Turinys

[Funkcijos interpoliavimas daugianariu 2](#_Toc342737508)

[Taškai pasiskirstę vienodai 3](#_Toc342737509)

[Naudojant Čiobyševo abscises 6](#_Toc342737510)

[Funkcijų aproksimavimas 9](#_Toc342737511)

[Diskrečioji Furje aproksimacija 9](#_Toc342737512)

[Haro bangelių aproksimacija 12](#_Toc342737513)

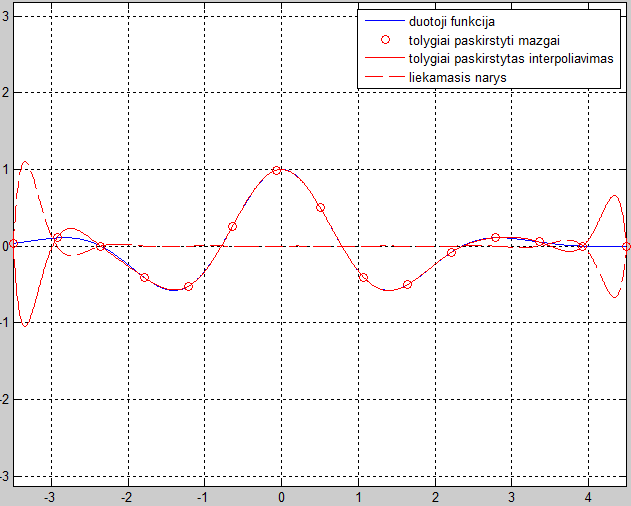
# Funkcijos interpoliavimas daugianariu

Užduotis:

|  |  |  |
| --- | --- | --- |
| Nr. | Interpoliavimo funkcijos išraiška | Bazinės funkcijos |
| 9 |  | Niutono |

1. Užduotis.

## Taškai pasiskirstę vienodai



2. Niutono interpoliavimo rezultatai kai mazgai tolygiai pasiskirstę.

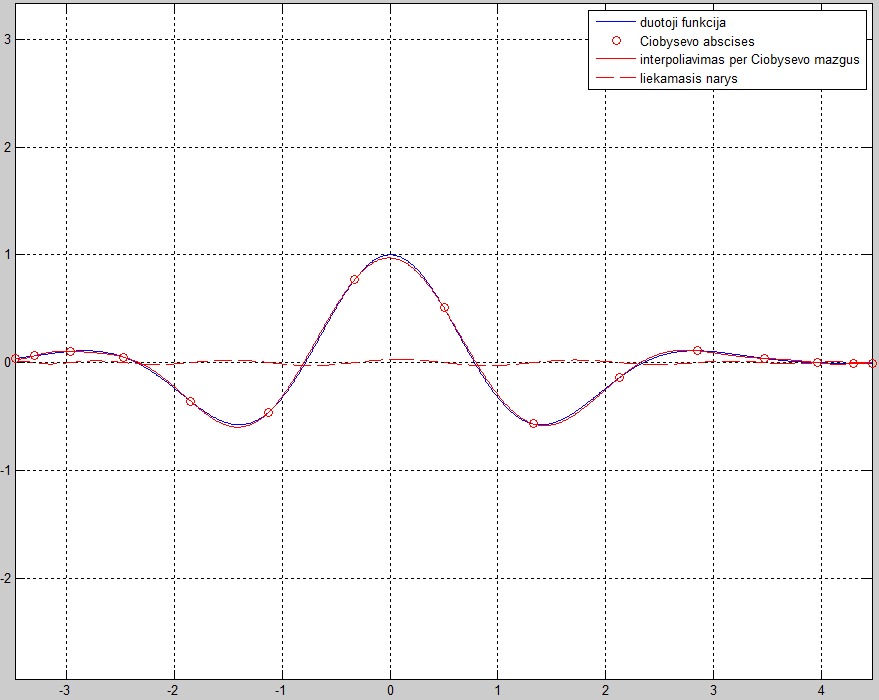
|  |  |  |
| --- | --- | --- |
| **Ordinatės** | **Abscisės** | **Funkcijos koeficientų vektorius** |
| 0.0353  0.1067  0.0005  -0.4096  -0.5232  0.2536  0.9886  0.5076  -0.4063  -0.5040  -0.0822  0.1088  0.0543  -0.0001  -0.0058 | -3.5000  -2.9286  -2.3571  -1.7857  -1.2143  -0.6429  -0.0714  0.5000  1.0714  1.6429  2.2143  2.7857  3.3571  3.9286  4.5000 | 0.0353  0.1250  -0.2720  -0.1127  0.2839  -0.1003  -0.0314  0.0358  -0.0120  0.0012  0.0006  -0.0003  0.0001  -0.0000  0.0000 |

