POGLAVJE II

II. 10 Iskanje lastnih frekvenc in pripadajočih lastnih vektorjev

1. Združevanje lokalnih togostnih in masnih matrik v globalni matriki v zanki
2. Iskanje lastnih frekvenc in pripadajočih lastnih vektorjev

1. Združevanje lokalnih togostnih in masnih matrik v globalni matriki v zanki

Do sedaj obravnavani primeri so pokazali, da lastne frekvence sistemov s porazdeljeno maso šele s povečevanjem števila končnih elementov konvergirajo k točnim rešitvam. Zato se bosta v tem poglavju globalna togostna in masna matrika za enostavno konstrukcijo sestavili v zanki, kar bo omogočilo izdelavo modula, ki bo sposoben izvesti analizo enostavne konstrukcije z različnimi stopnjami diskretizacije v odvisnosti od enega samega parametra – števila izbranih končnih elementov konstrukcije (ter seveda geometrijskih in mehanskih lastnosti konstrukcije), kar je ekvivalent klasičnemu programiranju.

Uporabljan bo standardni končni element za prečno nihanje, ki je že bil izpeljan in uporabljan v podpoglavju II.8, in zato bosta togostna in masna matrika kar privzeti iz že obstoječe datoteke.
Zato se najprej v datoteki z izpeljavo označi npr. togostna matrika (celotna celica z vhodno in izhodno vrstico ali pa samo izhodna vrstica):

```
MatrixForm[K]
```

$$\begin{pmatrix} \frac{12\,EI}{L^3} & \frac{6\,EI}{L^2} & -\frac{12\,EI}{L^3} & \frac{6\,EI}{L^2} \\ \frac{6\,EI}{L^2} & \frac{4\,EI}{L} & -\frac{6\,EI}{L^2} & \frac{2\,EI}{L} \\ -\frac{12\,EI}{L^3} & -\frac{6\,EI}{L^2} & \frac{12\,EI}{L^3} & -\frac{6\,EI}{L^2} \\ \frac{6\,EI}{L^2} & \frac{2\,EI}{L} & -\frac{6\,EI}{L^2} & \frac{4\,EI}{L} \end{pmatrix}$$

ki se shrani v odložišče s *Ctrl+C*, nato pa se v novo datoteko prilepi s *Ctrl+V*. Celica ali pa zgolj izhodna vrstica se ponovno označi, nato pa s kombinacijo *Ctrl+Shift+I* pretvori v obliko vhodne vrstice:

```
MatrixForm[K]

MatrixForm[{{(12*EI)/L^3, (6*EI)/L^2, -((12*EI)/L^3),
    (6*EI)/L^2}, {(6*EI)/L^2, (4*EI)/L, -((6*EI)/L^2),
    (2*EI)/L}, {-((12*EI)/L^3), -((6*EI)/L^2),
    (12*EI)/L^3, -((6*EI)/L^2)}, {(6*EI)/L^2, (2*EI)/L,
    -((6*EI)/L^2), (4*EI)/L}}]
```

ki pa še ni pripravljena za izvedbo. Kurzor se sedaj postavi na začetek vrstice, zapiše se ime matrike (kar povzroči tvorbo nove vrstice), izbriše se besedilo *MatrixForm[*, oglati zaklepaj *]* pa se nadomesti s podpičjem ter na koncu izvede vrstica:

```
MatrixForm[K]
```

```
MatrixForm[{{(12*EI)/L^3, (6*EI)/L^2, -((12*EI)/L^3),
(6*EI)/L^2},
   {(6*EI)/L^2, (4*EI)/L, -((6*EI)/L^2), (2*EI)/L},
   {-((12*EI)/L^3), -((6*EI)/L^2), (12*EI)/L^3,
-((6*EI)/L^2)},
   {(6*EI)/L^2, (2*EI)/L, -((6*EI)/L^2), (4*EI)/L}}]
```

```
In[1]:= K={{(12*EI)/L^3, (6*EI)/L^2, -((12*EI)/L^3), (6*EI)/L^2},
      {(6*EI)/L^2, (4*EI)/L, -((6*EI)/L^2), (2*EI)/L},
      {-((12*EI)/L^3), -((6*EI)/L^2), (12*EI)/L^3,
   -((6*EI)/L^2)},
      {(6*EI)/L^2, (2*EI)/L, -((6*EI)/L^2), (4*EI)/L}};
```

Enak postopek se izvede še za masno matriko, kar vodi do:

```
MatrixForm[M]
```

```
MatrixForm[{{(13*L*m)/35, (11*L^2*m)/210, (9*L*m)/70,
    -((13*L^2*m)/420)}, {(11*L^2*m)/210, (L^3*m)/105,
    (13*L^2*m)/420, -((L^3*m)/140)},
   {(9*L*m)/70, (13*L^2*m)/420, (13*L*m)/35,
    -((11*L^2*m)/210)}, {-((13*L^2*m)/420), -((L^3*m)/140),
    -((11*L^2*m)/210), (L^3*m)/105}}]
```

```
In[2]:= M={{(13*L*m)/35, (11*L^2*m)/210, (9*L*m)/70,
      -((13*L^2*m)/420)}, {(11*L^2*m)/210, (L^3*m)/105,
      (13*L^2*m)/420, -((L^3*m)/140)},
     {(9*L*m)/70, (13*L^2*m)/420, (13*L*m)/35,
      -((11*L^2*m)/210)}, {-((13*L^2*m)/420), -((L^3*m)/140),
      -((11*L^2*m)/210), (L^3*m)/105}};
```

Sedaj sta matriki pripravljeni za nadaljnjo uporabo in (neuporabna in s tem nepotrebna) ostanka kopiranih oz. prilepljenih vrstic iz stare datoteke je sedaj brez škode mogoče izbrisati.
Za namene zgleda se bo uporabila že dobro znana konzola iz vaje 10 (http://www.geocities.com/mcsDISK/e10.pdf).

Vpelje se spremenljivka, ki bo definirala število uporabljenih končnih elementov, npr. *Ne* (enaka 6 v obravnavanem zgledu):

Konzola iz Vaje 10

■ **diskretizacija s poljubno izbranim stevilom koncnih elementov**

```
In[3]:= Ne = 6;
```

Ranga (število prostostnih stopenj) obeh globalnih matrik (masne in togostne) konstrukcije sta sedaj odvisna od tega števila:

```
In[4]:= Kk = Table[0, {i, 2 Ne}, {j, 2 Ne}];
        Mk = Table[0, {i, 2 Ne}, {j, 2 Ne}];
```

Če se izbere diskretizacija s končnimi elementi enakih dolžin (kar je tudi najbolj praktično), se uporabita enaki togostna in masna matrika za vse elemente (enotna dolžina posameznega končnega elementa pa se seveda mora izračunati glede na izbrano število končnih elementov za diskretizacijo):

```
In[6]:= Ke = K /. L → L / Ne;
        Me = M /. L → L / Ne;
```

