{H}$ spełnia następujące równanie:

$$\mathbf{H}\mathbf{x} = \lambda\mathbf{x} \tag{5.32}$$

przy czym λ jest wartością własną macierzy $\mathbf{H}$. W takim przypadku mówi się, że wektor $\mathbf{x}$ jest skojarzony w wartością własną λ.

W celu określenia wartości własnych λ oraz odpowiadających im wektorów własnych $\mathbf{x}$ należy rozwiązać równanie (5.32), co jest równoważne następującej zależności:

$$(\mathbf{H} - \lambda\mathbf{1})\mathbf{x} = 0 \tag{5.33}$$

Jednoznaczne rozwiązanie (5.33) uzyskuje się wówczas, gdy spełniony jest warunek:

$$\det(\mathbf{H} - \lambda\mathbf{1}) = \begin{vmatrix} h_{11}-\lambda & h_{12} & \cdots & h_{1m} \\ h_{21} & h_{22}-\lambda & \cdots & h_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ h_{m1} & h_{m2} & \cdots & h_{mm}-\lambda \end{vmatrix} = 0 \tag{5.34}$$

Rozwiązanie równania (5.34) jest równoważne znalezieniu pierwiastków wielomianu m-tego stopnia względem λ. W ogólnym przypadku jest zatem m wartości własnych: $\lambda_1, \lambda_2, .. \lambda_m$. Podstawiając kolejno te wartości własne do (5.33) otrzymuje się m równań:

$$\begin{bmatrix} h_{11}-\lambda_i & h_{12} & h_{1m} \\ h_{21} & h_{22}-\lambda_i & h_{2m} \\ & & \\ h_{m1} & h_{m2} & h_{mm}-\lambda_i \end{bmatrix} \begin{bmatrix} x_{1i} \\ x_{2i} \\ \\ x_{mi} \end{bmatrix} = 0, \quad i = 1,2,..,m, \tag{5.35}$$

po rozwiązaniu których uzyskuje się m wektorów własnych: $\mathbf{x}_i = \begin{bmatrix} x_{1i} & x_{2i} & x_{mi} \end{bmatrix}^T$, $i = 1,2,..,m$.

W rozpatrywanym przypadku macierz $\mathbf{H} = \mathbf{A}\mathbf{A}^T$ (jeśli obliczana jest macierz $\mathbf{U}$) lub $\mathbf{H} = \mathbf{A}^T\mathbf{A}$ (jeśli obliczana jest macierz $\mathbf{V}$). Macierz $\mathbf{\Sigma}$ (5.30) powstaje przez uporządkowanie pierwiastków z wartości własnych macierzy $\mathbf{H}$. W standardowych programach do rozkładu macierzy według wartości szczególnych stosowane są

zazwyczaj inne, bardziej efektywne algorytmy w stosunku do przedstawionego powyżej.

Właściwości rozkładu SVD:

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T,\ \mathbf{A}^T = \mathbf{V}\mathbf{W}^T\mathbf{U}^T \tag{5.36}$$

$$\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{W}^T\mathbf{U}^T\mathbf{U}\mathbf{W}\mathbf{V}^T = \mathbf{V}\mathbf{W}^T\mathbf{W}\mathbf{V}^T,\ \ \mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{W}\mathbf{W}^T\mathbf{U}^T \tag{5.37}$$

$$\mathbf{A}^{\#} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T = \mathbf{V}\mathbf{W}^{-1}\mathbf{U}^T \text{ - macierz pseudo-odwrotna do } \mathbf{A} \tag{5.38}$$

gdzie $\mathbf{W}^{-1} = \mathbf{W}^{\#} = \begin{bmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{bmatrix}_{(n\times m)}$ - macierz pseudo-odwrotna do $\mathbf{W}$; elementy diagonalne macierzy $\mathbf{\Sigma}^{-1}$ można obliczyć, jako odwrotności odpowiednich elementów diagonalnych macierzy $\mathbf{\Sigma}$.

Ostatnią zależność można bezpośrednio wykorzystać do rozwiązywania zadań MNK:

$$\mathbf{x} = \mathbf{H}^{\#}\mathbf{y} = \mathbf{V}\mathbf{W}^{\#}\mathbf{U}^T\mathbf{y} \tag{5.39}$$

gdzie: $\mathbf{y}$ - wektor pomiarów, $\mathbf{x}$ - wektor poszukiwanych parametrów funkcji aproksymującej.

Zastosowanie z użyciem programu MATLAB:

[U,W,V]=svd(A); rozkład macierzy **A** według wartości szczególnych,

[C,D]=eig(A*A'); wartości własne macierzy **A***A'.

6. Całkowanie numeryczne

6.1. Wprowadzenie

Załóżmy, że należy obliczyć przybliżoną wartość całki oznaczonej:

$$I(f) = \int_a^b f(x)dx \tag{6.1}$$

na odcinku $< a, b >$.

