ОБ ОСВОЕНИИ ПАКЕТА MATLAB С ИСПОЛЬЗОВАНИЕМ ПРАКТИЧЕСКИХ ЗАДАЧ[1]

Кручинин П.А., Московский государственный университет им. М.В.Ломоносова, kruch@mech.math.msu.su

На механико-математическом факультете МГУ для студентов механиков младших курсов преподавателями кафедры прикладной механики и управления читается факультативный спецкурс «Компьютерный анализ механических систем». Целями этого курса являются – ознакомление студентов с задачами, решаемыми на кафедре и привлечение их интереса к интегрированным пакетам, используемым для научных исследований.

Одним из таких пакетов является MATLAB. Для знакомства с этим пакетом отведен семестровый курс лекционно-практических занятий с зачетом. На этих занятиях студенты знакомятся с основами MATLAB'а, решая практические задачи на компьютере. Особенность чтения подобного курса на мех-мате МГУ заключается в том, что слушатели хорошо освоили основы фундаментальных математических дисциплин, таких как математический анализ, линейная алгебра, аналитическая геометрия и др., неплохо представляют основы программирования. Однако знакомство с основными курсами прикладных дисциплин, таких как теория дифференциальных уравнений, методы вычислений, теория вероятности, теоретическая механика для них только начинается. Это обстоятельство предъявляет дополнительные требования к простоте изложения содержательных механических задач.

Первые три лекции курса посвящены знакомству студентов с основными командами и функциями пакета MATLAB. Дальнейшие занятия проводятся по следующему сценарию:

- Излагается новые для слушателей обращения к функциям MATLAB'а.
- Формулируется содержание несложной механической задачи и алгоритм ее решения.
- Слушателям предлагается написать программу решающую эту задачу. Перечислим предлагаемые задачи и разделы MATLAB, используемые при их решении .

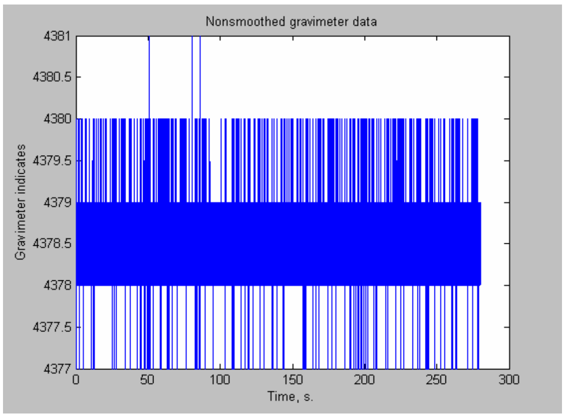

Рис. 1. Показания гравиметра.

[1] В работе нашли отражение исследования выполняемые по гранту программы Университеты России УР.04.03.10.

1. ***Технологии ядерного сглаживания дискретной последовательности случайных чисел на примере показаний гравиметра***. Эта задача используется для закрепления пройденного ранее материала: написание функций и циклов, использование поэлементных векторных операций, функций *eval* и *feval.*

Слушателям предоставляется файл показаний гравиметрической аппаратуры и предлагается написать функцию ядерного сглаживания этих показаний. Такая процедура является важной частью алгоритмов авиационной гравиметрии [1]. В связи с наличием ошибок дискретизации, температурных помех и прочих возмущений соотношение сигнал-шум в этой задаче может достигать значений порядка 10^3. Характерный вид показаний гравиметра приведен на рис. 1. Одна из эффективных процедур фильтрации высокочастотных шумов в таком сигнале связана с использованием процедур ядерного сглаживания [2]. В соответствии с этим методом новое сглаженное значение показаний для момента времени t вычисляется как среднее арифметическое $2N$ взвешенных значений взятых на интервале времени $[t-N*\Delta t,\ t-N*\Delta t]$ (Δt – величина такта съёма измерений). Выбор весовых коэффициентов можно сделать с использованием нескольких функций из *TOOLBOX*'а *ident* пакета MATLAB. Программа осуществляющая эти вычисления может иметь вид:

```
function [s] = winsms(h,nwin,winname)
% Window smoothing
%
% [s]=winsms(h,nwin,winname)
%
% h – матрица данных,
%       подлежащих сглаживанию
% nwin – ширина окна.
% winname – имя функции расчета
%             весовых коэффициентов
%
% s – сглаженные данные

if nargin < 3,
   winname='hanning';
end

wh=feval(winname,nwin);
wh=wh/sum(wh);
[N,k]=size(h);
s=zeros(N-nwin+1,k);
for j=1:k,
   for i=1:(N-nwin+1),
      s(i,j)= sum(h(i:(i+nwin-1),j).*wh);
```