Če je konzola npr. vpeta na začetku, se v matriko konstrukcije prenese samo del obeh matrik prvega končnega elementa (členi, ki pripadajo prostostnim stopnjam končnega vozlišča elementa, torej 3. in 4. vrstica ter stolpec):

```
In[8]:= Do[Kk[[i, j]] = Ke[[i + 2, j + 2]], {j, 1, 2}, {i, 1, 2}]
        Do[Mk[[i, j]] = Me[[i + 2, j + 2]], {j, 1, 2}, {i, 1, 2}]
```

Vsi ostali končni elementi (od drugega do zadnjega) pa prispevajo vse člene v matriki konstrukcije (vse stolpce in vrstice):

```
In[10]:= Do[Do[Kk[[2 (k - 2) + i, 2 (k - 2) + j]] =
            Kk[[2 (k - 2) + i, 2 (k - 2) + j]] + Ke[[i, j]], {j, 1, 4},
          {i, 1, 4}], {k, 2, Ne}];
         Do[Do[Mk[[2 (k - 2) + i, 2 (k - 2) + j]] =
            Mk[[2 (k - 2) + i, 2 (k - 2) + j]] + Me[[i, j]], {j, 1, 4},
          {i, 1, 4}], {k, 2, Ne}];
```

2. Iskanje lastnih frekvenc in pripadajočih lastnih vektorjev

Lastne frekvence se izračunajo s pomočjo ničel karakterističnega polinoma:

```
In[12]:= Det[Kk - ω^2 Mk];
```

ki se v delno urejeni obliki izpiše kot:

```
In[13]:= ExpandAll[%]

Out[13]= 1414873086212693490558566 4 EI^12 / L^24 -
  4126167306590181363862732 8 EI^11 m ω^2 / (35 L^20) +
  6849071953511691174543360 EI^10 m^2 ω^4 / (245 L^16) -
  7889888295396295704576 EI^9 m^3 ω^6 / (8575 L^12) +
  23725767340696179456 EI^8 m^4 ω^8 / (300125 L^8) -
  1249644962806454 88 EI^7 m^5 ω^10 / (52521875 L^4) +
  2589864007562089 EI^6 m^6 ω^12 / 88236750000 -
  3331901194399583 EI^5 L^4 m^7 ω^14 / 205838690400000 00 +
  1091798704149491 9 EI^4 L^8 m^8 ω^16 / 256096265048064 00000000 -
  5495002690435 03 EI^3 L^12 m^9 ω^18 / 995702278506872832000000000 +
  1383313425673 87 EI^2 L^16 m^10 ω^20 / 398189875765857075855360000000000 -
  2140430946451 1 EI L^20 m^11 ω^22 / 216742713176871323529589555200000000000 +
  1191982074220 9 L^24 m^12 ω^24 / 121348176311761301647158363448934400000000000 0
```

Seveda se faza izpisa, še posebej pri večjem številu končnih elementov, brez škode izpusti, saj ne nudi nobene potrebne informacije.

Mnogo zanimivejši in potrebnejši je namreč izračun in izpis lastnih frekvenc:

```
In[14]:= omege = N[Solve[% == 0, ω]]
```

$$\text{Out[14]}= \left\{\left\{\omega \to -\frac{3.51604\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{3.51604\sqrt{EI}}{L^2\sqrt{m}}\right\},\right.$$
$$\left\{\omega \to -\frac{22.0399\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{22.0399\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{61.8101\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{61.8101\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{121.681\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{121.681\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{202.863\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{202.863\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{303.532\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{303.532\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{468.023\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{468.023\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{642.849\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{642.849\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{878.454\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{878.454\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{1188.23\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{1188.23\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left\{\omega \to -\frac{1562.73\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{1562.73\sqrt{EI}}{L^2\sqrt{m}}\right\},$$
$$\left.\left\{\omega \to -\frac{2154.8\sqrt{EI}}{L^2\sqrt{m}}\right\}, \left\{\omega \to \frac{2154.8\sqrt{EI}}{L^2\sqrt{m}}\right\}\right\}$$

Čeprav lastna frekvenca ne more biti negativna, *Mathematica* rešitve obravnava kot ničle polinoma in zato izpiše tudi (strogo matematične) negativne rešitve.

Primerjava dobljenih vrednosti z numeričnimi vrednostmi, izračunanimi v podpoglavju II.5, pokaže, da divergenca rezultatov narašča z višjimi lastnimi frekvencami. Prvih 6 lastnih frekvenc ima napako manjšo od 2 %, pri sedmi frekvenci pa napaka naraste že na 12 %.

O uspešnosti posamezne izračunane lastne frekvence pa je mogoče soditi tudi s pomočjo pripadajočih lastnih vektorjev, ki predstavljajo nihajne oblike.

Ker pa je metoda končnih elementov v bistvu numerična metoda, se izberejo neke dimenzije in mehanske lastnosti konstrukcije:

```
In[15]:= L = 10;
         b = 0.4;
         h = 0.6;
         EI = 30 10^9 b h^3 / 12;
         m = b h 2400;
```

kar omoči najprej numerično izvrednotenje lastnih frekvenc:

```
In[20]:= omege

Out[20]= {{ω → -21.5312}, {ω → 21.5312}, {ω → -134.966}, {ω → 134.966},
          {ω → -378.508}, {ω → 378.508}, {ω → -745.141}, {ω → 745.141},
          {ω → -1242.28}, {ω → 1242.28}, {ω → -1858.74}, {ω → 1858.74},
          {ω → -2866.04}, {ω → 2866.04}, {ω → -3936.63}, {ω → 3936.63},
          {ω → -5379.41}, {ω → 5379.41}, {ω → -7276.4}, {ω → 7276.4},
          {ω → -9569.76}, {ω → 9569.76}, {ω → -13195.4}, {ω → 13195.4}}
```

nato pa še izračun lastnih vektorjev. Najprej se zato definira dinamična matrika DM:

```
In[21]:= DM = Inverse[Kk].Mk;
```

za določitev lastnih vektorjev (in sicer že znanih lastnih frekvenc) pa se uporabi ukaz *Eigensystem[matrika]*, ki kot rezultat vrne lastne frekvence in njim pripadajoče lastne vektorje:

```
In[22]:= resitve = Eigensystem[DM];
```

Rezultat analize, ki je shranjen v spremenljivki *resitve*, ima dva dela. Prvi del predstavljajo lastne vrednosti matrike (zaradi ranga matrike 12*12 jih je zgolj 12):

```
In[23]:= lastnevrednosti = resitve[[1]]

Out[23]= {0.00215706, 0.000054897, 6.97991×10^-6, 1.80104×10^-6,
          6.47981×10^-7, 2.89441×10^-7, 1.21741×10^-7, 6.45283×10^-8,
          3.45566×10^-8, 1.88872×10^-8, 1.09194×10^-8, 5.74324×10^-9}
```

ki so v obravnavanem primeru obratnosorazmerne kvadratom lastnih krožnih frekvenc:

```
In[24]:= ω = Sqrt[1 / lastnevrednosti]

Out[24]= {21.5312, 134.966, 378.508, 745.141, 1242.28, 1858.74,
          2866.04, 3936.63, 5379.41, 7276.4, 9569.76, 13195.4}
```

Iz rezultata je sedaj razvidno, da je *Mathematica* upoštevala zgolj pozitivne vrednosti korenov, kar posledično vodi zgolj do pozitivnih lastnih frekvenc, ki jih je seveda toliko, kolikor je prostostnih stopenj sistema.

