POGLAVJE II

II.4 Reševanje diferencialne enačbe prečnega pomika

1. Reševanje diferencialnih enačb s programom Mathematica
2. Analitično iskanje lastnih frekvenc za prostoležeči nosilec
3. Analitično iskanje lastnih nihajnih oblik za prostoležeči nosilec

1. Reševanje diferencialnih enačb s programom *Mathematica*

Druga diferencialna enačba, ki sledi iz parcialne diferencialne enačbe iz poglavja II.1, je navadna diferencialna enačba pomika, ki je funkcija lokalne koordinate x, in se rešuje z že predstavljenim ukazom *DSolve[diferencialna enačba, funkcija, neodvisni parameter]*. Čeprav *Mathematica* poskusi poiskati rešitev v analitični obliki in hkrati upoštevati vse robne pogoje ter se tako izogniti uporabi integracijskih konstant (C[1], C[2],....), se pri reševanju diferencialne enačbe pomika običajno najprej poišče splošna rešitev diferencialne enačbe (torej z vsemi konstantami vred), ustrezne pripadajoče rešitve (upoštevajo se ustrezni robni pogoji) pa šele nato poiščejo za obravnavano konstrukcijo.
Diferencialna enačba pomika se zapiše in reši z:

■ **Funkcija pomika**

```
In[1]:= DSolve[w''''[x] - b^4 w[x] == 0, w[x], x]

Out[1]= {{w[x] → e^(-b x) C[2] + e^(b x) C[4] + C[1] Cos[b x] + C[3] Sin[b x]}}
```

Ker v izrazu nastopata dve eksponentna člena, ki sta sicer korektna, vendar bi ju radi nadomestili z inženirsko prijaznejšima trigonometričnima funkcijama, se rezultat najprej shrani v neko spremenljivko, npr. *funkcija*, saj so ukazi za manipuliranje mnogo uspešnejši, če se izvedejo direktno na funkciji (torej na inženirsko uporabnem konkretnem izrazu med → in }.

```
In[2]:= funkcija = w[x] /. %[[1]]

Out[2]= e^(-b x) C[2] + e^(b x) C[4] + C[1] Cos[b x] + C[3] Sin[b x]
```

Šele sedaj se zahteva nadomestitev eksponentnih funkcij z trigonometričnimi:

```
In[3]:= ExpToTrig[%]

Out[3]= C[1] Cos[b x] + C[2] Cosh[b x] + C[4] Cosh[b x] +
         C[3] Sin[b x] - C[2] Sinh[b x] + C[4] Sinh[b x]
```

ki izraz prevede v bolj znano obliko, ki pa se očitno še lahko poenostavi:

```
In[4]:= Simplify[%]

Out[4]= C[1] Cos[b x] + (C[2] + C[4]) Cosh[b x] +
         C[3] Sin[b x] + (-C[2] + C[4]) Sinh[b x]
```

Iz izraza se jasno vidi, da sta C[2] + C[4] ter –C[2] + C[4] neki novi konstanti. Zaradi poenostavitve zapisa konstant (njegove prednosti bodo kmalu vidne) in skrajšanja zapisa ju preimenujemo kar v:

```
In[5]:= Simplify[%] /. {C[2] + C[4] → C[2], -C[2] + C[4] → C[4]}

Out[5]= C[1] Cos[b x] + C[2] Cosh[b x] +
         C[3] Sin[b x] + C[4] Sinh[b x]
```

Enačba končno dobi obliko, znano iz učbenikov dinamike, in s katero bi lahko analizo pravzaprav kar začeli. Na tem mestu je potrebno opozoriti, da so različne verzije *Mathematice* tudi različno uspešne pri izpeljavi gornje enačbe. Prikazani postopek velja za verzijo 5, pri uporabi verzije 4 pa je potrebnih še nekaj dodatnih korakov (pri izpeljavi konstant C[1] + C[3]), da se izpelje enaki rezultat.
Tako se sedaj končno mogoče definirati splošno funkcijo prečnega pomika (zaenkrat še z neznanimi konstantami):

```
In[6]:= w[x_] = %;
```