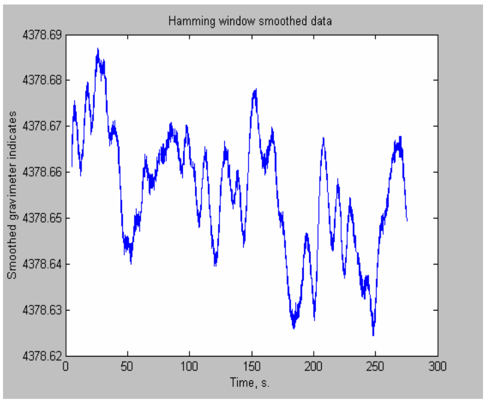

Рис. 2. Результат сглаживания показаний гравиметра окном Хамминга

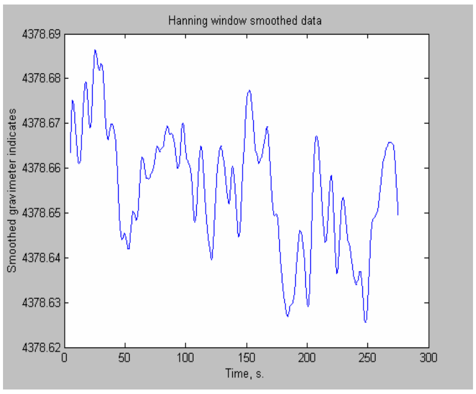

Рис. 3. Результат сглаживания показаний гравиметра окном фон Ханна.

```
    end
end
```

Результат вычислений для окон Хамминга (функция *hamming*) и фон Ханна (функция *hanning*) приведены на рис.2 и 3.

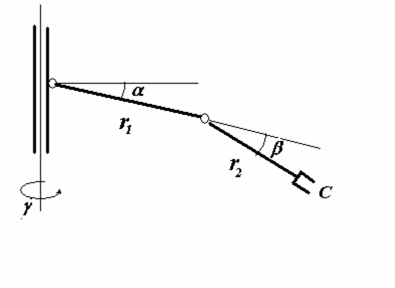

Рис. 4. Схема двузвенного манипулятора.

2. ***Построение рабочей области двузвенного манипулятора***. Эта задача используется при освоении функций двумерной и трехмерной графики.

Слушателям предлагается построить область точек пространства до которых может дотягиваться захват *C* двузвенного манипулятора, изображенного на рис. 4, при заданных значениях длин звеньев и ограничениях на углы поворота в сочлененениях $-\alpha_{max}\le\alpha\le\alpha_{max}$, $-\beta_{max}\le\beta\le\beta_{max}$, $-\gamma_{max}\le\gamma\le\gamma_{max}$. Задача такого типа является традиционной при построении и исследовании манипуляторов [3] и шагающих аппаратов [4].

Задачу предлагается решать в два этапа: на первом этапе построить непрерывные векторы *X* и *Y*, содержащие координаты границы рабочей области плоского манипулятора при $\gamma=0$, а на втором использовать эту границу для построения пространственной поверхности.

Алгоритм построения границы плоского сечения рабочей области сводится к кусочному построению границы: сначала рисуется часть границы для которой α меняется на интервале $[-\alpha_{max},\alpha_{max}]$, а $\beta=0$. далее $\alpha=\alpha_{max}$, $\beta\in[0, \beta_{max}]$; $\alpha\in[-\alpha_{max},\alpha_{max}]$, а $\beta=\beta_{max}$ и так далее до замыкания кривой. Функция решающая эту задачу имеет вид:

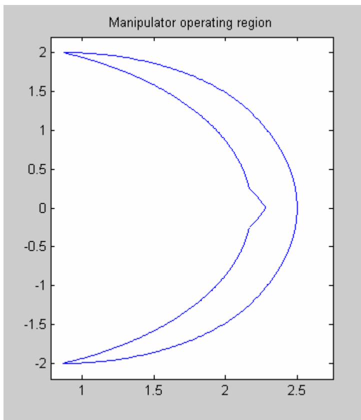

Рис.5. Рабочая область плоского манипулятора при $r_1=1$, $r_2=1.5$, $\alpha_{max}=\pi/6$, $\beta_{max}=\pi/3$.

```
function [X,Y]=achie(r1, r2, Amax, Bmax)

% r1,r2 – длины звеньев
%  Amax – максимальное значения угла alpha
%  Bmax –  максимальное значения угла beta
%
% X,Y – координаты границы рабочей области

n=100;
Ap=linspace(0,Amax,n);
Bp=linspace(0,Bmax,n);
```

```
Am=fliplr(Ap);
Bm=fliplr(Bp);


X=r1*cos([-Am,Ap])+r2*cos([-Am,Ap]+Bmax);
Y=r1*sin([-Am,Ap])+r2*sin([-Am,Ap]+Bmax);
X(2*n+1:3*n)=r1*cos(Amax)+r2*cos(Amax+Bm);
Y(2*n+1:3*n)=r1*sin(Amax)+r2*sin(Amax+Bm);
X(3*n+1:4*n)=(r1+r2)*cos(Am);
Y(3*n+1:4*n)=(r1+r2)*sin(Am);
X(4*n+1:5*n)=(r1+r2)*cos(-Ap);
Y(4*n+1:5*n)=(r1+r2)*sin(-Ap);
X(5*n+1:6*n)=r1*cos(-Amax)+r2*cos(-Amax-Bp);
Y(5*n+1:6*n)=r1*sin(-Amax)+r2*sin(-Amax-Bp);
X(6*n+1:8*n)=r1*cos([-Am,Ap])+r2*cos([-Am,Ap]-Bmax);
Y(6*n+1:8*n)=r1*sin([-Am,Ap])+r2*sin([-Am,Ap]-Bmax);

if r1*sin(Amax)+r2*sin(Amax-Bmax) < r1*sin(-Amax)+r2*sin(-Amax+Bmax),
        Bk=Amax+asin(r1/r2*sin(Amax));
        Bp=linspace(Bk,Bmax,n);
        Bm=fliplr(Bp);
        X(8*n+1:9*n)=r1*cos(Amax)+r2*cos(Amax-Bm);
        Y(8*n+1:9*n)=r1*sin(Amax)+r2*sin(Amax-Bm);
        X(9*n+1:10*n)=r1*cos(-Amax)+r2*cos(-Amax+Bp);
        Y(9*n+1:10*n)=r1*sin(-Amax)+r2*sin(-Amax+Bp);
end

plot(X,Y)
h=gca;
xmin=min(X)-0.1*abs(min(X));
ymax=max(Y)*1.1;
set(h,'Xlim',[xmin,(r1+r2)*1.1],'Ylim',[-ymax,ymax]);
```

Границу пространственной рабочей области манипулятора можно представить как часть поверхности образованной в результате вращения кривой, полученной при решении предыдущей задачи. Она представлена на рис. 6 и получена в результате выполнения следующих команд

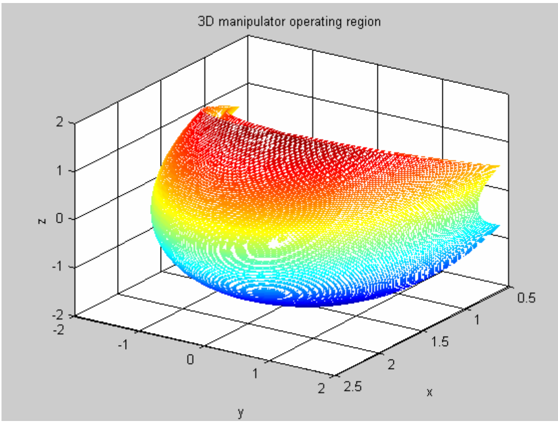

Рис. 6. Рабочая область пространственного манипулятора при $\gamma_{max}=\pi/4$.

```
G=linspace(-Gmax,Gmax,100);
Xp=X'*cos(G);
Yp=Y'*ones(size(G));
Zp=X'*sin(G);
mesh(Xp,Yp,Zp)
view([3,-1,-1])
```

3. ***Моделирование плоского движения тяжелой материальной точки, брошенной в круглую трубу (случай абсолютно упругого удара).*** Решение этой задачи позволяет закрепить пройденный материал по работе с визуализацией результатов и используется для освоения процедуры отыскания корней полинома и простейшего ввода данных при выполнении функции. Одновременно слушатели знакомятся с классом детерминированных динамических систем, чье поведение может носить хаотический характер.