Drugi del rešitve pa so lastnim vrednostih oz. frekvencam pripadajoči lastni vektorji:

```
In[25]:= vektorji = resitve[[2]];
```

ki so shranjeni po vrsticah, kar pomeni, da prvo vrstico predstavlja lastni vektor, ki pripada prvi lastni frekvenci, itd..

Njihov izpis je seveda možen, vendar ne prinaša nobene bistvene informacije v tej fazi. Seveda pa je tudi brez izpisa možno preveriti, če so vektorji (ki seveda niso normirani na masno matriko, kot je običaj v dinamiki) res ortogonalni na masno in togostno matriko. Tako sledi za masno matriko (izpis ni prikazan v celoti):

```
In[26]:= vektorji.Mk.Transpose[vektorji]

Out[26]= {{680.055, 1.26121×10^-12, -3.05533×10^-13,
    -1.06581×10^-13, 2.70006×10^-13, 7.81597×10^-14,
    -9.23706×10^-14, -5.68434×10^-14, -7.10543×10^-14,
    -2.13163×10^-13, -2.84217×10^-14, 9.9476×10^-14},
   {1.2168×10^-12, 526.478, -4.1922×10^-13,
    3.58824×10^-13, 6.03961×10^-14, -5.68434×10^-13,
    2.66454×10^-13, -6.75016×10^-14, -1.95399×10^-13,
    6.75016×10^-14, -1.3145×10^-13, 2.4869×10^-13},
   {-2.49578×10^-13, -4.93827×10^-13, 389.108,
    -8.13571×10^-13, -2.34479×10^-13, 1.53477×10^-12,
    -5.3646×10^-13, -2.11386×10^-12, 1.99307×10^-12,
```

Iz izpisa je razvidno, da so izvediagonalni členi zelo majhni, zato se uporabi ukaz *Chop[]* ter izpis v matrični obliki *MatrixForm[]:*

```
In[27]:= MatrixForm[Chop[%]]

Out[27]//MatrixForm=
680.055     0       0       0       0       0       0       0       0       0       0       0
    0   526.478     0       0       0       0       0       0       0       0       0       0
    0       0   389.108     0       0       0       0       0       0       0       0       0
    0       0       0   281.618     0       0       0       0       0       0       0       0
    0       0       0       0   209.469     0       0       0       0       0       0       0
    0       0       0       0       0   199.343     0       0       0       0       0       0
    0       0       0       0       0       0   64.2979     0       0       0       0       0
    0       0       0       0       0       0       0   47.023      0       0       0       0
    0       0       0       0       0       0       0       0   30.9821     0       0       0
    0       0       0       0       0       0       0       0       0   19.7551     0       0
    0       0       0       0       0       0       0       0       0       0   13.3724     0
    0       0       0       0       0       0       0       0       0       0       0   3.66513
```

kar potrdi uspešnost izračuna.

Slično se izvede kontrola še za togostno matriko (tudi ta izpis ni prikazan v celoti):

```
In[28]:= vektorji.Kk.Transpose[vektorji]

Out[28]= {{315270., 2.48401×10^-8, -3.41552×10^-8,
    -8.73843×10^-8, -3.25599×10^-8, 9.46129×10^-8,
    -1.01383×10^-7, 8.55289×10^-8, 2.5284×10^-8,
    -5.05206×10^-8, -1.1607×10^-7, 2.20753×10^-8},
   {2.1173×10^-8, 9.5903×10^6, -5.11645×10^-8,
    2.66184×10^-7, 2.56696×10^-7, -9.73465×10^-7,
    7.71601×10^-7, -8.31555×10^-7, -1.24366×10^-6,
    1.39931×10^-7, -7.82718×10^-7, -6.51577×10^-7},
   {-2.09548×10^-8, -1.3737×10^-8, 5.57469×10^7,
    -1.2829×10^-6, -8.5216×10^-8, 7.19726×10^-6,
    -2.60957×10^-6, 4.03542×10^-6, 7.00401×10^-6,
```

Tudi sedaj je razvidno, da so izvediagonalni členi manjši kot diagonalni, vendar uporaba osnovne oblike ukaza *Chop[]* ter izpisa v matrični obliki *MatrixForm[]*

```
In[29]:= MatrixForm[Chop[%]]

Out[29]//MatrixForm=
```

315270.	2.48401×10^{-8}	-3.41552×10^{-8}	-8.73843×10^{-8}	-3.25599×10^{-8}	9.46129×10^{-8}	-1.01383×10^{-7}
2.1173×10^{-8}	9.5903×10^{6}	-5.11645×10^{-8}	2.66184×10^{-7}	2.56696×10^{-7}	-9.73465×10^{-7}	7.71601×10^{-7}
-2.09548×10^{-8}	-1.3737×10^{-8}	5.57469×10^{7}	-1.2829×10^{-6}	-8.5216×10^{-8}	7.19726×10^{-6}	-2.60957×10^{-6}
-5.1572×10^{-8}	2.59373×10^{-7}	-1.22795×10^{-6}	1.56364×10^{8}	-1.28988×10^{-6}	7.65827×10^{-6}	-9.56189×10^{-6}
-4.17931×10^{-8}	2.62167×10^{-7}	-1.14087×10^{-7}	-1.20793×10^{-6}	3.23264×10^{8}	0.0000193594	0.0000172658
9.13278×10^{-8}	-9.97679×10^{-7}	7.15302×10^{-6}	7.70693×10^{-6}	0.0000193599	6.88715×10^{8}	0.0000178791
-6.61239×10^{-8}	7.24569×10^{-7}	-2.57418×10^{-6}	-9.47714×10^{-6}	0.0000172481	0.0000179	5.28155×10^{8}
5.12227×10^{-8}	-8.82894×10^{-7}	4.02331×10^{-6}	0.0000228956	-4.20958×10^{-6}	-0.0000407249	-0.000255622
5.58794×10^{-8}	-1.2368×10^{-6}	7.0557×10^{-6}	0.0000259131	-0.0000566244	0.0000421107	-0.000257596

sedaj še ne vodi do pričakovane oblike, saj so diagonalni členi (in posledično tudi izvendiagonalni) precej večjega ranga kot je to bilo pri masni matriki in zato je potrebno postaviti strožjo (višjo) mejo za zaokroževanje pri ukazu *Chop[]*, npr. 10^{-2}, ki pa je, glede na velikost diagonalnih členov, inženirsko vseeno popolnoma opravičljiva:

```
In[30]:= MatrixForm[Chop[%, 10^-2]]

Out[30]//MatrixForm=
```