Preveriti je seveda mogoče tudi, ali tako zapisana funkcija ustreza začetni diferencialni enačbi (če bi se diferencialna enačba shranila v neko spremenljivko, kot je to bilo storjeno v poglavju II.1, diferencialne enačbe sedaj ne bi bilo potrebno ponovno zapisati):

```
In[7]:= Simplify[w''''[x] - b^4 w[x]]

Out[7]= 0
```

Enačbo je mogoče uporabljati za različno podprte linijske konstrukcijske elemente, vendar pa je potrebno za vsakega izmed njih poiskati vrednosti neznanih konstant glede na dejanske pogoje vpetja (robne pogoje) in zato se vpelje vektor neznanih konstant. Zaradi primernega imenovanja konstant se to lahko naredi izredno elegantno kar v zanki, ki kreira vektor z ukazom *Table[definicija člena,{indeks zanke, začetna meja, končna meja, korak}]*. Če sta začetna meja ter korak enaka 1, se lahko izpustita:

```
In[8]:= neznanke = Table[C[i], {i, 4}]

Out[8]= {C[1], C[2], C[3], C[4]}
```

2. Analitično iskanje lastnih frekvenc za prostoležeči nosilec

Za prostoležeči nosilec dolžine L veljajo naslednji robni pogoji, ki se lahko zapišejo v vektor:

```
■ Pogoji za prostolezeci nosilec

In[9]:= pogoji = {w[0] == 0, w''[0] == 0, w[L] == 0, w''[L] == 0};
```

ki predstavljajo štiri linearne homogene enačbe s štirimi (petimi) neznankami:

```
In[10]:= pogoji

Out[10]= {C[1] + C[2] == 0, -b^2 C[1] + b^2 C[2] == 0,
  C[1] Cos[b L] + C[2] Cosh[b L] + C[3] Sin[b L] +
   C[4] Sinh[b L] == 0, -b^2 C[1] Cos[b L] + b^2 C[2] Cosh[b L] -
   b^2 C[3] Sin[b L] + b^2 C[4] Sinh[b L] == 0}
```

Klasično reševanje sistema linearnih homogenih enačb z ukazom *Solve[{sistem enačb, neznanke}]* pripelje seveda zgolj do neuporabne trivialne oz. ničelne rešitve:

```
In[11]:= resitev = Solve[pogoji, neznanke]

Out[11]= {{C[3] → 0, C[4] → 0, C[1] → 0, C[2] → 0}}
```

Ker pa je znano, da ima homogeni sistem tudi netrivialno rešitev, kadar je determinanta sistema enaka nič, je potrebno za izračun determinante enačbe zapisati v matrični obliki, kar se izvede z ukazom *Coefficient[enačba,parameter]*, ki v spremenljivki *enačba* poišče vrednost, ki pripada spremenljivki *parameter*. Ker gre za sistem enačb, je te koeficiente potrebno določiti za vsako enačbo in vsak parameter posebej, kar se najelegantneje naredi v dvojni zanki. Rezultat pa seveda ni ena sama vrednost, ampak matrika, shranjena v spremenljivi *matrika*, in zato je potrebno še uporabit ukaz *Table[]*, vendar tokrat z dvema indeksoma, saj gre za matriko:

```
In[12]:= matrika =
   Table[Coefficient[pogoji[[i, 1]], neznanke[[j]]],
    {i, 4}, {j, 4}];
```

Rezultat je tako shranjen v matriki, ki se lahko pregledno zapiše z ukazom *MatrixForm[matrika]*:

```
In[13]:= MatrixForm[matrika]

Out[13]//MatrixForm=
```

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ -b^2 & b^2 & 0 & 0 \\ \mathrm{Cos}[b\,L] & \mathrm{Cosh}[b\,L] & \mathrm{Sin}[b\,L] & \mathrm{Sinh}[b\,L] \\ -b^2\,\mathrm{Cos}[b\,L] & b^2\,\mathrm{Cosh}[b\,L] & -b^2\,\mathrm{Sin}[b\,L] & b^2\,\mathrm{Sinh}[b\,L] \end{pmatrix}$$

Determinanta se izračuna z ukazom *Det[matrika]*:

■ Lastna frekvenca

```
In[14]:= determinanta = Det[matrika]

Out[14]= 4 b^4 Sin[b L] Sinh[b L]
```

Rešitve te enačbe so vrednosti koeficienta b (pete neznanke), ki je povezan z lastno frekvenco konstrukcije.