W celu wyprowadzenia odpowiedniej formuły obliczeniowej przyjmijmy, że odcinek $< a, b >$ jest podzielony na n przedziałów elementarnych $< x_i, x_{i+1} >$, $i = 1, 2, ..., n$, przy czym $x_1 = a$, $x_{n+1} = b$ oraz $x_1 < x_2 < ... < x_{n+1}$. Długość i-tego odcinka oznaczymy przez h_i:

$$h_i = x_{i+1} - x_i \tag{6.2}$$

Wartość całki (6.1) można przedstawić w postaci sumy całek na elementarnych odcinkach:

$$I(f) = \sum_{i=1}^{n} I_i \tag{6.3}$$

przy czym

$$I_i = I_i(f) = \int_{x_i}^{x_{i+1}} f(x)dx$$

Numeryczne przybliżanie wartości całek na danym odcinku nazywa się kwadraturą.

6.2. Metoda Simpsona

Najprostsza kwadratura powstaje przez przyjęcie, że całka na elementarnym odcinku $< x_i, x_{i+1} >$ jest równa polu trójkąta wyznaczonego przez boki: h_i oraz wartości funkcji w środku przedziału:

$$y_i = \frac{x_i + x_{i+1}}{2}$$

Zatem:

$$I_i \approx h_i f(y_i) = h_i f\left(\frac{x_i + x_{i+1}}{2}\right) \tag{6.4}$$

lub

$$I_i \approx h_i f\left(x_i + \frac{h_i}{2}\right) \tag{6.5}$$

Jeśli funkcja podcałkowa dana jest tylko w węzłach x_i, to wówczas należy stosować formułę:

$$I_{Pi} \approx h_i f(x_i) \tag{6.6}$$

Zwiększenie dokładności obliczeń można uzyskać przez prostą operację zamiany prostokąta do obliczania pola pod krzywą na trapez. Uzyskuje się wówczas następującą zależność:

$$I_{Ti} \approx h_i \frac{f(x_i) + f(x_{i+1})}{2} \tag{6.7}$$

Metodą kombinacji liniowej obu metod otrzymuje się znacznie bardziej dokładną metodę Simpsona:

$$I_{Si} = \frac{2}{3} I_{Pi} + \frac{1}{3} I_{Ti} = \frac{1}{3} h_i \left(f(x_i) + 4 f\left(\frac{x_i + x_{i+1}}{2}\right) + f(x_{i+1}) \right) \tag{6.8}$$

Dla przypadku, gdy funkcja jest dostępna tylko w węzłach x_i powyższa formuła przyjmuje następującą postać:

$$I_{Si} = \frac{1}{3} h_i \left(f(x_i) + 4 f(x_{i+1}) + f(x_{i+2}) \right) \tag{6.9}$$

7. Numeryczne rozwiązywanie równań różniczkowych zwyczajnych

7.1. Wprowadzenie

Numeryczne metody rozwiązywania równań różniczkowych są stosowane w takich przypadkach, gdy rozwiązania w postaci analitycznej nie są znane. Ponieważ jednak tylko dla nielicznych równań można podać analityczne rozwiązanie, więc metody numeryczne są w takich przypadkach bardzo pomocnym narzędziem poszukiwania rozwiązania. Ważnym obszarem zastosowania tych metod są numeryczne modele układów i zjawisk dynamicznych, które służą do ich komputerowej symulacji.

W odróżnieniu jednak od metod analitycznych, w metodach numerycznych zakłada się, że zmienna niezależna dostępna jest tylko w wybranych, dyskretnych wartościach. Konsekwencją tego jest nieuchronne przybliżenie rozwiązania.

Rozpatrzmy równanie różniczkowe pierwszego rzędu:

$$y' = \frac{dy(t)}{dt} = f(y,t) \tag{7.1}$$

Równanie tego typu ma, w ogólnym przypadku, rodzinę rozwiązań. Na przykład, równanie:

$$y'(t) = -y(t)$$

ma następujące rozwiązanie ogólne

$$y(t) = Ce^{-t},$$

gdzie C jest dowolną stałą.

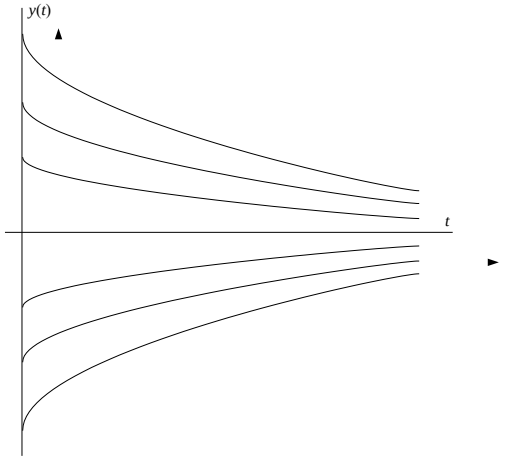
Konkretne rozwiązanie jest związane z tą trajektorią, na której znajduje się jakieś rozwiązanie, spełniające określone wymagania, na przykład, warunki początkowe: $y(0) = y_0$ (rys. 7.1). Można wówczas wyznaczyć stałą C:

$y_0 = Ce^{-t}\Big|_{t=0} = C$, zatem: $C = y_0$.

W wielu przypadkach obiekt (zjawisko) jest opisywane za pomocą większej liczby równań różniczkowych. Wówczas do jednoznacznego określenia trajektorii rozwiązania potrzebne są odpowiednie warunki dla każdego z równań. Jednoznaczne rozwiązania można uzyskać, jeśli warunki te są dane dla tej samej wartości zmiennej niezależnej *t*. Na przykład, dla układu dwóch równań różniczkowych:

$$x' = f(x,z,t)$$
$$z' = g(x,z,t)$$

potrzebne są dwa warunki początkowe: $x(t_0)$, $z(t_0)$.