3. Tolygiai paskirstytos abscises, jas atitinkančios ordinatės ir bazinės funkcijos koeficientų vektorius.

Programos kodas

|  |
| --- |
| function pagrindine  clc,close all    xmin=-3.5; xmax=4.5; % duotas funkcijos apibrezimo intervalas  N = 15; % interpoliavimo tasku skaicius    %naudoti='ciobysevo'  naudoti='tolygiai'    if strcmp(naudoti,'tolygiai') == 1  X=[xmin:(xmax-xmin)/(N-1):xmax] % tolygiai paskirstytu interpoliavimo tasku abscises  Y=funkcija(X); % tolygiai paskirstytu interpoliavimo tasku ordinates  leg={'duotoji funkcija',...  'tolygiai paskirstyti mazgai',...  'tolygiai paskirstytas interpoliavimas',...  'liekamasis narys',...  };  else  leg={'duotoji funkcija',...  'Ciobysevo abscises',...  'interpoliavimas per Ciobysevo mazgus',...  'liekamasis narys',...  };  k=[0:N-1];  X=(xmax+xmin)/2+(xmax-xmin)/2\*cos((2\*k+1)\*pi/(2\*N)); % "Ciobysevo abscises"  Y=funkcija(X); % ordinates "Ciobysevo abscisiu" taskuose  end    x=min(X):(max(X)-min(X))/1000:max(X); %x reiksmes vaizdavimui    figure(1), hold on, grid on,box on,axis equal%, set(gcf,'Color','w');  plot(x,funkcija(x),'b-') % vaizduojama duotoji funkcija (t.y. pagal kuria interpoliuojama)  plot(X,Y,'ro') % vaizduojami tolygiai isdestyti interpoliavimo taskai    F=niuton(X,Y,x);    plot(x,F,'r-') % vaizduojama funkcija, interpoliuojanti tolygiai paskirstytuose mazguose  plot(x,funkcija(x)-F,'r--'), % vaizduojama netiktis duotos funkcijos atzvilgiu  legend(leg{1:4});  return  end    function fv=niuton(x,y,t)  % NIUTON apskai?iuoja interpoliacinio polinomo,  % nusakyto interpoliavimo taškais (x(i),y(i)),i=1,2,...,n+1),  % reikšmes fv, kai argumento reikšmes apibr?žia masyvo t elementai.  % Polinomo reikšm?s skai?iuojamos pagal Niutono iterpoliacin?form?.  % ??jimo parametrai  % (x,y) - interpoliavimo taškai,  % t - argumento reikšmi?masyvas.  % Iš?jimo parametrai  %fv - interpoliacinio polinomo reikšm?s.  n=numel(x)-1;%interpoliavimo tasku kiekis  m=numel(t); %argumentu kiekis    [k,l]=size(t);  if k ==1  t=t';  end    [k,l]=size(x);    if k ~=1  x=x'; y=y';  end    d=y;    for k=1:n  h=x(k+1:end)-x(1:end-k);  tt=(d(k+1:end)-d(k:end-1))./h;  d(k+1:end)=tt;  end    xx=repmat(x,m,1);  dd=repmat(d,m,1);  tt=repmat(t,1,n);    p=tt-xx(:,1:end-1);  r=ones(m,1);  s=[r cumprod(p,2)];    fv=sum((dd.\*s)');  end    function fnk=funkcija(x)  % apskaiciuoja interpoliuojamos funkcijos reiksmes taskuose x  fnk=cos(2.\*x).\*exp(-(x/2).^2);  return  end |

## Naudojant Čiobyševo abscises



4.Niutono interpoliavimo rezultatai kai naudojamos Čiobyševo abscises.

|  |  |  |
| --- | --- | --- |
| **Ordinatės** | **Abscisės** | **Funkcijos koeficientų vektorius** |
| -0.0059  -0.0067  -0.0015 0.0387 0.1095  -0.1428  -0.5699 0.5076 0.7666  -0.4595  -0.3596  0.0500 0.1043 0.0618 0.0380 | 4.4781 4.3042 3.9641 3.4726 2.8511 2.1269 1.3316 0.5000  -0.3316  -1.1269  -1.8511  -2.4726  -2.9641  -3.3042  -3.4781 | -0.0059  0.0042  0.0381  -0.0415  -0.0470  0.0128  0.0247  0.0100  0.0012  -0.0005  -0.0003  -0.0001  -0.0000  -0.0000  0.0000 |

5 Čiobyševo abscises, jas atitinkančios ordinatės ir bazinės funkcijos koeficientų vektorius.

Programos kodas

|  |
| --- |
| function pagrindine  clc,close all  xmin=-3.5; xmax=4.5; % duotas funkcijos apibrezimo intervalas  N = 15; % interpoliavimo tasku skaicius    naudoti='ciobysevo'  %naudoti='tolygiai'    if strcmp(naudoti,'tolygiai') == 1  X=[xmin:(xmax-xmin)/(N-1):xmax] % tolygiai paskirstytu interpoliavimo tasku abscises  Y=funkcija(X) % tolygiai paskirstytu interpoliavimo tasku ordinates  leg={'duotoji funkcija',...  'tolygiai paskirstyti mazgai',...  'tolygiai paskirstytas interpoliavimas',...  'liekamasis narys',...  };  else  leg={'duotoji funkcija',...  'Ciobysevo abscises',...  'interpoliavimas per Ciobysevo mazgus',...  'liekamasis narys',...  };  k=[0:N-1]  X=(xmax+xmin)/2+(xmax-xmin)/2\*cos((2\*k+1)\*pi/(2\*N)) % "Ciobysevo abscises"  Y=funkcija(X) % ordinates "Ciobysevo abscisiu" taskuose  end    x=min(X):(max(X)-min(X))/1000:max(X); %x reiksmes vaizdavimui    figure(1), hold on, grid on,box on,axis equal%, set(gcf,'Color','w');  plot(x,funkcija(x),'b-') % vaizduojama duotoji funkcija (t.y. pagal kuria interpoliuojama)  plot(X,Y,'ro') % vaizduojami tolygiai isdestyti interpoliavimo taskai    F=niuton(X,Y,x);    plot(x,F,'r-') % vaizduojama funkcija, interpoliuojanti tolygiai paskirstytuose mazguose  plot(x,funkcija(x)-F,'r--'), % vaizduojama netiktis duotos funkcijos atzvilgiu  legend(leg{1:4});  return  end    function fv=niuton(x,y,t)  % NIUTON apskai?iuoja interpoliacinio polinomo,  % nusakyto interpoliavimo taškais (x(i),y(i)),i=1,2,...,n+1),  % reikšmes fv, kai argumento reikšmes apibr?žia masyvo t elementai.  % Polinomo reikšm?s skai?iuojamos pagal Niutono iterpoliacin?form?.  % ??jimo parametrai  % (x,y) - interpoliavimo taškai,  % t - argumento reikšmi?masyvas.  % Iš?jimo parametrai  %fv - interpoliacinio polinomo reikšm?s.  n=numel(x)-1;%interpoliavimo tasku kiekis  m=numel(t); %argumentu kiekis    [k,l]=size(t);  if k ==1  t=t';  end    [k,l]=size(x);    if k ~=1  x=x'; y=y';  end    d=y;    for k=1:n  h=x(k+1:end)-x(1:end-k);  tt=(d(k+1:end)-d(k:end-1))./h;  d(k+1:end)=tt  end    xx=repmat(x,m,1);  dd=repmat(d,m,1);  tt=repmat(t,1,n);    p=tt-xx(:,1:end-1);  r=ones(m,1);  s=[r cumprod(p,2)];    fv=sum((dd.\*s)');  end    function fnk=funkcija(x)  % apskaiciuoja interpoliuojamos funkcijos reiksmes taskuose x  fnk=cos(2.\*x).\*exp(-(x/2).^2);  return  end |