315270.	0	0	0	0	0	0	0	0	0	0	0
0	9.5903×10^{6}	0	0	0	0	0	0	0	0	0	0
0	0	5.57469×10^{7}	0	0	0	0	0	0	0	0	0
0	0	0	1.56364×10^{8}	0	0	0	0	0	0	0	0
0	0	0	0	3.23264×10^{8}	0	0	0	0	0	0	0
0	0	0	0	0	6.88715×10^{8}	0	0	0	0	0	0
0	0	0	0	0	0	5.28155×10^{8}	0	0	0	0	0
0	0	0	0	0	0	0	7.28719×10^{8}	0	0	0	0
0	0	0	0	0	0	0	0	8.96561×10^{8}	0	0	0
0	0	0	0	0	0	0	0	0	1.04595×10^{9}	0	0
0	0	0	0	0	0	0	0	0	0	1.22465×10^{9}	0
0	0	0	0	0	0	0	0	0	0	0	6.38165×10^{8}

Lastne vektorje, ki pripadajo posameznim lastnim frekvencam, je smiselno tudi izrisati. Da se izvede izris, ki bo imel na abscisi dejanske koordinate oz. dejanske dimenzije konstrukcije (ne pa zaporedne številke členov vektorja), se definira vektor, ki bo zajel koordinate vozlišč med končnimi elementi, kjer so tudi izračunane vrednosti lastnih vektorjev:

```
In[31]:= koordinate = Table[L / Ne i, {i, 0, Ne}]

Out[31]= {0, 5/3, 10/3, 5, 20/3, 25/3, 10}
```

Ker je bil računski model sestavljen z metodo končnih elementov, v vsakem vozlišču nastopita dve prostostni stopnji – prečni pomik in zasuk. Posledično to pomeni, da obravnavanem primeru lihi členi posameznega vektorja (vrstice) predstavljajo prečne pomike, sodi pa zasuke. Izmed vseh členov lastnega vektorja jih je torej za izris direktno uporabnih zgolj polovica (lihi členi) in zato se definira vektor kot:

```
In[32]:= lastnivektor = Table[0, {i, 0, Ne}];
```

Prvi lastni vektor (ki pripada prvi lastni frekvenci) se sedaj dobi tako, da se iz prve vrstice matrike *vektorji* v vektor *lastnivektor* prenesejo po vrsti vsi lihi členi (indeks *i+1* zagotovi, da bo prvi člen vektorja enak 0, kar sicer odgovarja začetni vrednosti na mestu vpetja, vendar te vrednosti ni v matriki *vektorji*, ker ta prostostna stopnja v analizi ne nastopa):

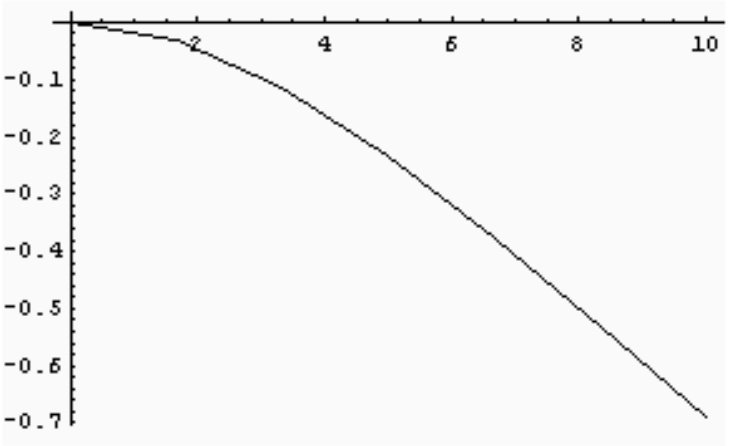
```
In[33]:= Do[lastnivektor[[i + 1]] = vektorji[[1, 2 i - 1]],
          {i, 1, Ne}]
```

Da bodo vrednosti pomikov izrisane s pravimi koordinatami na abscisi (izris vektorja *lastnivektor* je sicer možen že sedaj, vendar bodo na abscisi nastopile zaporedne številke členov in ne koordinate), potrebno tvoriti dvostolpčno matriko, katere prvi stolpec (bodočo absciso) bodo predstavljale že zgoraj definirane koordinate, drugega pa vrednosti vektorja oz. prečni pomiki (bodoča ordinata). Da se dvovrstična matrika pretvori v dvostolpčno, se izvede transponiranje z uporabo ukaza *Transpose[matrika]*:

```
In[34]:= tabela = Transpose[{koordinate, lastnivektor}];
```

Za izris se najprej uporabi ukaz *Line[]*, ki zgolj pripravi podatke za grafiko tako, da z linijami poveže točke v tabeli, nato pa se zahteva še prikaz pripravljenih podatkov z ukazom *Show[Graphics[]]*. Da pa se še doseže izris koordinatnih osi, se ukazu doda še opcija *Axes→Automatic*.

```
In[35]:= linija = Line[tabela];
         Show[Graphics[linija], Axes → Automatic];
```



Opomba: do enakega izrisa je mogoče priti tudi v enem koraku z ukazom *ListPlot[tabela,PlotJoined->True]*.

Uporabljen postopek je mogoče smiselno uporabiti še za izris preostalih lastnih vektorjev (v vhodni vrstici 33 se zamenja številka lastnega vektorja (številka vrstice v matriki *vektorji*):

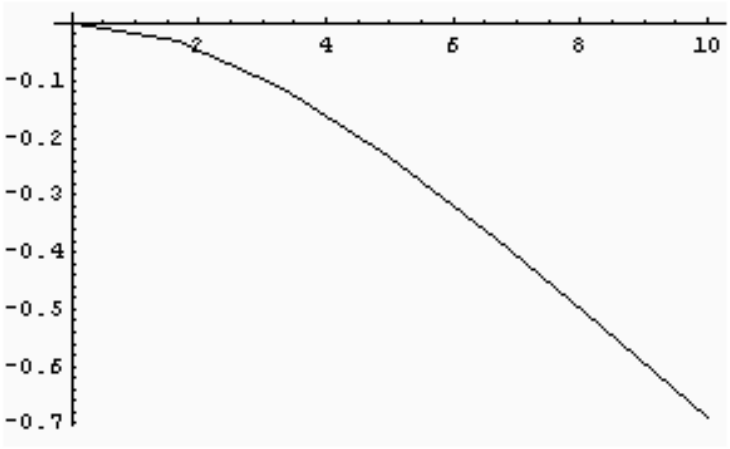
```
In[33]:= Do[lastnivektor[[i + 1]] = vektorji[[1, 2 i - 1]],
          {i, 1, Ne}]
```

Vendar je tudi ta proces mogoče avtomatizirati. Zato se definira vektor, kamor se bodo (npr. zaradi kasnejše možnosti primerjave) shranili slike vseh lastnih vektorjev:

```
In[37]:= Slikelastnihvektorjev = Table[0, {2 Ne}];
```