Rys. 7.1.

Układ równań różniczkowych można zapisać w postaci macierzowej:

$$\frac{d\mathbf{y}}{dt} = F(\mathbf{y},t) \tag{7.2}$$

gdzie dla przypadku dwóch równań:

$$\mathbf{y} = \begin{bmatrix} x(t) \\ z(t) \end{bmatrix},\ F(\mathbf{y},t) = \begin{bmatrix} f(x,z,t) \\ g(x,z,t) \end{bmatrix}$$

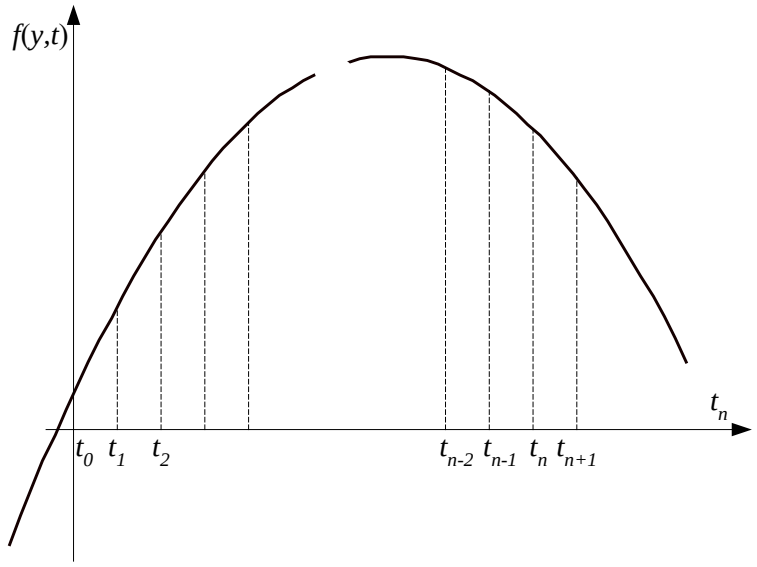
Łatwo można pokazać, że równanie *n*-tego rzędu można sprowadzić do *n* równań pierwszego rzędu. Na przykład

równanie: $\frac{d^2u}{dt^2} = g(u,u',t)$ można zapisać w postaci następujących równań:

$z' = g(u,z,t),$

$u' = z.$

Podstawowy sposób numerycznego rozwiązywania równań różniczkowych polega na zastosowanie jakiejś numerycznej procedury całkowania obu stron tego równania. W tym celu, dla równania o postaci (7.1), funkcja wymuszająca $f(y,t)$ (prawa strona równania) powinna zostać poddana dyskretyzacji, co oznacza, że zmienna niezależna t przyjmuje wartości dyskretne t_0, t_1, t_2, ... W zależności od sposobu dyskretnej reprezentacji funkcji wymuszającej (może tu być zastosowana interpolacja lub aproksymacja), powstają różne metody rozwiązywania równań różniczkowych. Ogólnie, metody te dzielą się na *jednokrokowe* i *wielokrokowe* w zależności od tego, czy do obliczania bieżącej próbki rozwiązania korzysta się z wartości funkcji i rozwiązania odległych o jeden krok do tyłu (metody jednokrokowe), czy też z historii odległej o większą liczbę kroków (metody wielokrokowe).



Rys. 7.2.

7.2. Metody jednokrokowe

Metoda Eulera

Rozpatrzmy równanie (7.1) dla warunków początkowych $y_0 = y(t_0)$. Załóżmy, że poszukiwane jest rozwiązanie tego równania dla $t = t_1$, przy czym: $T = t_1 - t_0$. Poszukiwane rozwiązanie można przedstawić, korzystając z jej rozwinięcia w szereg Taylora w pobliżu punktu początkowego t_0:

$$y_1 = y(t_0 + T) = y(t_0) + Ty'(t_0) + \frac{T^2}{2!}y''(t_0) + \ldots \tag{7.3}$$

Ponieważ, w danym przypadku, dostępna jest tylko pierwsza pochodna poszukiwanej funkcji, więc do przybliżenia wartości $y(t_0 + T)$ można wykorzystać pierwsze dwa składniki rozwinięcia (7.3):

$$y_1 \approx y(t_0) + Ty'(t_0) \tag{7.4}$$

Zgodnie z (7.1), w miejsce pochodnej można podstawić prawą stronę równania różniczkowego (funkcję wymuszającą), co prowadzi do następującej zależności:

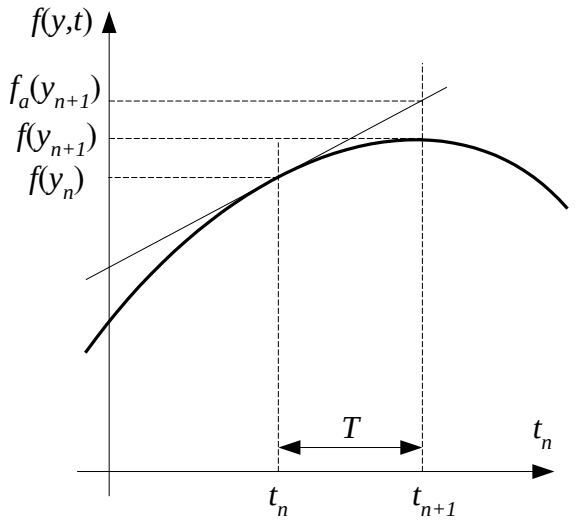
$$y_1 \approx y(t_0) + Tf\left(y(t_0), t_0\right) \tag{7.5}$$

Powtarzając ten wywód dla kolejnych kroków otrzymamy ogólną zależność:

$$y_{n+1} = y_n + Tf(y_n) \tag{7.6}$$

która jest znana jako *jawna metoda Eulera*.