# Funkcijų aproksimavimas

|  |  |  |
| --- | --- | --- |
| Nr | Periodinė funkcija F(t) | Triukšmas R(t) |
| 9 |  |  |

6. Užduotis



7. Paveikslėlis Haro bangelių aproksimacijai

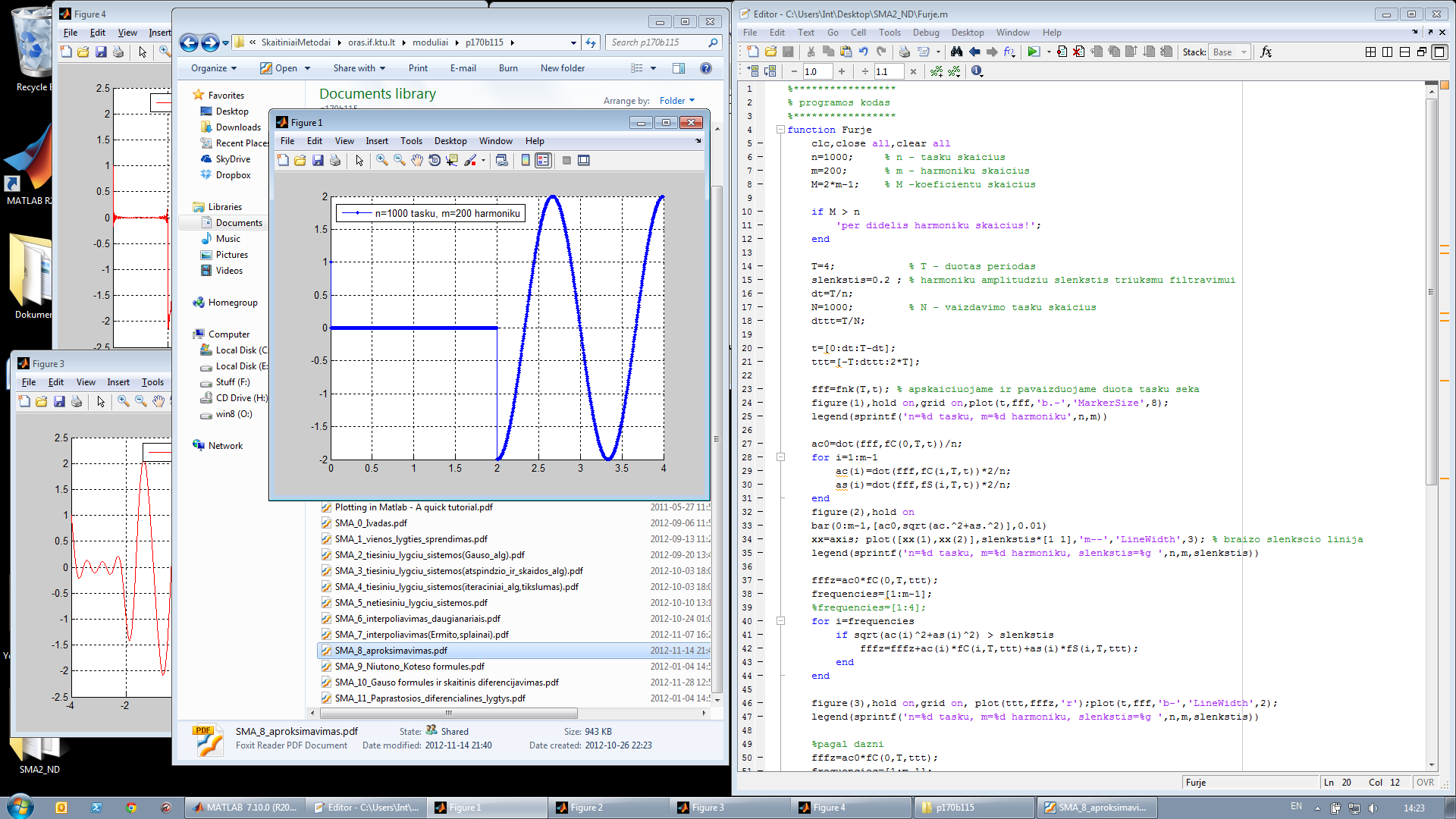
## Diskrečioji Furje aproksimacija

Duota:

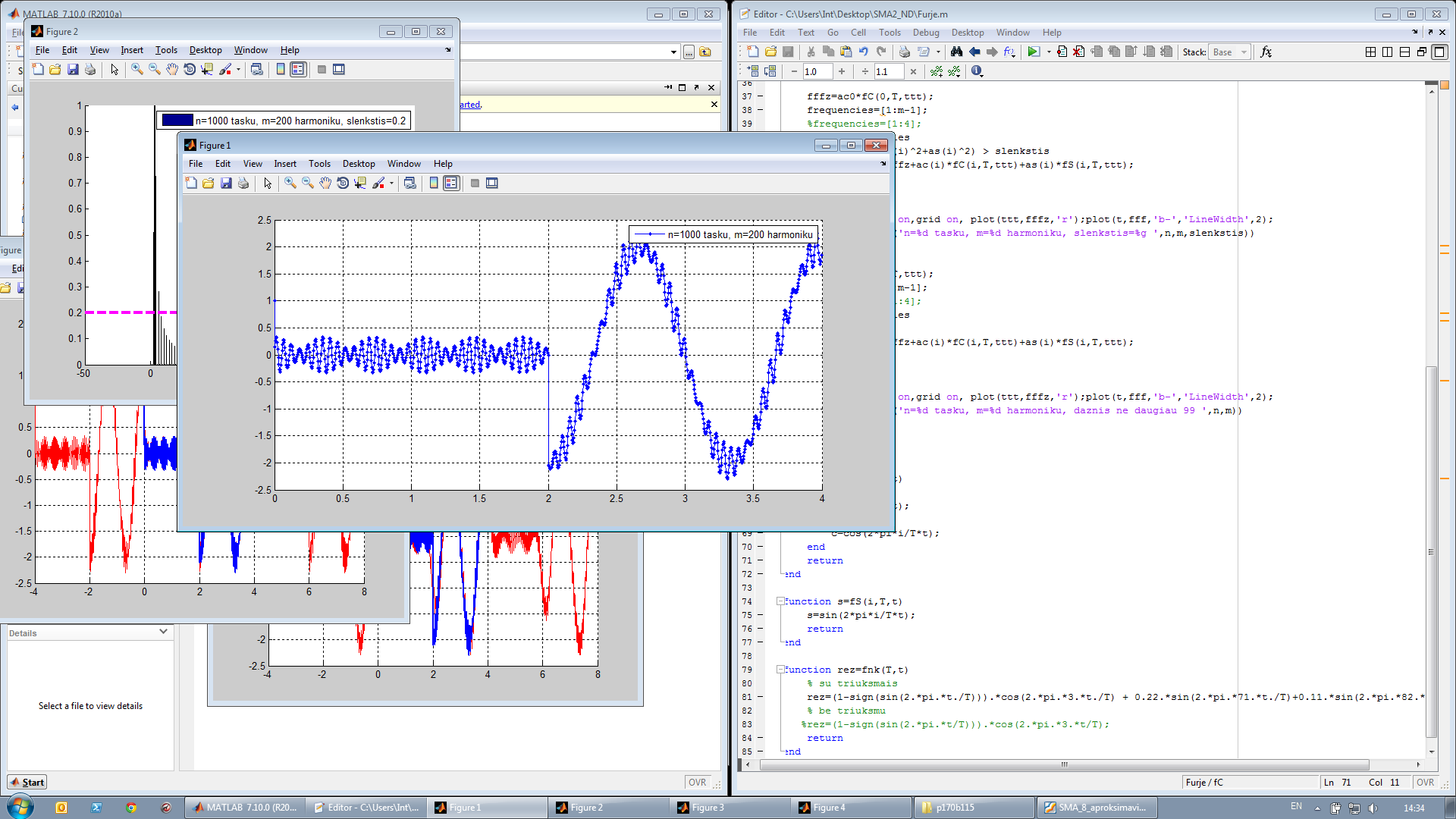
Taškų skaičius n=1000

Harmonikų skaičius m=200

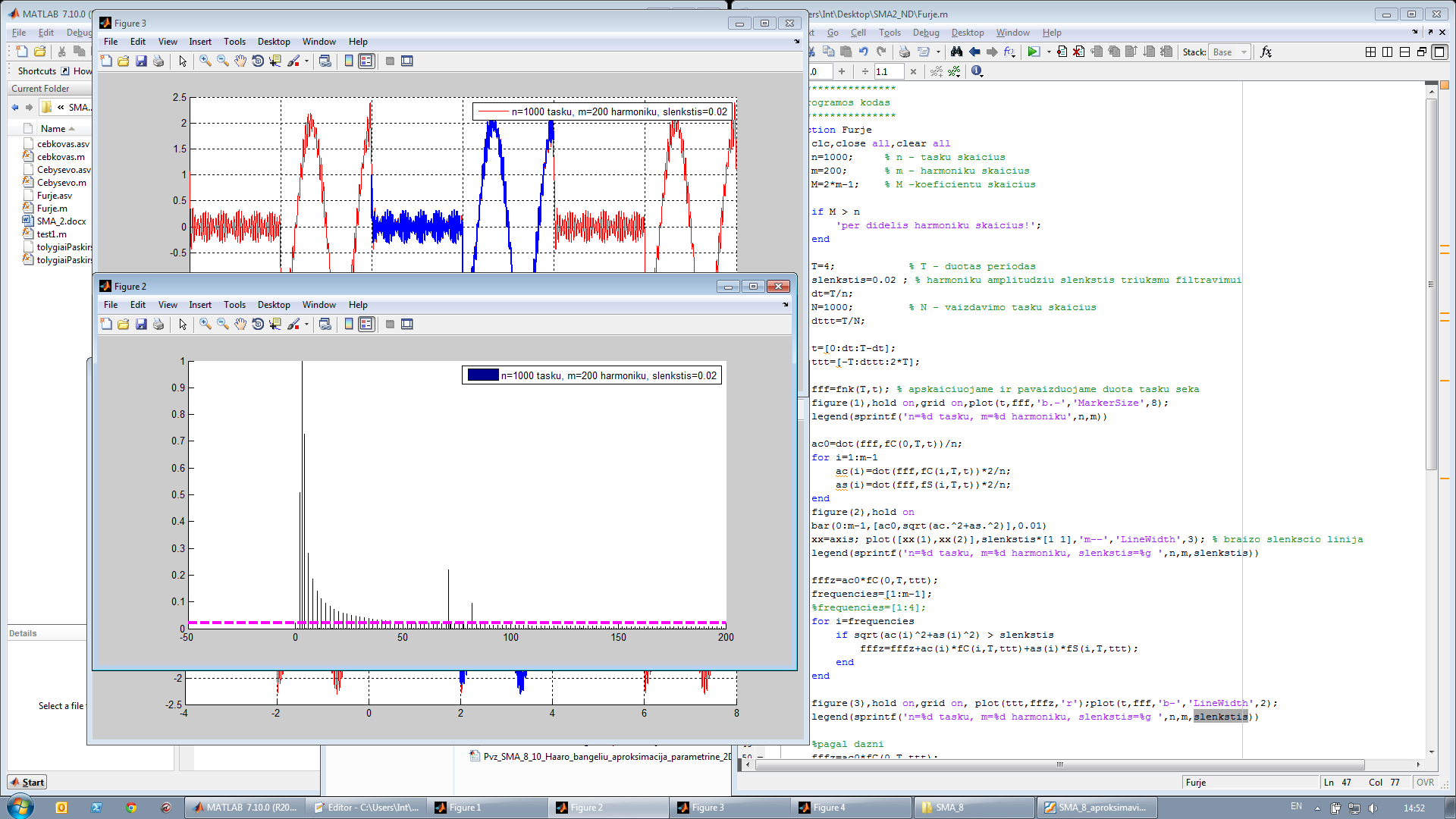