Produkt treh števil je enak nič samo, če je vsaj eden izmed treh členov enak nič. Če velja najpreprostejša rešitev, torej b=0, sledi pomik kot:

```
In[15]:= w[x] /. b → 0

Out[15]= C[1] + C[2]
```

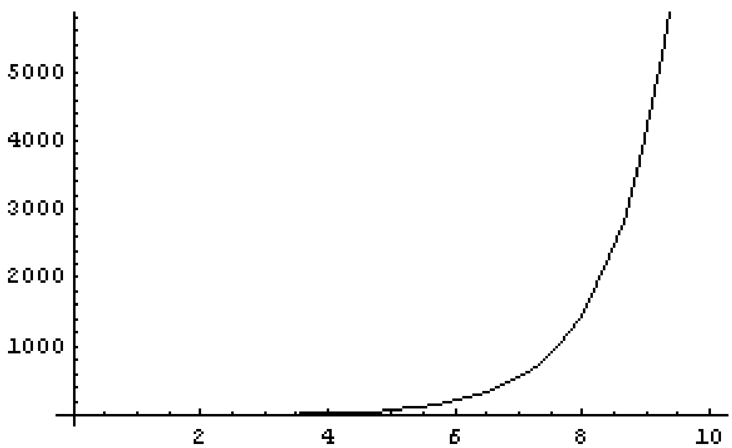
kar predstavlja konstantni pomik (ki fizikalno ni mogoč, hkrati pa ne omogoča vpeljave vseh dejanskih pogojev vpetja - robnih pogojev). Zato je zanimiva rešitev $b \neq 0$, in se lahko izraz ustrezno krajša:

```
In[16]:= determinanta = determinanta / 4 / b^4

Out[16]= Sin[b L] Sinh[b L]
```

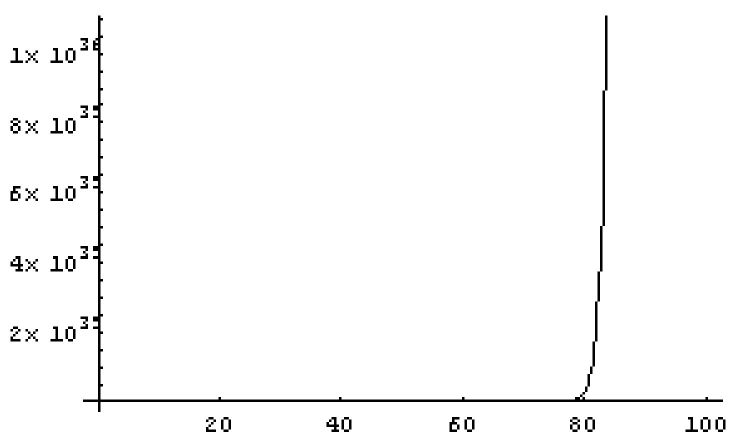
Za rešitev problema mora torej biti vsaj eden izmed izrazov enak nič. Vendar je *sinh* funkcija, ki narašča preko vsake meje, v kar se je mogoče prepričati tudi z enostavnim izrisom funkcije:

```
In[17]:= Plot[Sinh[x], {x, 0, 10}];
```



v različnih intervalih

```
In[18]:= Plot[Sinh[x], {x, 0, 100}];
```



Edina vrednost, ko je funkcija enaka nič, spet vodi do b=0, in tako postane jasno, da bodo ničle vezane zgolj na člen *sin(b L)*, kar omogoči dodatno krajšanje:

```
In[19]:= determinanta = determinanta / Sinh[b L]

Out[19]= Sin[b L]
```