Określenie ‘jawna’ bierze się stąd, że do określenia kolejnej wartości rozwiązania wykorzystuje się znane wartości z poprzedniego okresu próbkowania (dyskretyzacji). Ilustracja tej metody jest pokazana na rys. 7.3. Widać tam błąd wynikający z obcięcia szeregu Taylora: rzeczywista wartość funkcji dla $t = t_{n+1}$ jest równa $f(y_{n+1})$, podczas gdy jej aproksymacja według pierwszej pochodnej wynosi $f_a(y_{n+1})$.



Rys. 7.3.

Zależność (7.6) można także uzyskać korzystając ze wspomnianej metody obustronnego całkowania wyjściowego równania (7.1). Wartość rozwiązania dla $t = t_{n+1}$ można wówczas określić następująco:

$$y_{n+1} = \int_{t_0}^{t_{n+1}} f\big(y(t)\big)dt = \int_{t_0}^{t_n} f\big(y(t)\big)dt + \int_{t_n}^{t_{n+1}} f\big(y(t)\big)dt \tag{7.7}$$

Ponieważ rozwiązanie y_n jest znane (gdyż wcześniejsze kroki: t_0, t_1,.., t_n, zostały już wykonane, a wartość początkowa $f(y_0)$ także jest znana), to:

$$\int_{t_0}^{t_n} f\big(y(t)\big)dt = y_n$$

Pozostaje więc wyznaczyć drugą całkę w (7.7). Przybliżając ją metodą prostokątów (w którym jeden bok jest utworzony przez odcinek *T*, a drugi – przez odcinek $f(y_n)$), otrzymamy zależność (7.6). Ta interpretacja wyjaśnia drugą nazwę rozpatrywanego algorytmu: *metoda prostokątów* (w tym przypadku – jawna, ekstrapolacyjna).

Wartość całki na odcinku $T = t_{n+1} - t_n$ można także przybliżyć prostokątem o boku określonym przez wartość funkcji w bieżącym punkcie t_{n+1}: $f\big(y_{n+1}\big)$. Wówczas, zależność (7.6) przyjmie zastępującą postać:

$$y_{n+1} = y_n + Tf(y_{n+1}) = y_n + Ty'_{n+1} \tag{7.8}$$

która jest znana jako *niejawna metoda Eulera* (prostokątów). Jak widać, nazwa jest uzasadniona tym, że postać funkcji (7.7) jest uwikłana, gdyż z obu stron znaku równości występuje odwołanie do wartości funkcji (lub jej pochodnej) dla tej samej wartości zmiennej niezależnej (czasu). Spotykana jest także inna nazwa: *formuła interpolacyjna Eulera* (prostokątów).

Metoda trapezów

Można zauważyć, że znaki błędów w obu powyższych metodach (jawnej i niejawnej) są przeciwne (niezależnie od przebiegu funkcji). Można to wykorzystać w celu zwiększenia dokładności przybliżenia rozwiązania: uzyskuje się to przez obliczenie średniej z wyników uzyskanych obiema metodami:

$$y_{n+1} = y_n + T\frac{f(y_n)+f(y_{n+1})}{2} = y_n + T\frac{y'_n + y'_{n+1}}{2} \tag{7.9}$$

W ten sposób, całka na odcinku $T = t_{n+1} - t_n$ jest określona przez pole trapezu wyznaczonego przez boki $f(y_n)$ oraz $f(y_{n+1})$. Można zauważyć, że metoda trapezów jest także niejawna.

Metody Rungego-Kutty

W odróżnieniu od przedstawionych powyżej metod jednokrokowych, w metodach Rungego –Kutty nie ma odwołania do pochodnej funkcji, występującej w równaniu różniczkowym. W jej miejsce występuje odpowiednia kombinacja wartości samej funkcji, obliczanej w stosownych miejscach. W przypadku najbardziej znanej metody Rungego-Kutty czwartego rzędu, kolejne przybliżenie rozwiązania jest określane według następującego wzoru:

$$y_{n+1} = y_n + \frac{1}{6}\left(F_1 + 2F_2 + 2F_3 + F_4\right) \tag{7.10}$$

gdzie:

$F_1 = Tf(y_n, t)$, $F_2 = Tf(y_n + F_1/2, t + T/2)$,

$F_3 = Tf(y_n + F_2/2, t + T/2)$, $F_4 = Tf(y_n + F_3, t + T)$.

Jest to metoda 4 rzędu, gdyż błąd przybliżonego wzoru (7.10) wynosi $O(T^5)$. Metoda ma charakter jawny, gdyż obliczana wartość y_{n+1} nie występuje po prawej stronie formuły (7.10).