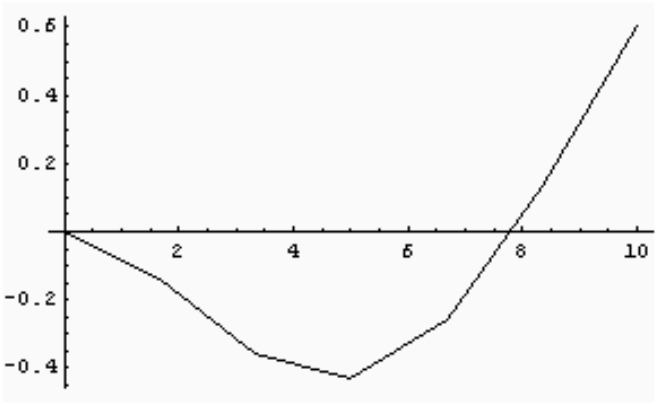
Sedaj se vhodne vrstice 33, 34 in 35 smiselno združijo v zanko, katere spremenljivka definira lastni vektor. Zaradi lažjega ločevanja posameznih vektorjev, ki se bodo izrisali, se pred vsako sliko npr. še izpiše komentar, ki pove, kateri lastni vektor sploh je izrisan, ter tudi njegove vrednosti. Za oba izpisa se uporabi ukaz *Print[spremenljivka]* oz. *Print["komentar", spremenljivka]*:

```
In[38]:= Do[Print["lastni vektor ", j];
          Do[lastnivektor[[i + 1]] = vektorji[[j, 2 i - 1]],
           {i, 1, Ne}]; Print[lastnivektor];
         tabela = Transpose[{koordinate, lastnivektor}];
         linija = Line[tabela];
         Slikelastnihvektorjev[[j]] =
           Show[Graphics[linija], Axes → Automatic], {j, 2 Ne }]

lastni vektor 1

{0, -0.0309939, -0.11376, -0.233327,
 -0.375869, -0.529821, -0.687221}
```

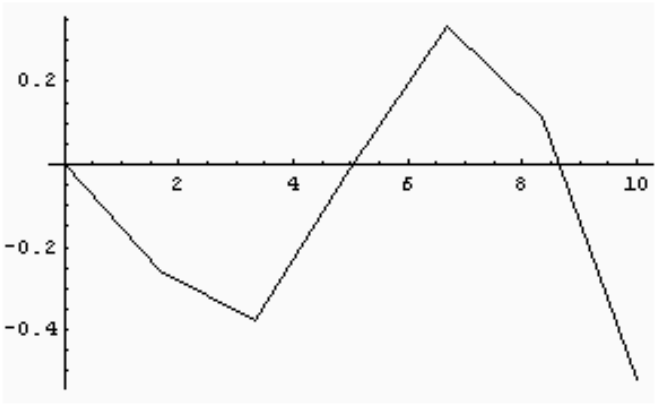


Dejansko pa se izpišejo in izrišejo ter v istem zaporedju v vektor *Slikelastnihvektorjev* še shranijo vsi lastni vektorji, katerih krivulja pa postaja s frekvenco vedno bolj lomljena:

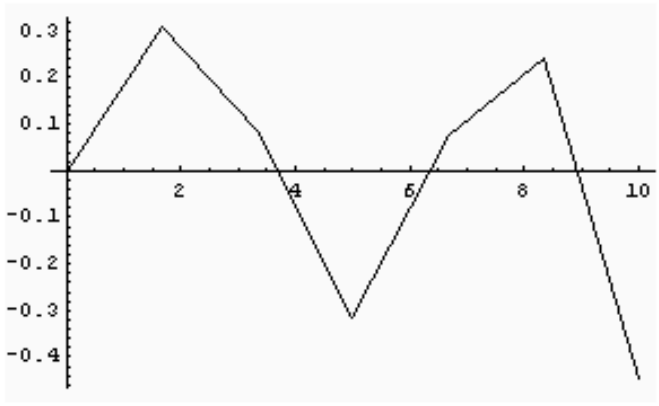
```
lastni vektor 2

{0, -0.136135, -0.356707,
 -0.431738, -0.255724, 0.130863, 0.604952}
```




```
lastni vektor 3

{0, -0.255095, -0.376674,
 -0.0103764, 0.335793, 0.11363, -0.521641}
```

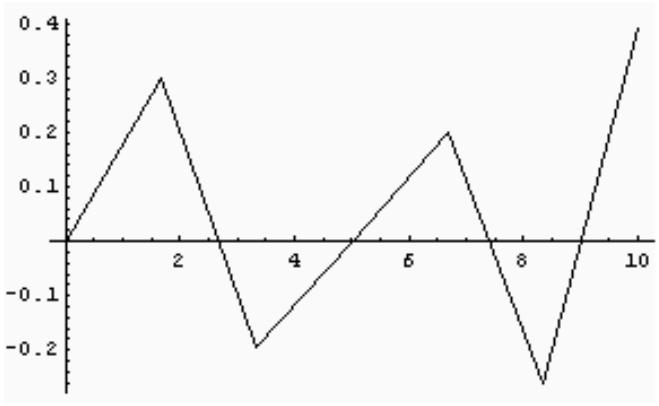



```
lastni vektor 4

{0, 0.310242, 0.0880365, -0.316904,
 0.0756225, 0.239048, -0.447326}
```