Kljub vsem možnostim, ki jih nudi, *Mathematica* ni sposobna inženirsko zadovoljivo rešiti zgornje enačbe oz. poiskati ustrezne ničle, saj ukaz *Solve[enačba,neznanka]* ne vrne splošne rešitve, ampak zgolj prvo, inženirsko neuporabno, resda z izpisom opozorila, da morda niso najdene vse rešitve:

```
In[20]:= Solve[% == 0, b]

         Solve::ifun :
          Inverse functions are being used by Solve, so some
            solutions may not be found; use Reduce
            for complete solution information. More…

Out[20]= {{b → 0}}
```

Tudi uporaba opcije *InverseFunctions→True* ne prinese napredka:

```
In[21]:= Solve[determinanta == 0, b, InverseFunctions → True]

Out[21]= {{b → 0}}
```

in zato je potrebno neznanko b poiskati z upoštevanjem, da je *sin(x)* periodična funkcija, ki ima ničlo ob vsakem mnogokratniku števila π (ki se lahko zapiše tudi kot Pi):

```
In[22]:= Solve[b L == n Pi, b]
```

Out[22]= $\left\{\left\{b \to \frac{n \pi}{L}\right\}\right\}$

Rešitev se shrani npr. v spremenljivko B:

```
In[23]:= B = b /. %[[1]]
```

Out[23]= $\frac{n \pi}{L}$

Jasno je, da je *n* celoštevilska spremenljivka, večja od 0 (saj *n*=0 ponovno vodi do b=0, kar je bilo analizirano že zgoraj). Rešitev je mogoče preveriti tako, da se vstavi v determinanto (ob izbrani vrednosti *n*):

```
In[24]:= determinanta /. {b → B, n → 1}

Out[24]= Sin[n π]
```

ki pa pokaže slabost *Mathematice*, da očitno ni sposobna opraviti zaporedne zamenjave v že zamenjanem členu. Zato se izraz zapiše kot:

```
In[25]:= determinanta /. b → B /. n → 1

Out[25]= 0
```

Iz rešitve za konstanto *b* je sedaj mogoče poiskati lastno frekvenco. Ker pa je *b* funkcija števila *n*, je jasno, da vsaki vrednosti *n* pripada drugačna lastna frekvenca in tako sledijo lastne frekvence ω_n iz rešitve:

In[26]:= $\text{Solve}\left[B^4 == \frac{\omega^2 m}{EI}, \omega\right]$

Out[26]= $\left\{\left\{\omega \to -\frac{\sqrt{EI}\, n^2 \pi^2}{L^2 \sqrt{m}}\right\}, \left\{\omega \to \frac{\sqrt{EI}\, n^2 \pi^2}{L^2 \sqrt{m}}\right\}\right\}$

od katerih je zanimiva zgolj pozitivna vrednost:

```
In[27]:= ω = ω /. %[[2]]
```

Out[27]= $\frac{\sqrt{EI}\, n^2 \pi^2}{L^2 \sqrt{m}}$

3. Analitično iskanje lastnih nihajnih oblik za prostoležeči nosilec

Pri iskanju lastnih nihajnih oblik se je potrebno vrniti k linearnemu sistemu homogenih enačb. Ker je že bilo zagotovljeno, da je vrednost *b* takšna, da je determinanta enaka nič, je potrebno še določiti koeficiente C[1], C[2], C[3] in C[4]. Ker pa v bistvu nastopa neskončna množica vrednosti *b*, nastopa tudi neskončna množica rešitev za konstante C[1], C[2], C[3] in C[4] (za vsak *n* ena). Zaradi tega je smiselno enačbe, zapisane v vektorju *pogoji*, čim bolje analizirati na čim splošnejšem nivoju.