Realizacja tej metody wymaga wykonania w każdym kroku czasowym obliczeń czterech wartości funkcji prawej strony równania różniczkowego $f(y,t)$, a więc, funkcja ta musi być dostępna w jawnej formie. Przy spełnieniu tych wymagań, omawiana metoda jest prosta w realizacji i zapewnia dużą dokładność rozwiązania. Jest najczęściej stosowną metodą w zastosowaniach inżynierskich i naukowych. Bez żadnych dodatkowych ograniczeń może być stosowana do rozwiązywania układów równań różniczkowych, również nieliniowych. Profesjonalne programy z tą metodą rozwiązywania równań różniczkowych są najczęściej wyposażone w mechanizm automatycznego doboru długości kroku całkowania T, co może przyśpieszyć czas

obliczeń przy zachowaniu założonej dokładności. Niestety, przy stosowaniu tej metody mogą wystąpić problemy ze stabilnością w przypadku układów sztywnych (gdy w modelowanym systemie występują znacznie różniące się stałe czasowe.

Dokładność metody

Omówione powyżej metody można zapisać następująco:

$y'_n = \frac{1}{T}(y_n - y_{n-1})$ jawna metoda prostokątów,

$y'_{n+1} = \frac{1}{T}(y_{n+1} - y_n)$ niejawna metoda prostokątów,

$\frac{1}{2}(y'_{n+1} + y'_n) = \frac{1}{T}(y_{n+1} - y_n)$ metoda trapezów (jest to także metoda niejawna).

Ogólna postać tych związków przybiera następującą formę:

$$a_1 y_{n+1} + a_0 y_n = T(b_1 y'_{n+1} + b_0 y'_n) \tag{7.11}$$

Współczynniki tego równania określają odpowiedni algorytm:

$a_1 = 1, \quad a_0 = -1$ - dla wszystkich metod,

$b_1 = 0, \; b_0 = 1$ - jawna metoda prostokątów,

$b_1 = 1, \; b_0 = 0$ - niejawna metoda prostokątów,

$b_1 = b_0 = \frac{1}{2}$ - metoda trapezów.

Na podstawie równania (7.11) można przeprowadzić dyskusję dokładności rozpatrywanych metod. Analogicznie do (7.3) napiszemy:

$$y_{n+1} = y(t_n + T) = y(t_n) + Ty'(t_n) + \frac{T^2}{2!} y''(t_n) + \frac{T3}{3!} y'''(t_n) + \ldots \tag{7.12}$$

Po zróżniczkowaniu:

$$y'_{n+1} = y'(t_n + T) = y'(t_n) + Ty''(t_n) + \frac{T^2}{2!} y'''(t_n) + \frac{T3}{3!} y''''(t_n) + \ldots \tag{7.13}$$

Podstawiając powyższe zależności do formuły (7.11) otrzymamy:

$$a_1\left(y_n + Ty'_n + \frac{T^2}{2} y''_n + \frac{T3}{6} y'''_n + \ldots\right) + a_0 y_n = Tb_1\left(y'_n + Ty''_n + \frac{T^2}{2} y'''_n + \ldots\right) + Tb_0 y'_n \tag{7.14}$$

skąd:

$$y_n(a_1 + a_0) + Ty'_n(a_1 - b_1 - b_0) = -T^2 y''_n\left(\frac{1}{2}a_1 - b_1\right) - T^3 y'''_n\left(\frac{1}{6}a_1 - \frac{1}{2}b_1\right) - \ldots \tag{7.15}$$

Równanie (7.15) jest spełnione dla dowolnych wartości funkcji i jej pochodnych wtedy, gdy współczynniki określone przez a_0, a_1, b_0, b_1 będą równe zero, a więc:

$$a_1 + a_0 = 0$$

$$a_1 - b_1 - b_0 = 0$$

$$\frac{1}{2}a_1 - b_1 = 0$$

$$\frac{1}{6}a_1 - \frac{1}{2}b_1 = 0$$

Dla metody prostokątów ($a_1 = 1$, $a_0 = -1$ oraz $b_1 = 0$, $b_0 = 1$ lub $b_1 = 1$, $b_0 = 0$), pierwszy niezerowy współczynnik stoi przy drugiej pochodnej funkcji:

$$T^2 y''_n \left(\frac{1}{2} \cdot 1 - 1 \right) = C_2 T^2 y''_n,$$

skąd: $C_2 = \frac{1}{2}$ lub $C_2 = -\frac{1}{2}$.

Podobnie, dla metody trapezów ($a_1 = 1$, $a_0 = -1$ oraz $b_1 = b_0 = \frac{1}{2}$) pierwszy niezerowy współczynnik stoi przy trzeciej pochodnej funkcji:

$$-T^3 y'''_n \left(\frac{1}{6} \cdot 1 - \frac{1}{2} \cdot \frac{1}{2} \right) = \frac{1}{12} T^3 y'''_n + \ldots = C_3 T^3 y'''_n + \ldots$$

skąd: $C_3 = \frac{1}{12}$.

Powyższe związki charakteryzują dokładność metody zgodnie z następującymi regułami.