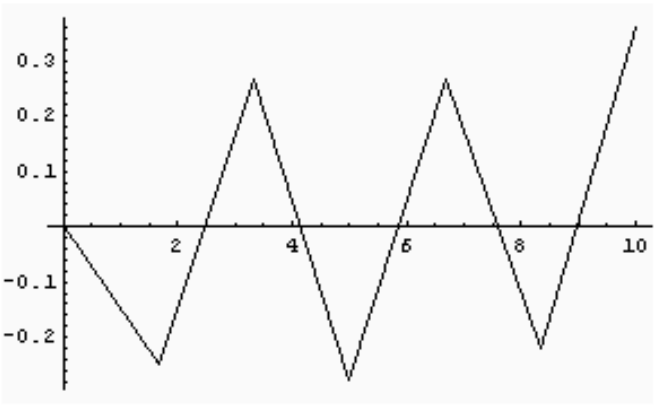
lastni vektor 5

{0, 0.29772, -0.19444, -0.0019636,
 0.201345, -0.259973, 0.389957}



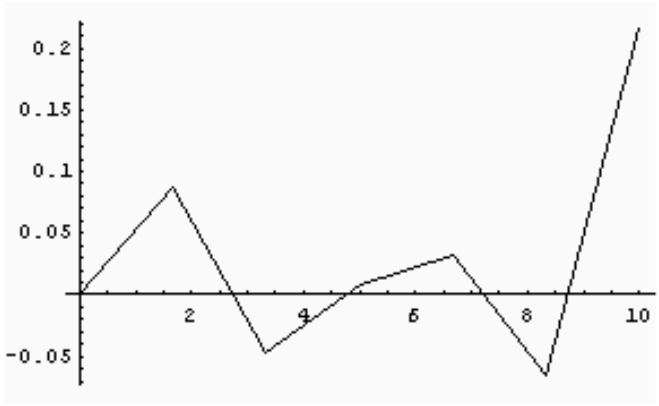
lastni vektor 6

{0, -0.248113, 0.267843,
 -0.278003, 0.265417, -0.218874, 0.361015}

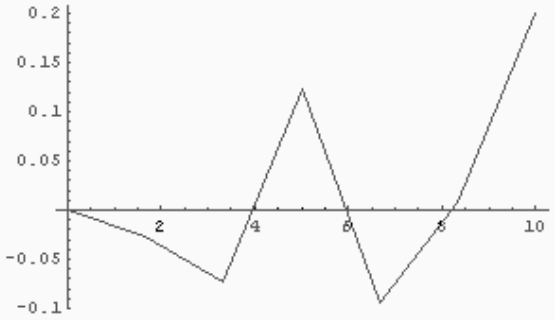


lastni vektor 7

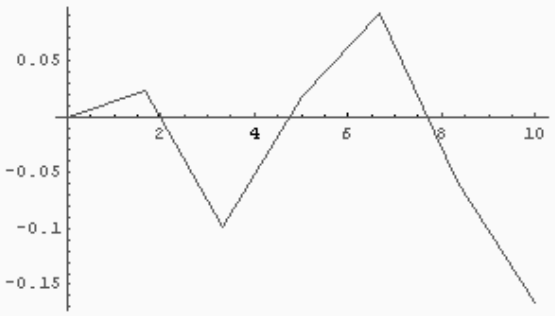
{0, 0.0866607, -0.047673, 0.00796009,
 0.0328337, -0.0661242, 0.214934}



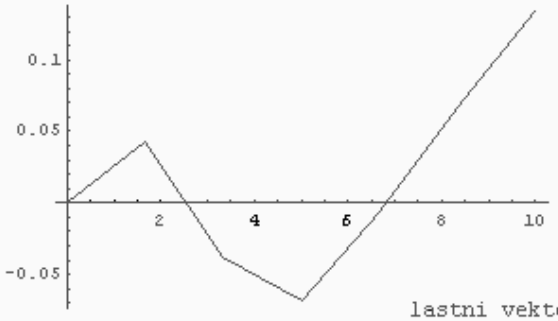
```
lastni vektor 8

{0, -0.0269391, -0.0728574,
 0.122973, -0.0936648, 0.00665254, 0.200737}
```



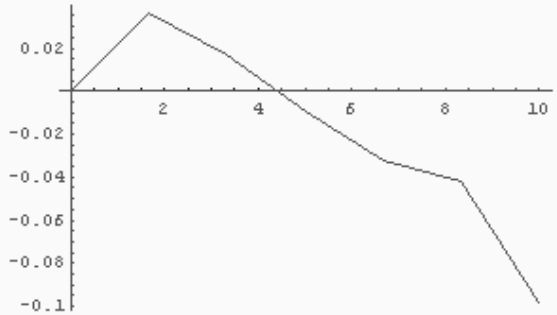

```
lastni vektor 9

{0, 0.0224506, -0.0988918,
 0.0177508, 0.0913156, -0.058241, -0.16683}
```



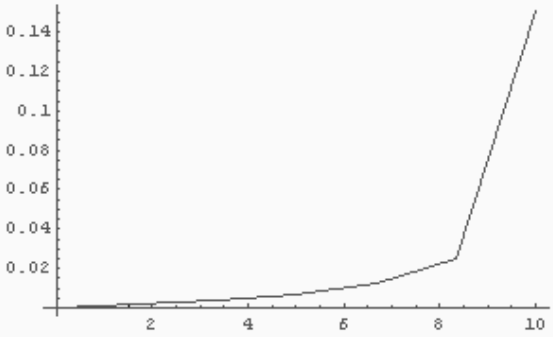

```
lastni vektor 10

{0, 0.0430312, -0.0382548, -0.069003,
 -0.00669012, 0.0659736, 0.133358}
```




```
lastni vektor 11

{0, 0.0360735, 0.0172612, -0.00946841,
 -0.0322024, -0.0422251, -0.0983968}
```




```
lastni vektor 12

{0, 0.00243775, 0.00374809,
 0.00666303, 0.0125078, 0.025286, 0.150114}
```



Čeprav se izrišejo še slike vseh preostalih lastnih vektorjev, je že iz slike sedmega vektorja jasno, da ne predstavlja korektne nihajne oblike, saj ne zadosti pogoju, da mora nihajna oblika pri konzoli imeti eno ničlo manj kot je številka pripadajoče lastne frekvence (npr. peta nihajna oblika mora imeti 4 ničle).

Iz tega je mogoče zaključiti, da je pri modeliranju z metodo končnih elementov uporabnih zgolj največ polovica nihajnih oblik.

Zato se lahko procedura ponovi zgolj s prvimi šestimi vektorji, najprej pa se vektor, kamor se shranjujejo slike lastnih vektorjev, ponovno definira (saj bi drugače v njemu ostale shranjene tudi višje, nepotrebne oz. netočne nihajne oblike):

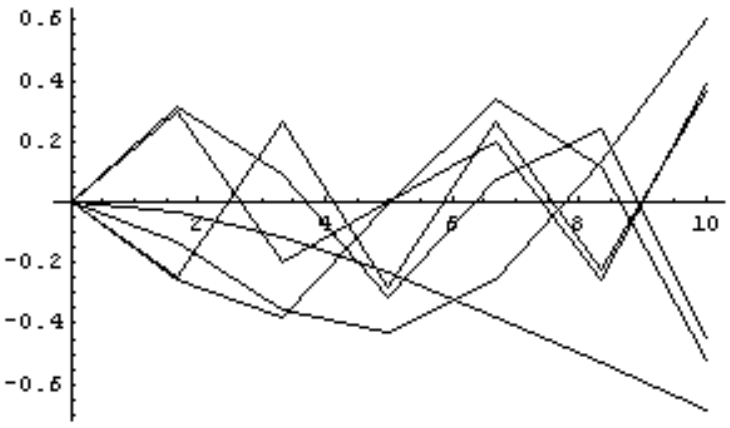
```
In[39]:= Slikelastnihvektorjev = Table[0, {Ne}];
```

in procedura za izris vseh vektorjev se izvede ponovno, vendar tokrat z manjšim, a bolj realnim številom lastnih vektorjev (zgolj polovica vseh):

```
In[40]:= Do[Print["lastni vektor ", j];
          Do[lastnivektor[[i + 1]] = vektorji[[j, 2 i - 1]],
           {i, 1, Ne}]; Print[lastnivektor];
         tabela = Transpose[{koordinate, lastnivektor}];
         linija = Line[tabela];
         Slikelastnihvektorjev[[j]] =
           Show[ Graphics[linija], Axes → Automatic], {j, Ne }]
```