Tako je mogoče opaziti, da je mogoče drugo in četrto vrstico oz. drugo in četrto enačbo krajšati z b^2 (za katerega je sedaj že znano, da je lahko zgolj od nič različen). Prvi poskus ne vodi do pričakovanega rezultata:

■ Oblikovna funkcija

```
In[28]:= pogoji[[2]] / b^2
```

Out[28]= $\frac{-b^2 C[1] + b^2 C[2] == 0}{b^2}$

še slabše pa se obnese poskus z uporabo ukaza *Simplify[]*:

```
In[29]:= Simplify[pogoji[[2]] / b^2]
```

Out[29]= $\frac{b\, C[1] == b\, C[2]}{b^2}$

Zato se robna pogoja zapišeta ponovno, vendar kar s sprotno izvedbo krajšanja, najprej drugi:

```
In[30]:= pogoji[[2]] = Expand[w''[0] / b^2] == 0

Out[30]= -C[1] + C[2] == 0
```

in nato še četrti:

```
In[31]:= pogoji[[4]] = Expand[w''[L] / b^2] == 0

Out[31]= -C[1] Cos[b L] + C[2] Cosh[b L] -
           C[3] Sin[b L] + C[4] Sinh[b L] == 0
```

kar pogoje zapiše v preglednejši (a še vedno popolnoma splošni) obliki:

```
In[32]:= pogoji

Out[32]= {C[1] + C[2] == 0, -C[1] + C[2] == 0,
          C[1] Cos[b L] + C[2] Cosh[b L] + C[3] Sin[b L] +
            C[4] Sinh[b L] == 0, -C[1] Cos[b L] +
            C[2] Cosh[b L] - C[3] Sin[b L] + C[4] Sinh[b L] == 0}
```

Opaziti je mogoče, da v prvih dveh enačbah nastopata zgolj dve neznanki: C[1] in C[2], kar omogoči, da se ti dve enačbi rešita najprej. Ker gre za homogeni sistem, je potrebno preveriti ustrezno poddeterminanto sistema dveh enačb z dvema neznankama:

```
In[33]:= sistem1 =
          Table[Coefficient[pogoji[[i, 1]], neznanke[[j]]],
           {i, 2}, {j, 2}]

Out[33]= {{1, 1}, {-1, 1}}
```

Pripadajoča determinanta je tako:

```
In[34]:= Det[sistem1]

Out[34]= 2
```

Ker je očitno od nič različna, ima sistem zgolj trivialno rešitev C[1]=0 in C[2]=0, kar se poskusi upoštevati:

```
In[35]:= C[1] = 0

         Set::write : Tag C in C[1] is Protected. More…

Out[35]= 0
```

Ker pa so konstante C[ ] predefinirane in zaščitene konstante, operacije ni mogoče izvesti. Zato se v novi vektor zapišeta preostali enačbi (tretja in četrta), vendar z upoštevanjem izračunanih vrednosti obeh konstant:

```
In[36]:= drugienacbi = Table[pogoji[[i]] /. {C[1] → 0, C[2] → 0},
           {i, 3, 4}]

Out[36]= {C[3] Sin[b L] + C[4] Sinh[b L] == 0,
          -C[3] Sin[b L] + C[4] Sinh[b L] == 0}
```

Za ti dve enačbi se prav tako poiščejo koeficienti, s pomočjo katerih je mogoče izračunati determinanto:

```
In[37]:= sistem2 =
          Table[Coefficient[drugienacbi[[i, 1]], neznanke[[j]]],
           {i, 1, 2}, {j, 3, 4}]

Out[37]= {{Sin[b L], Sinh[b L]}, {-Sin[b L], Sinh[b L]}}
```

ki znaša:

```
In[38]:= Det[sistem2]

Out[38]= 2 Sin[b L] Sinh[b L]
```

Do sedaj je bila analiza iskanja lastnih nihajnih oblik izvedena za splošno vrednost spremenljivke *b*, za odločitev o poteku nadaljnje analize pa je potrebno preveriti, kam vodi vstavitev že znane vrednosti za konstanto *b*:

```
In[39]:= % /. b → B

Out[39]= 2 Sin[n π] Sinh[n π]
```

Ker je *n* celo število (večje od nič), je jasno, da je determinanta tega sistema enaka nič, in tako nastopi tudi netrivialna oz. neničelna rešitev.
Pregled obeh enačb ob upoštevanju vrednosti za *b*:

```
In[40]:= drugienacbi /. b → B

Out[40]= {C[3] Sin[n π] + C[4] Sinh[n π] == 0,
          -C[3] Sin[n π] + C[4] Sinh[n π] == 0}
```

pokaže, da sta koeficienta ob konstanti C[3] vedno enaka nič, kar velja za poljubno celoštevilčno vrednost *n*, tudi npr. za n=1:

```
In[41]:= % /. n → 1

Out[41]= {C[4] Sinh[π] == 0, C[4] Sinh[π] == 0}
```