- Rząd metody p określa się rzędem pochodnej (p+1), dla której pierwszy współczynnik jest różny od zera. Zatem: dla metody prostokątów p=1; dla metody trapezów p=2;
- Błąd odcięcia jest związany z wartościami wyrazów, które nie spełniają równości (1.15). Współczynnik stojący przy najniższej pochodnej, który nie spełnia tego równania jest właśnie błędem odcięcia. Dla metody trapezów jest on równy: $C_3 = C_{p+1} = 1/12$.

Stabilność metody

Do badania stabilności określonego algorytmu numerycznego rozwiązywania równań różniczkowych wygodnie jest posługiwać się wybranym równaniem modelowym. Rozpatrzmy równanie o postaci:

$$y'(t) = \lambda y(t) \tag{7.16}$$

z warunkiem początkowym $y(0) = y_0$, przy czym, współczynnik λ jest w ogólnym przypadku zespolony: $\lambda = u + jw$,

Rozwiązaniem równania (7.16) jest funkcja wykładnicza: $y(t) = e^{\lambda t} = e^{ut} e^{jwt}$.

1. Zastosujmy do rozwiązania równania (7.16) jawną metodę prostokątów. Załóżmy, że dyskretna postać zmiennej niezależnej t dostępna jest ze stałym przedziałem T. Rozwiązanie w kolejnych krokach przybiera następujące wartości:

$y_1 = y_0 + Ty'_0 = y_0 + T\lambda y_0 = (1+T\lambda)y_0$,

$y_2 = y_1 + Ty'_1 = (1+T\lambda)y_0 + T\lambda y_1 = (1+T\lambda)y_0 + T\lambda(1+T\lambda)y_0 = (1+T\lambda)^2 y_0$,...

i ogólnie:

$$y_n = (1+T\lambda)^n y_0 . \tag{7.17}$$

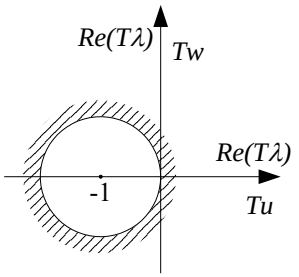
Jeśli wyjściowe równanie ciągłe jest stabilne ($u = \mathrm{Re}(\lambda) < 0$), to uzyskana aproksymacja dyskretna rozwiązania także powinna być stabilna. W odniesieniu do (7.17) jest to równoważne warunkowi:

$$|1+T\lambda| \le 1 \tag{7.18}$$

Zatem:

$|1+Tu+jTw| \le 1$, czyli: $(1+Tu)^2 + (Tw)^2 \le 1$.

Równanie powyższe przedstawia okrąg na płaszczyźnie zespolonej o promieniu jednostkowym i o środku leżącym w punkcie (-1, 0) (jeśli współrzędne są przeskalowane: (Tu, Tw)). Obszar stabilności leży wewnątrz okręgu (rys. 7.4). Można zauważyć, że przy danej wartości λ obszar stabilności zależy od długości przedziału T: im większa jest wartość T tym mniejszy obszar stabilności (granica stabilności: Tu=Tw=1).



Rys. 7.4.

2. Powtórzmy powyższe rozważania dla niejawnej metody prostokątów. Tym razem, rozwiązanie w kolejnych krokach jest określone następująco:

$y_1 = y_0 + Ty'_1$. Stąd: $y_1 = \dfrac{1}{1-T\lambda} y_0$

$y_2 = y_1 + Ty'_2 = \dfrac{1}{1-T\lambda} y_0 + T\lambda y_2$. Stąd: $y_2 = \left(\dfrac{1}{1-T\lambda}\right)^2 y_0$

i ogólnie:

$$y_n = \left(\frac{1}{1-T\lambda}\right)^n y_0 . \tag{7.19}$$

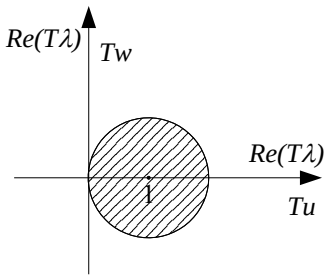
Ciąg ten jest ograniczony dla:

$$\left|\frac{1}{1-T\lambda}\right| \le 1, \tag{7.20}$$

co jest równoważne :

$$(1-Tu)^2 + (Tw)^2 \ge 1$$

Równanie to jest spełnione dla całej płaszczyzny zespolonej za wyjątkiem wnętrza okręgu (przy przeskalowanych osiach *Tu*, *Tw*) o środku w punkcie (1,0) – rys. 7.5. A zatem, przy danej wartości λ, procedura mogłaby być niestabilna dla małych wartości *T*. Jednak, w rzeczywistych warunkach, zawsze $u \le 0$ (warunek stabilności równania ciągłego), co oznacza, że niejawna procedura Eulera numerycznego rozwiązywania równania (7.16) jest globalnie stabilna, jeśli tylko wyjściowe równanie jest stabilne.



Rys. 7.5.

3. Dla formuły trapezów mamy:

$y_1 = y_0 + \frac{T}{2}\lambda(y_0 + y_1) = \frac{2+T\lambda}{2} y_0 + \frac{T\lambda}{2} y_1$, skąd: $y_1 = \frac{2+T\lambda}{2-T\lambda} y_0$.