Periodas T=4



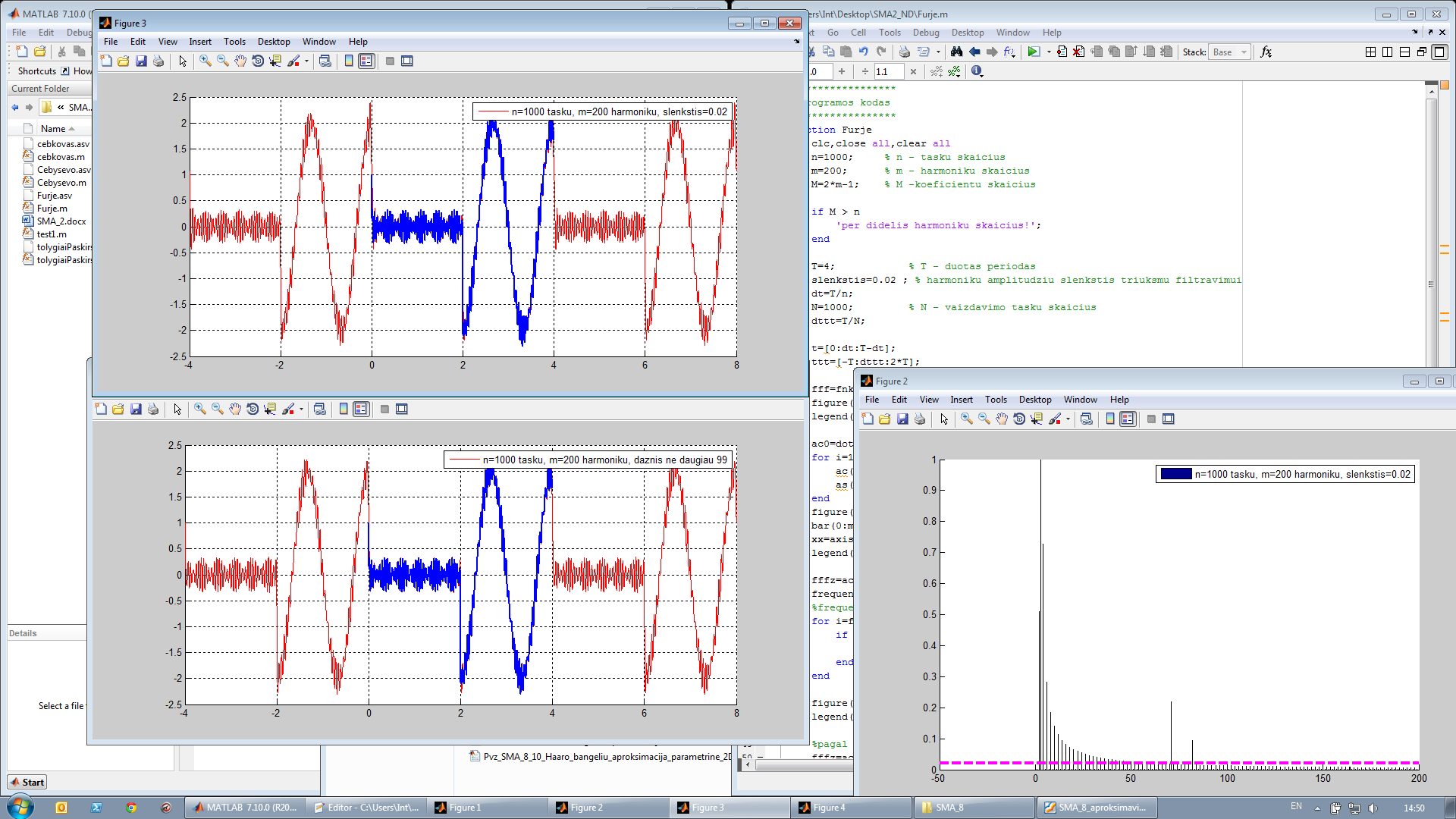
8. Funkcijos grafikas be triukšmo.