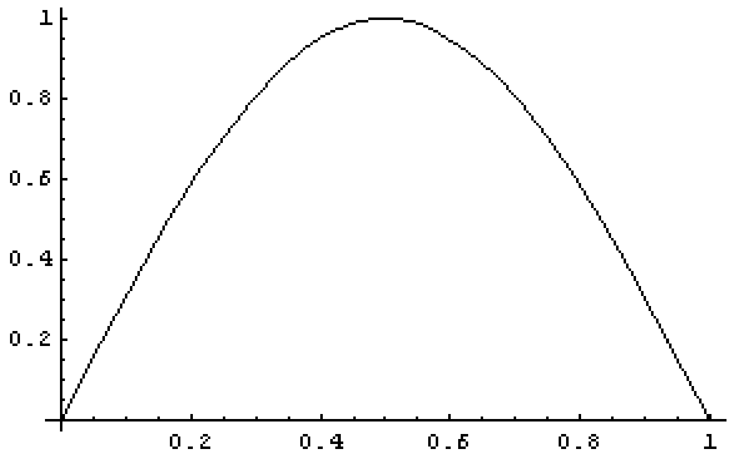
ki jasno pokaže rešitev C[4]=0. Tako se definira lastna nihajna oblika kot:

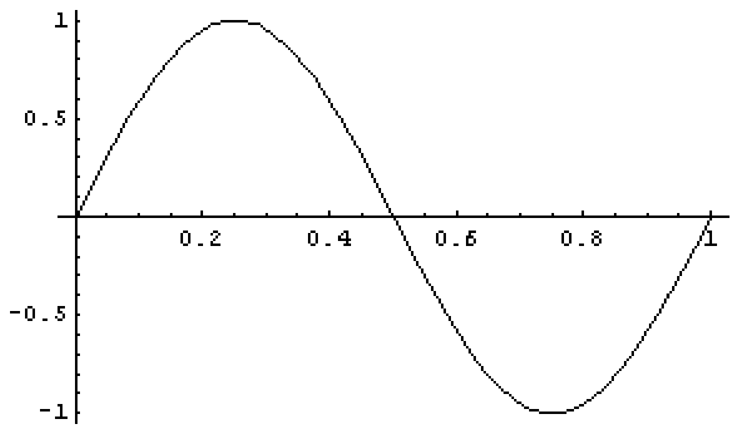
```
In[42]:= w[x_] = w[x] /. {C[1] → 0, C[2] → 0, C[4] → 0, b → B}
```

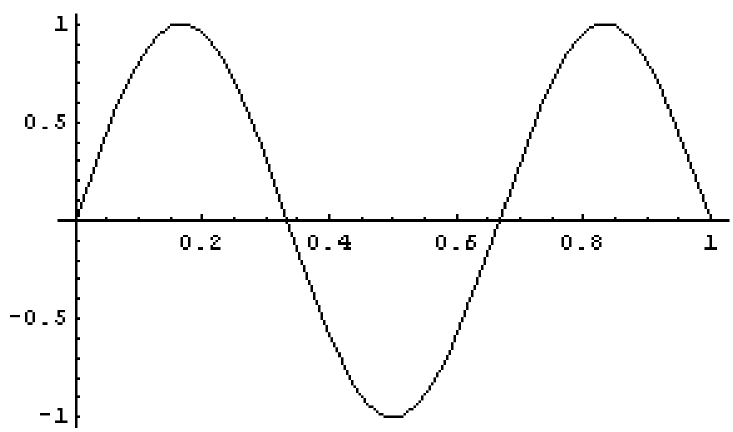
Out[42]= $C[3]\ \mathrm{Sin}\left[\frac{n\,\pi\,x}{L}\right]$