Ogólnie:

$$y_n = \left(\frac{2+T\lambda}{2-T\lambda}\right)^n y_0 \tag{7.21}$$

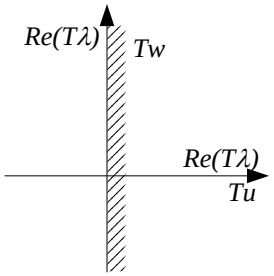
Ciąg ten jest stabilny dla:

$$\left|\frac{2+T\lambda}{2-T\lambda}\right| \le 1, \tag{7.22}$$

co jest równoważne relacji:

$\left|\frac{2+Tu+jTw}{2-Tu+jTw}\right| \le 1$ oraz: $\frac{(2+Tu)^2+(Tw)^2}{(2-Tu)^2+(Tw)^2} \le 1$

Powyższy związek jest słuszny dla: $u \le 0$, co oznacza, że procedura jest stabilna dla całej ujemnej części płaszczyzny zespolonej współczynnika λ (rys. 7.6) - a więc procedura jest również stabilna globalnie, jeśli tylko wyjściowe równanie ciągłe jest stabilne.



Rys. 7.6.

7.3. Metody wielokrokowe

Metody Geara

Spośród metod wielokrokowych, szczególnie ważne są metody całkowania, projektowane z myślą o zastosowaniu w odniesieniu do tzw. sztywnych systemów. Sztywność układu równań różniczkowych oznacza szybkie zanikanie (dążenie do zera) rozwiązania. Stosowane w takim przypadku metody nie mogą być wysokiego rzędu, które należą do grupy metod niejawnych. Są to przede wszystkim metody Geara i niejawne metody Rungego–Kutty (R–K)[1], które odznaczają się dużą stabilnością [5].

Metody Geara wywodzą się bezpośrednio z numerycznej reprezentacji pochodnej w podstawowym równaniu (7.1), Na przykład, przy najprostszym zapisie pochodnej:

$$y' = \frac{dy(t)}{dt} \approx \frac{y_{n+1} - y_n}{T}$$

oraz przy założeniu, że funkcja prawej strony równania będzie liczona według zasady ekstrapolacji: $f(y_{n+1}, t_{n+1})$, to otrzymamy:

$$y_{n+1} = y_n + Tf(y_{n+1}, t_{n+1}), \tag{7.23}$$

co pokrywa się z niejawną metodą prostokątów.

W celu podwyższenia rzędu metody, można zastosować dokładniejszą zależność na obliczanie pochodnej. W tym celu można zastosować wielomian interpolacyjny Newtona, w którym pochodna na końcu ostatniego przedziału jest obliczana na podstawie trzech punktów, w których znana jest różniczkowana funkcja. Prowadzi to do następującej zależności (p. 4.3):

$$y' = \frac{dy(t)}{dt} \approx \frac{3y_{n+1} - 4y_n + y_{n-1}}{2T}$$

[1] Należy zauważyć, że powszechnie są stosowane jawne metody R–K (p. 7.2), które nie mają wymaganych tu właściwości w odniesieniu do układów sztywnych, np. metoda R–K IV rzędu [5], [14].

Podstawienie tej zależności do (7.1) prowadzi do następującej formuły Geara 2-go rzędu:

$$y_{n+1} = \frac{4}{3} y_n - \frac{1}{3} y_{n-1} + \frac{2}{3} T f(y_{n+1}, t_{n+1}) \tag{7.24}$$

Metody wyższych rzędów nie są stosowane, gdyż występują również problemy z ich stabilnością w przypadku sztywnych systemów równań.

Niejawna metoda Rungego-Kutty

Prezentowane w p. 7.2 jednokrokowe algorytmy Rungego-Kutty są metodami jawnymi i w związku z tym zazwyczaj nie są stosowane do modelowania układów dynamicznych. Odrębną grupę stanowią wielokrokowe niejawne metody Rungego-Kutty [2]. Spośród tych ostatnich zachęcający jest, zwłaszcza w algorytmach modelowania numerycznego, 2-stopniowy algorytm R–K II rzędu. W literaturze anglojęzycznej jest on oznaczany akronimem: 2S-DIRK (ang. *2-stage diagonally implicite Runge–Kutta*). Numeryczne przybliżenie rozwiązania równania (7.1) w *k*-tym kroku jest określane za pomocą następującego 2-stopniowego algorytmu [5]:

1. $$\tilde{y}(k) = y(k-1) + \tilde{T} f\left(\tilde{t}_k, \tilde{y}(k)\right) \tag{7.25}$$

aproksymacja: $$\tilde{y}(k-1) = \alpha y(k-1) + \beta \tilde{y}(k) \tag{7.26}$$

2. $$y(k) = \tilde{y}(k-1) + \tilde{T} f\left(t_k, y(k)\right) \tag{7.27}$$

gdzie zmienne oznaczone tyldą są wielkościami pomocniczymi, odnoszącymi się do punktu pośredniego w czasie, pomiędzy t_{k-1} i t_k:

$$\tilde{t}_k = t_{k-1} + \tilde{T}, \quad \tilde{T} = \left(1 + \frac{1}{\alpha}\right) T \tag{7.28}$$

$\alpha = -\sqrt{2}$, $\alpha + \beta = 1$.