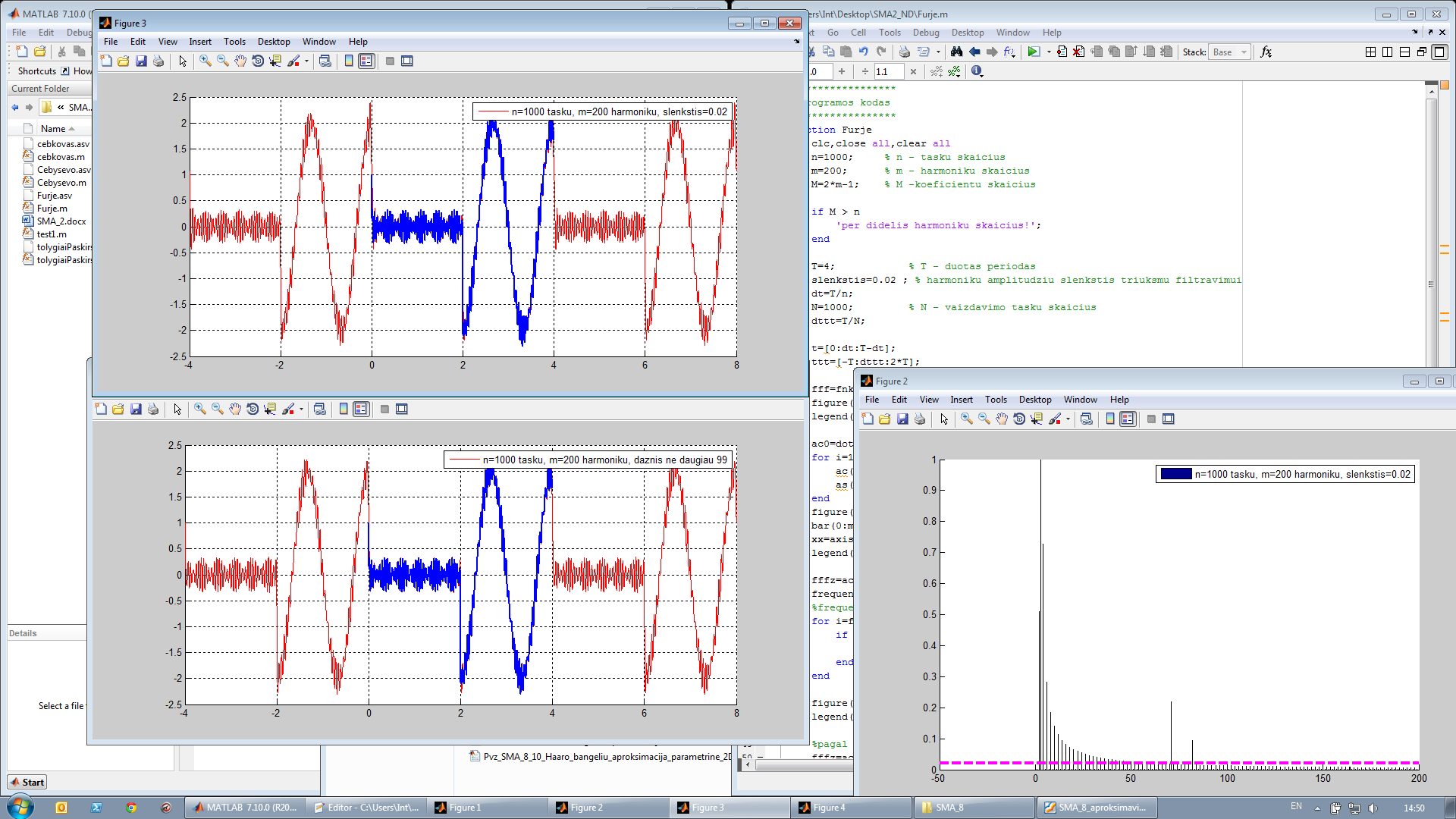
9. Funkcijos grafikas su triukšmu.



. Atliekame filtravimą, kurio metu atmetame harmonikas kurių amplitudės yra mažesnės nei nurodytas slenkstis.



. Harmonikų ir aproksimuotos funkcijos grafikas.

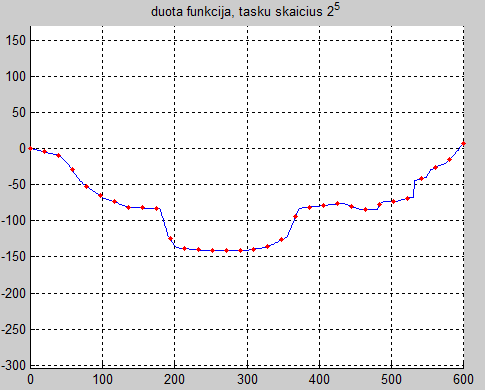
****

. Atlikto filtravimo kai buvo atmetamos harmonikos kurių dažnis didesnis nei 99 grafikas.

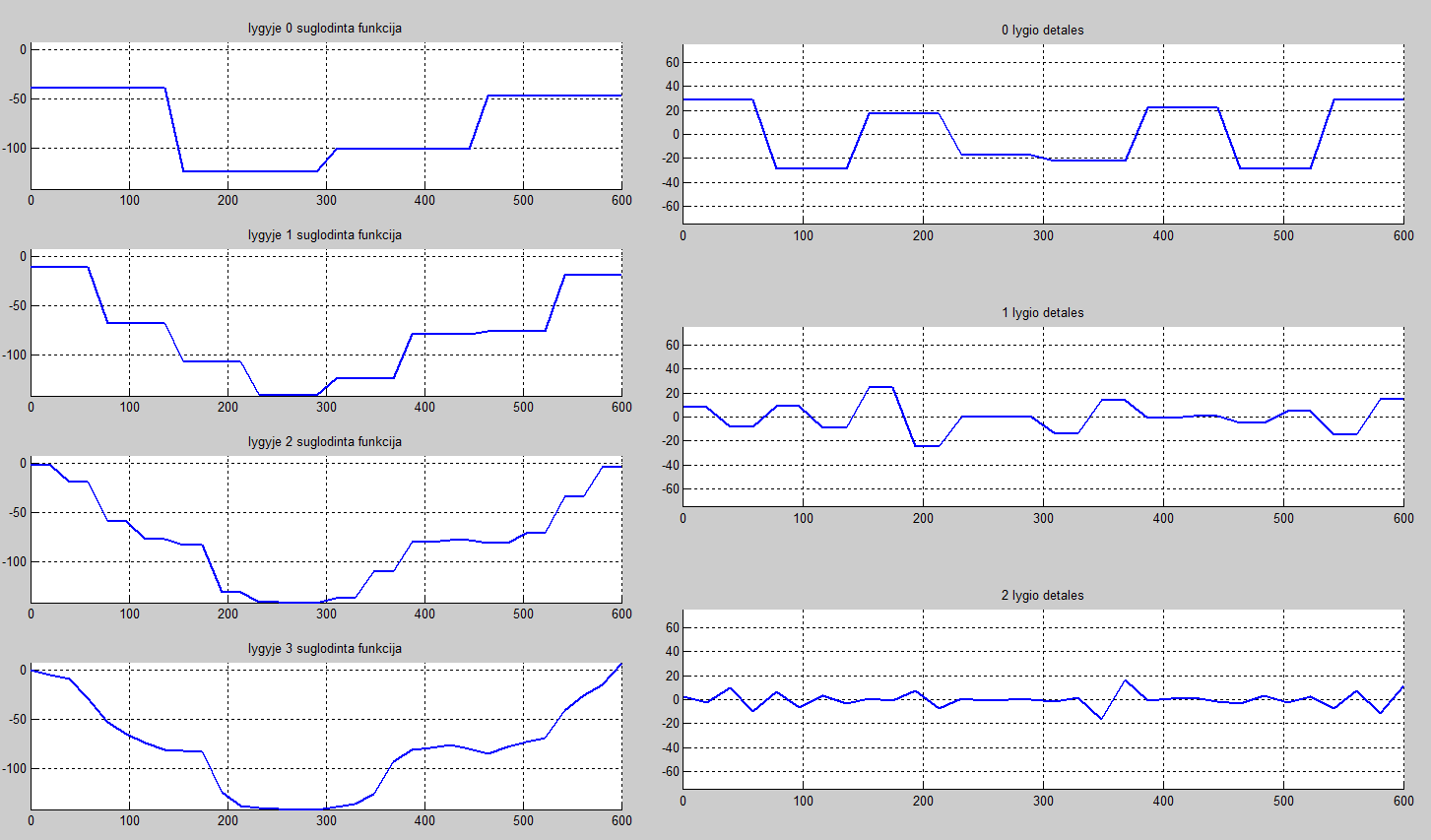
Programos kodas:

|  |
| --- |
| %\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  % programos kodas  %\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  function Furje  clc,close all,clear all  n=1000; % n - tasku skaicius  m=200; % m - harmoniku skaicius  M=2\*m-1; % M -koeficientu skaicius    if M > n  'per didelis harmoniku skaicius!';  end    T=4; % T - duotas periodas  slenkstis=0.02 ; % harmoniku amplitudziu slenkstis triuksmu filtravimui  dt=T/n;  N=1000; % N - vaizdavimo tasku skaicius  dttt=T/N;    t=[0:dt:T-dt];  ttt=[-T:dttt:2\*T];    fff=fnk(T,t); % apskaiciuojame ir pavaizduojame duota tasku seka  figure(1),hold on,grid on,plot(t,fff,'b.-','MarkerSize',8);  legend(sprintf('n=%d tasku, m=%d harmoniku',n,m))    ac0=dot(fff,fC(0,T,t))/n;  for i=1:m-1  ac(i)=dot(fff,fC(i,T,t))\*2/n;  as(i)=dot(fff,fS(i,T,t))\*2/n;  end  figure(2),hold on  bar(0:m-1,[ac0,sqrt(ac.^2+as.^2)],0.01)  xx=axis; plot([xx(1),xx(2)],slenkstis\*[1 1],'m--','LineWidth',3); % braizo slenkscio linija  legend(sprintf('n=%d tasku, m=%d harmoniku, slenkstis=%g ',n,m,slenkstis))    fffz=ac0\*fC(0,T,ttt);  frequencies=[1:m-1];  %frequencies=[1:4];  for i=frequencies  if sqrt(ac(i)^2+as(i)^2) > slenkstis  fffz=fffz+ac(i)\*fC(i,T,ttt)+as(i)\*fS(i,T,ttt);  end  end    figure(3),hold on,grid on, plot(ttt,fffz,'r');plot(t,fff,'b-','LineWidth',2);  legend(sprintf('n=%d tasku, m=%d harmoniku, slenkstis=%g ',n,m,slenkstis))    %pagal dazni  fffz=ac0\*fC(0,T,ttt);  frequencies=[1:m-1];  %frequencies=[1:4];  for i=frequencies  if i < 99  fffz=fffz+ac(i)\*fC(i,T,ttt)+as(i)\*fS(i,T,ttt);  end  end    figure(4),hold on,grid on, plot(ttt,fffz,'r');plot(t,fff,'b-','LineWidth',2);  legend(sprintf('n=%d tasku, m=%d harmoniku, daznis ne daugiau 99 ',n,m))    return  end    function c=fC(i,T,t)  if i==0  c=1\*cos(0\*t);  else  c=cos(2\*pi\*i/T\*t);  end  return  end    function s=fS(i,T,t)  s=sin(2\*pi\*i/T\*t);  return  end    function rez=fnk(T,t)  % su triuksmais  rez=(1-sign(sin(2.\*pi.\*t./T))).\*cos(2.\*pi.\*3.\*t./T) + 0.22.\*sin(2.\*pi.\*71.\*t./T)+0.11.\*sin(2.\*pi.\*82.\*t./T);  % be triuksmu  %rez=(1-sign(sin(2.\*pi.\*t/T))).\*cos(2.\*pi.\*3.\*t/T);  return  end |

## Haro bangelių aproksimacija



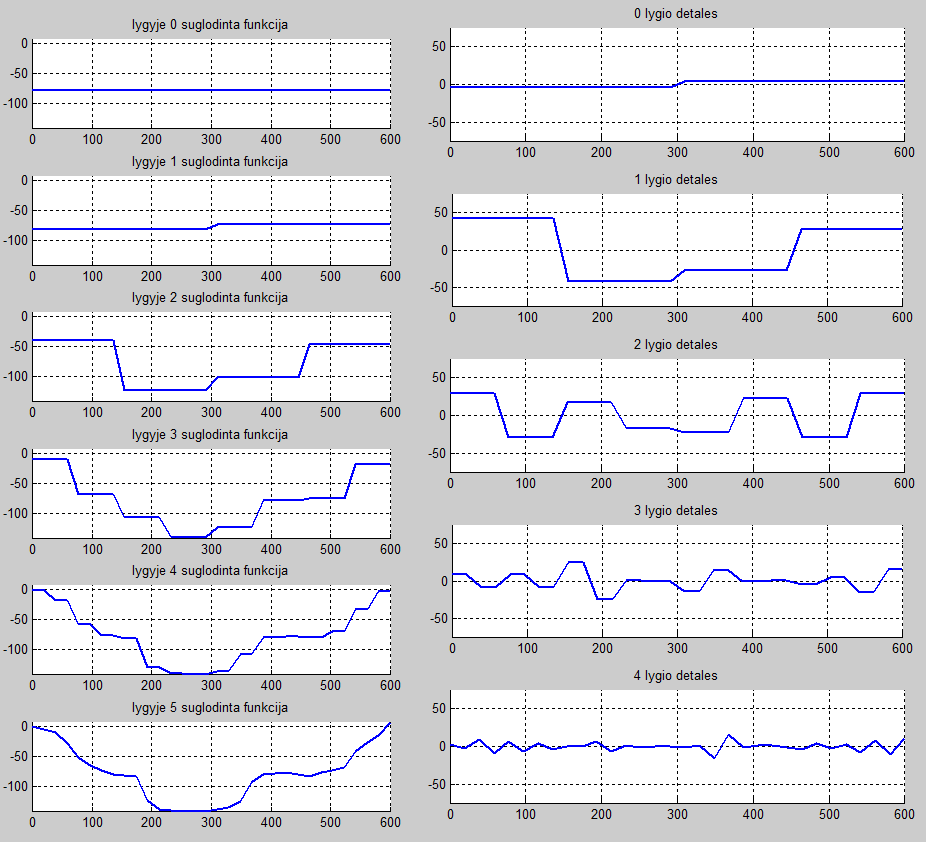
13. Duotos funkcijos grafikas.