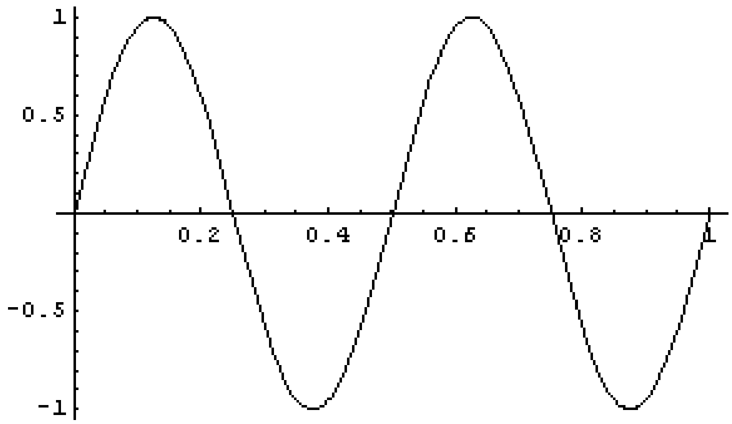
Ker gre za nihajno obliko, velikost konstante C[3] ni pomembna (vse dokler je od nič različna). Tako je mogoče zlahka izrisati nekaj prvih lastnih nihajnih oblik. Te nihajne oblike se lahko ločeno izrišejo vsaka s svojim ukazom, lahko pa se tudi uporabi zanka Do[] (kot dolžina je bila za potrebe izrisa izbrana vrednost npr. 1):

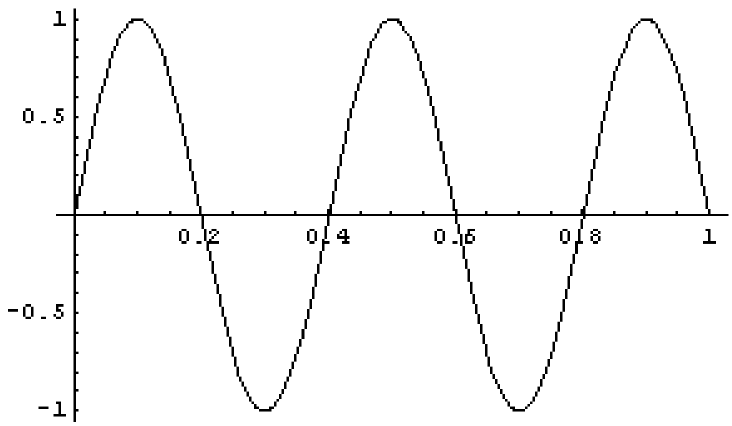
```
In[43]:= Do[Plot[w[x] /. {C[3] → 1, L → 1}, {x, 0, 1}],
           {n, 1, 6}]
```

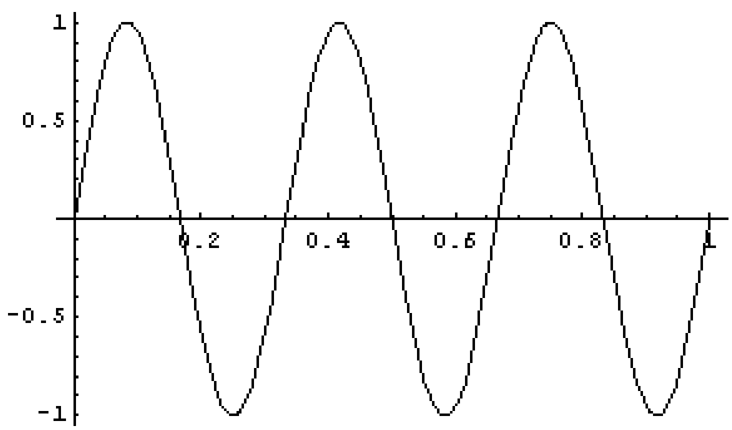












Za katero nihajno obliko gre, je mogoče ugotoviti kar neposredno iz slike, saj je število ničel funkcije med krajiščema elementa za 1 manjše od številke nihajne oblike (npr. druga nihajna oblika enkrat seka sliko »konstrukcije«).