Można zauważyć, że w obu stopniach algorytmu jest realizowana niejawna metoda Eulera z początkowymi punktami w $y(k-1)$ i $\tilde{y}(k-1)$, odpowiednio, z krokiem $\tilde{T} = (1 - \sqrt{2}/2)T$. Łatwo więc wyznaczyć odpowiednie zależności odnoszące się do konkretnego zastosowania metody.

7.4. Metody ekstrapolacyjno-interpolacyjne

Powyższe rozwiązania pokazują, że metody niejawne rozwiązywania równań różniczkowych mają szersze zastosowanie, gdyż wykazują większą stabilność. Jednak formuła niejawna nie zawsze może być stosowana, gdyż w ogólnym przypadku, wymaga rozwiązania równania uwikłanego (niewiadoma występuje po obu stronach równania), co niekiedy może być kłopotliwe. W takim przypadku można stosować dwie różne formuły w każdym kroku rozwiązania równania, według następującego schematu:

1. Obliczyć przybliżone rozwiązanie $\tilde{y}_{n+1}$ według procedury jawnej.

2. Poprawić rozwiązanie y_{n+1} według procedury niejawnej, z wykorzystaniem rozwiązania przybliżonego.

Pierwszy krok jest prognozą (ekstrapolacją) rozwiązania, a drugi: korekcją (interpolacją), skąd schemat ten jest nazywany metodą prognozy i korekcji.

W pierwszym kroku stosuje się metodę niejawną o jeden rząd mniej dokładną, niż w drugim kroku z metodą niejawną. Na przykład, niejawna metoda prostokątów (I rząd) może być połączona z metodą trapezów (niejawna metoda II rzędu):

$$\begin{aligned} \tilde{y}_{n+1} &= y_n + Tf\left(y_n, t_n\right) \\ y_{n+1} &= y_n + \frac{T}{2} f\left(y_n, t_n\right) + f\left(\tilde{y}_{n+1}, t_{n+1}\right) \end{aligned} \tag{7.29}$$

Zazwyczaj są w takim przypadku stosowane metody wyższych rzędów.

8. Literatura

[1] AL-KHAFAJI A.W., TOOLEY J.R., *Numerical methods in engineering practice*, Holt, Rinehart and Winston, Inc., New York,1986.

[2] АРУШАНЯН О.Б., ЗАЛЁТКИН С.Ф., *Численные методы решения обыкновенных дифференциальных уравнений (задача Коши)*. Dostępne w: http://www.srcc.msu.su/num_anal/list_wrk/sb3_doc/part6.htm

[3] DRYJA M., JANKOWSCY J. i M., *Przegląd metod i algorytmów numerycznych*. Część 2, WNT, Warszawa, 1982.

[4] FORSYTHE G.E., MALCOLM M.A., MOLER C.B., *Computer methods for mathematical computations*, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1977.

[5] FORTUNA Z., MACUKOW B., WĄSOWSKI J., *Metody numeryczne*, WNT, Warszawa, 2003.

[6] JANKOWSCY J. i M., *Przegląd metod i algorytmów numerycznych*. Część 1, WNT, Warszawa, 1981.

[7] KACZOREK T., *Wektory i macierze w automatyce i elektrotechnice*, WNT, Warszawa, 1998.

[8] KIEŁBASIŃSKI A., SCHWETLICK H., *Numeryczna algebra liniowa*, WNT, Warszawa, 1992.

[9] KINCAID D., CHENEY W., *Analiza numeryczna*. WNT, Warszawa, 2005.

[10] KRUPKA J., MORAWSKI R.Z., OPALSKI L.J., *Wstęp do metod numerycznych dla studentów elektroniki i technik informacyjnych*. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 1999.

[11] PRESS W.H., TEUKOLSKY S.A., VETTERLING W.T., FLANNERY B.P., *Numerical recipes in C. The art of scientific computing*. Cambridge University Press, Cambridge 1992 (oddzielne fragmenty książki dostępne są na stronie internetowej: http://apps.nrbook.com/empanel/index.html#).

[12] ROSOŁOWSKI E., *Cyfrowe przetwarzanie sygnałów w automatyce elektroenergetycznej*. Akademicka Oficyna Wydawnicza EXIT, Warszawa 2002.

[13] ROSOŁOWSKI E., *Komputerowe metody analizy elektromagnetycznych stanów przejściowych*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2009.

[14] STOER J., BULRISCH R., *Wstęp do analizy numerycznej*, PWN, Warszawa, 1987.

[15] WANAT K., *Wybór metod numerycznych*, Wydawnictwo DIR, Gliwice, 1993.

[16] WEISSTEIN E.W., *Singular Value Decomposition*. From *MathWorld*--A Wolfram Web Resource: http://mathworld.wolfram.com/SingularValueDecomposition.html

Podstawowe idee związane z aproksymacją średniokwadratową zostały sformułowane niezależnie przez dwóch matematyków:

Johann Carl Friedrich Gauss (1777-1855) – Niemiec,

Adrien-Marie Legendre (1752-1833) – Francuz.

Inne ciekawe strony:

http://www.csit.fsu.edu/~burkardt/
http://public.lanl.gov/mewall/kluwer2002.html
http://www.american.edu/cas/mathstat/People/kalman/pdffiles/svd.pdf
http://www.cs.toronto.edu/NA/index.html - Department of Computer Science, University of Toronto.

Skorowidz