Izrisani vektorji (slike so izpuščene) seveda enaki kot prej, le da so sedaj izrisani zgolj inženirsko zanimivi vektorji. Ker so vsi shranjeni v vektorju *Slikelastnihvektorjev*, jih je sicer mogoče vse naenkrat tudi izrisati na skupni sliki, vendar to ne prinese nekega dodatnega vpogleda:

```
In[41]:= Show[Slikelastnihvektorjev];
```



Dokaj nazobčane lastne vektorje predvsem višjih nihajnih oblik pa je mogoče še izboljšati, saj so dejansko znane informacije ne samo o diskretnih vrednostih funkcije (pomiki), temveč tudi o odvodih funkcije v teh točkah (zasuki), kar omogoča ne samo precejšnje kvalitetno izboljšanje slike, ampak omogoča tudi pridobitev informacije o zvezni funkciji – polinomu, ki ustreza izračunanim vrednostim. Ker bodo informacije podane v obliki para *{vrednost funkcije, vrednost odvoda}* v neki diskretni točki, se najprej pripravi vektor, kamor so bodo shranjevali pari informacij o (bodoči) krivulji:

■ Kvalitetnejsi izris lastnih vektorjev in pridobitev polinoma

```
In[42]:= pari = Table[{0, 0}, {i, 0, Ne}];
```

Da se procedura napiše čimbolj splošno, se najprej definira številka lastnega vektorja, npr. *j*, ki se bo obravnaval, nato pa se zanj prenesejo pripadajoči pari podatkov iz matrike *vektorji* v vektor *pari*:

```
In[43]:= j = 1;
         Do[pari[[i + 1, 1]] = vektorji[[j, 2 i - 1]];
          pari[[i + 1, 2]] = vektorji[[j, 2 i ]], {i, 1, Ne}]
```

Sedaj je potrebno podatkom o funkcijskih vrednostih in pripadajočih odvodih dodati še podatke o točkah, torej o koordinatah na abscisi in sicer v obliki, ki je primerna za izračun interpoliranega polinoma :

```
In[45]:= podatki = Transpose[{koordinate, pari}]

Out[45]= {{0, {0, 0}}, {5/3, {-0.248113, 0.239312}},
          {10/3, {0.267843, -0.108703}},
          {5, {-0.278003, -0.00987994}},
          {20/3, {0.265417, 0.130126}},
          {25/3, {-0.218874, -0.21896}},
          {10, {0.361015, 0.637707}}}
```

Z ukazom *InterpolatingPolynomial[podatki, argument]* se sedaj poišče polinom, ki ustreza vhodnim podatkom, in je funkcija spremenljivke *argument*:

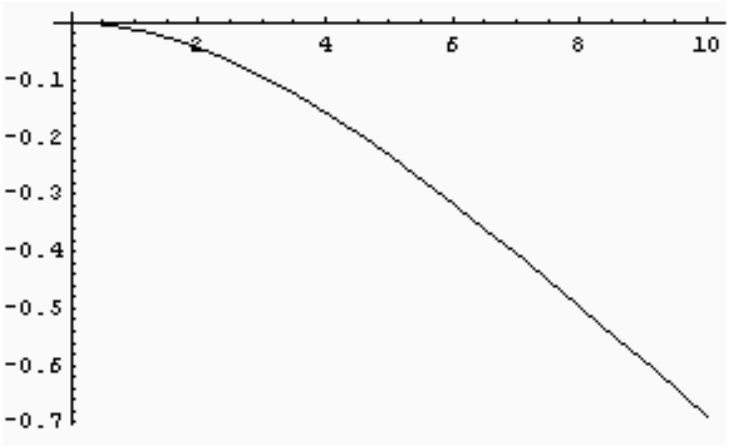
```
In[46]:= polinom = InterpolatingPolynomial[podatki, x]

Out[46]= (-0.0893205 +
            (0.193337 + (-0.0751688 + (0.0118723 + (0.00159066 +
                  (-0.0013076 + (0.000235857 +
                        (-8.67696 × 10^-6 + (-4.60462 × 10^-6 +
                              (1.21879 × 10^-6 +
                              (-1.20057 × 10^-7 - 2.589 × 10^-10
                              (-10 + x)) (-25/3 + x)) (-25/3 +
                              x)) (-20/3 + x)) (-20/3 + x))
                        (-5 + x)) (-5 + x)) (-10/3 + x))
                  (-10/3 + x)) (-5/3 + x)) (-5/3 + x)) x^2
```

Stopnjo polinoma *Mathematica* določi avtomatično in je seveda odvisna od števila podatkov. Ker *Mathematica* izpelje rezultat – dobljeni polinom v neurejeni obliki, ga je zato smiselno izpisati v urejeni obliki:

```
In[47]:= Expand[%]

Out[47]= -0.0120814 x^2 + 0.000554353 x^3 - 1.90394 × 10^-8 x^4 +
         1.26987 × 10^-8 x^5 - 4.68856 × 10^-8 x^6 + 2.35759 × 10^-9 x^7 -
         3.02974 × 10^-10 x^8 + 4.10964 × 10^-11 x^9 - 3.79179 × 10^-12 x^10 +
         2.25514 × 10^-13 x^11 - 7.84809 × 10^-15 x^12 + 1.21183 × 10^-16 x^13
```

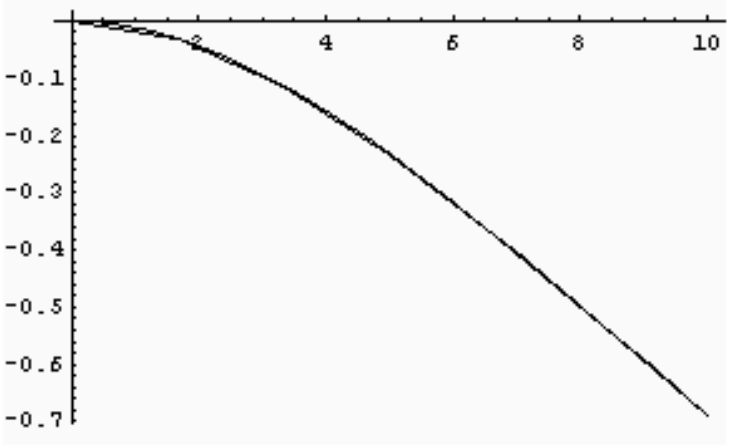
Njegova slika je:

```
In[48]:= krivulja = Plot[polinom, {x, 0, L}];
```