.Suglodinta funkcija bei detalių lygiai, kai detalių lygiai yra 3.

Banglių koeficientai kai detalumo lygių skaičiu m=3:

|  |
| --- |
| 348.387 1454.83 955.065 524.081 86.3468 1054.77 103.065 0 -264.702 -2459.18 -169.694 278.758 -509.516 -327.218 -1154.18 -1668.97 |
| 1768.82 1952.3 5143.41 72.8776 -2954.64 -162.965 -1068.07 -3204.97 |
| 8639.43 5135.13 -6673.71 -8616.52 |



15. Suglodinta funkcija bei detalių lygiai, kai detalių lygiai yra 5.

Banglių koeficientai kai detalumo lygių skaičiu m=5:

|  |
| --- |
| 348.387 1454.83 955.065 524.081 86.3468 1054.77 103.065 0 -264.702 -2459.18 -169.694 278.758 -509.516 -327.218 -1154.18 -1668.97 |
| 1768.82 1952.3 5143.41 72.8776 -2954.64 -162.965 -1068.07 -3204.97 |
| 8639.43 5135.13 -6673.71 -8616.52 |
| 17978.3 -11450.5 |
| -2204.58 |

Programos kodas:

|  |
| --- |
| % Haro bangeliu aproksimacija    function main  clc;close all;clear all;  spalvos={'r-','g-','m-','c-','k-','y-','r.','g.','m.','c.','k.','y.'};      n=5  nnn=2^n;  fclose all;  fhx=fopen('x.txt','r');  fhy=fopen('y.txt','r');    figure(1); axis equal,hold on,grid on    SX=fscanf(fhx,'%g '); SY=fscanf(fhy,'%g ');    fclose all; plot(SX,SY);  pause  a=min(SX),b=max(SX),t=[a:(b-a)/(nnn-1):b];  ts=interp1(SX,SY,t);  clear SX SY, SX=t;SY=ts;plot(SX,SY,'r.');  title(sprintf('duota funkcija, tasku skaicius 2^%d',n));  xmin=min(SX);xmax=max(SX);  ymin=min(SY);ymax=max(SY);      % Aproksimavimas Haro bangelemis:  m=5 % detalumo lygiu skaicius  smooth=(b-a)\*SY\*2^(-n/2); % auksciausio detalumo suglodinimas (pagal duota funkcija)    for i=1:m  smooth1=(smooth(1:2:end)+smooth(2:2:end))/sqrt(2);  details{i}=(smooth(1:2:end)-smooth(2:2:end))/sqrt(2);  fprintf(1,'\n details %d : ',i);fprintf('%g ', details{i});  smooth=smooth1;  end  fprintf(1,'\n smooth %d : ',i);fprintf('%g ', smooth);fprintf('\n');  % Funkcijos rekonstrukcija:    h=zeros(1,nnn); for k=0:2^(n-m)-1, h=h+smooth(k+1)\*Haar\_scaling(SX,n-m,k,a,b); end % suglodinta funkcija  leg={sprintf('suglodinta funkcija, detalumo lygmuo %d',n-m)};  figure(2);subplot(m+1,1,1),axis equal,axis([xmin xmax ymin ymax]); hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d suglodinta funkcija',0));    for i=0:m-1 %detalumo didinimo ciklas  % apskaiciuojamos funkcijos detales:  h1=zeros(1,nnn); for k=0:2^(n-m+i)-1, h1=h1+details{m-i}(k+1)\*Haar\_wavelet(SX,n-m+i,k,a,b); end  figure(3),subplot(m,1,i+1), axis equal,hold on,grid on  yshift=(ymin+ymax)/2;axis([xmin xmax ymin-yshift ymax-yshift]), plot(SX,h1,'b-','Linewidth',2);title(sprintf('%d lygio detales',i));  leg={leg{1:end},sprintf('lygmens %d detales',n-m+i)};  h=h+h1; % detales pridedamos prie ankstesnio suglodinto vaizdo  figure(2);subplot(m+1,1,i+2),axis equal,axis([xmin xmax ymin ymax]), hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d suglodinta funkcija' ,i+1));  end    return  end  function h=Haar\_scaling(x,j,k,a,b) % \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  eps=1e-9;  xtld=(x-a)/(b-a); % (a,b) intervale duota kintamojo reiksme perskaiciuojama i "standartini"  % intervala (0,1), kuriame uzrasyta bangeles formule  xx=2^j\*xtld-k; h=2^(j/2)\*(sign(xx+eps)-sign(xx-1-eps))/(2\*(b-a));  return  end    function h=Haar\_wavelet(x,j,k,a,b) % \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  eps=1e-9;  xtld=(x-a)/(b-a); % (a,b) intervale duota kintamojo reiksme perskaiciuojama i "standartini"  % intervala (0,1), kuriame uzrasyta bangeles formule  xx=2^j\*xtld-k; h=2^(j/2)\*(sign(xx+eps)-2\*sign(xx-0.5)+sign(xx-1-eps))/(2\*(b-a));  return  end |