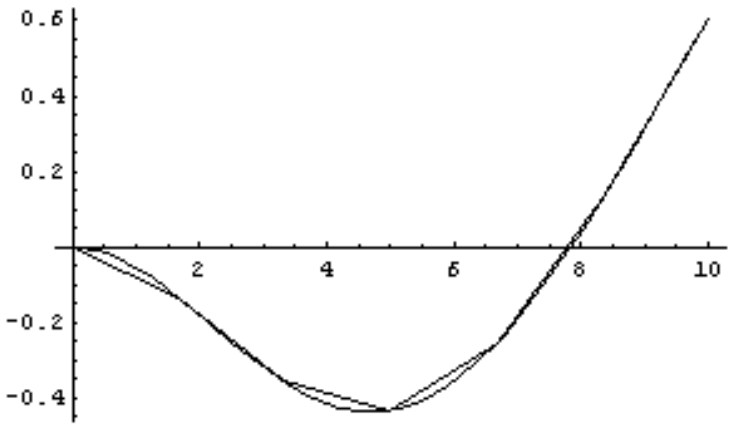
Za oceno uspešnosti dobljenega polinoma je potrebno grafično primerjati oba rezultata:

```
In[49]:= Show[krivulja, Slikelastnihvektorjev[[j]]];
```



Ker gre za prvi lastni vektor, je razlika minimalna in bistveni napredek mogoče opaziti šele pri višjih vektorjih. Tako sledi npr. drugi lastni vektor:

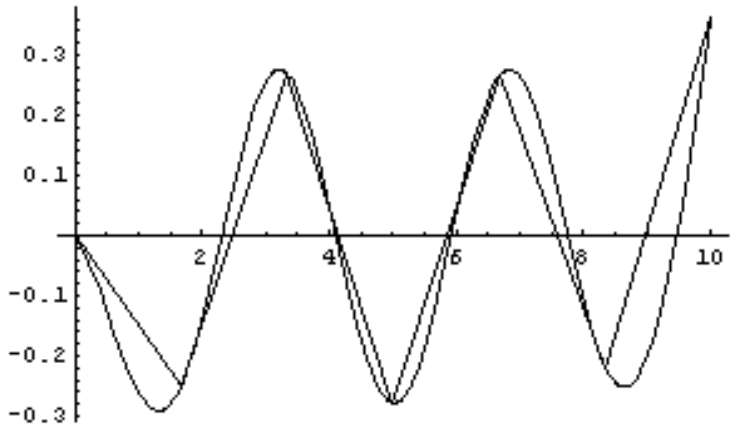
```
In[56]:= Show[krivulja, Slikelastnihvektorjev[[j]]];
```



kjer je napredek že opaznejši.

Najbolj pa je razlika opazna pri zadnjem, v obravnavanem primeru šestem lastnem vektorju:

```
In[84]:= Show[krivulja, Slikelastnihvektorjev[[j]]];
```



Dobljeno krivuljo pa je mogoče sedaj primerjati z nihajno obliko, dobljeno v podpoglavju II.5, ki se po že znanem postopku prenese v datoteko in ustrezno poimenuje, npr. *analiticna*:

```
In[85]:= analiticna[x_] :=
          C[1] * Cos[(17.278759532088237 * x) / L] -
           C[1] * Cosh[(17.278759532088237 * x) / L] -
           1.0000000626556285 * C[1] *
            Sin[(17.278759532088237 * x) / L] +
           1.0000000626556285 * C[1] *
            Sinh[(17.278759532088237 * x) / L]
```

V funkciji nastopa še neznana nedoločena konstanta C[1] in njeno vrednost je (zaradi lažje grafične primerjave z interpolirano krivuljo iz metode končnih elementov) najprimerneje določiti npr. iz pogoja, da je pomik na prostem koncu enak pri obeh funkcijah:

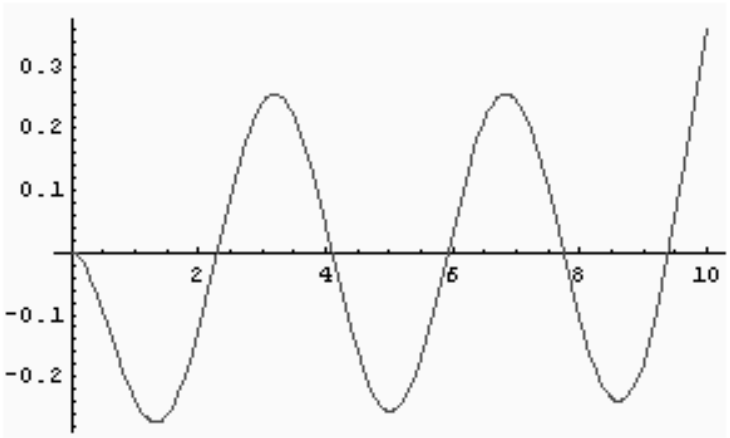
```
In[86]:= c1 = Solve[analiticna[L] == (polinom /. x → L), C[1]]

Out[86]= {{C[1] → 0.180508}}
```

Funkcija pomika, sedaj normirana na pomik, je tako:

```
In[87]:= normirana[x_] = analiticna[x] /. c1[[1]]

Out[87]= 0.180508 Cos[1.72788 x] - 0.180508 Cosh[1.72788 x] -
         0.180508 Sin[1.72788 x] + 0.180508 Sinh[1.72788 x]
```

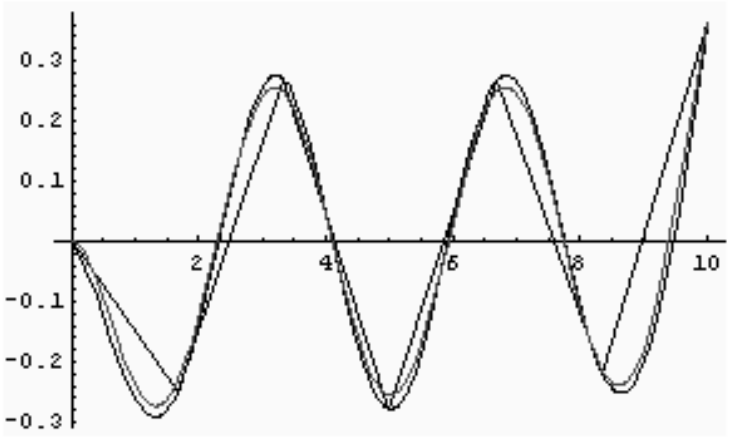
kar omogoči najprej samostojni izris te funkcije:

```
In[88]:= Plot[normirana[x], {x, 0, L},
           PlotStyle → RGBColor[1, 0, 0]];
```



nato pa še njeno primerjavo z interpolirano funkcijo na osnovi z metodo končnih elementov izračunanega lastnega vektorja:

```
In[89]:= Show[krivulja, Slikelastnihvektorjev[[j]], %];
```



Zapisano proceduro je mogoče uporabiti za diskretizacijo s poljubnim številom končnih elementov, saj se za ponovni izračun z novim številom končnih elementov v 3 vrstici zgolj spremeni spremenljivka *Ne* in se ves izračun ponovi, vendar je potrebno še prej izbrisati vrednosti spremenljivke ω z ukazom *Clear[]*:

```
In[90]:= Clear[ω]
```