Слушателям предлагается построить траектории движения шарика (материальной точки), брошенного в горизонтально расположенную круглую трубу в поле вертикальной силы тяжести [5]. Для упрощения задачи будем считать массу точки и радиус трубы единичными, а удельную силу тяжести полагаем равной $\vec{a} = [0;-2]^T$.

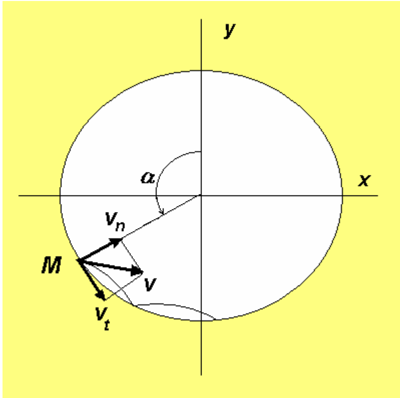

Рис. 7. Описание параметров состояния при отскоке шарика в трубе.

После отскока шарика от стенки трубы его положение опишем углом α, образованным вертикалью и радиусом трубы, проведенным в точку столкновения, как показано на рис. 7. Зададим начальные условия в момент отскока шарика от стенки трубы. Начальную скорость шарика до и после соударения опишем в виде суммы двух составляющих: v_t - касательной к поверхности трубы и v_n – нормальной к этой поверхности. Радиус вектор и скорость точки M, записанные в проекциях на оси неподвижной декартовой системы координат имеют вид

$$\vec{r}_+ = -\begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix}; \qquad \vec{v}_+ = \begin{bmatrix} v_{n+}\sin\alpha - v_{t+}\cos\alpha \\ -v_{n+}\cos\alpha - v_{t+}\sin\alpha \end{bmatrix}$$

Будем значениям скоростей до соударения приписывать индекс ‘-’ и индекс ‘+’ составляющим скорости после соударения. Траекторию движеия шарика после соударения запишем в виде

$$\vec{r}(t) = \vec{r}_+ + \vec{v}_+ t + \vec{a}\frac{t^2}{2}, \qquad \vec{v}(t) = \vec{v}_+ + \vec{a}t.$$

Момент следующего соударения шарика со стенкой трубы найдем из условия $||x||^2=1$. Это условие можно преобразовать к уравнению 3-ей степени

$$t^3 - 2v_{y+}t^2 + (2y_+ + \|\vec{v}\|^2)t + 2(x_+v_{x+} + y_+v_{y+}) = 0$$

Здесь x_+ и y_+ -- компоненты радиус вектора $\vec{r}(t)$, а v_{x+} и v_{y+} проекции вектора $\vec{v}_+$ на оси x и y. Наименьший положительный действительный корень этого уравнения определяет момент времени τ следующего соударения , которое происходит в точке $\vec{r}_- = \vec{r}(\tau)$ (α=-atan2($y_-(\tau)$,$x_-(\tau)$)) со скоростью $\vec{v}_- = \vec{v}(\tau)$. Условие изменения скорости в результате абсолютно упругого удара запишем в виде

$$\vec{v}_+ = \vec{v}_- - 2(\vec{v}_-, \vec{r}_-)\vec{r}_-$$

Программа, реализующая моделирование в соответствии с представленными вычислениями имеет вид.

```
function [x,y]=billiard(ax,vt,vr,n)

plot(sin(0:(pi/360):(2*pi)),cos(0:(pi/360):(2*pi)),'k')
hold on
acc=[0,-1];
if nargin==3,
  n=100;
elseif nargin < 3,
  error('Не хватает аргументов функции.')
  return
end

 xp=[cos(ax),-sin(ax)];
 vp=[vr*cos(ax)-vt*sin(ax), -vr*sin(ax)+vt*cos(ax)];

a=zeros(n,1);
a1=zeros(n,1);
x=zeros(100*n,1);
y=zeros(100*n,1);

for i=1:n,
    c=[1,-4*vp(2),4*(vp(1)^2+vp(2)^2
-xp(2)),8*(xp(1)*vp(1) +xp(2) *vp(2))];
    r=roots(c);
    r=sort(r);
    tau = 0;
     for j=1:3,
       if imag(r(j)) == 0 & real(r(j))
             > 0,
         tau = real(r(j));
         break;
       end
     end
```

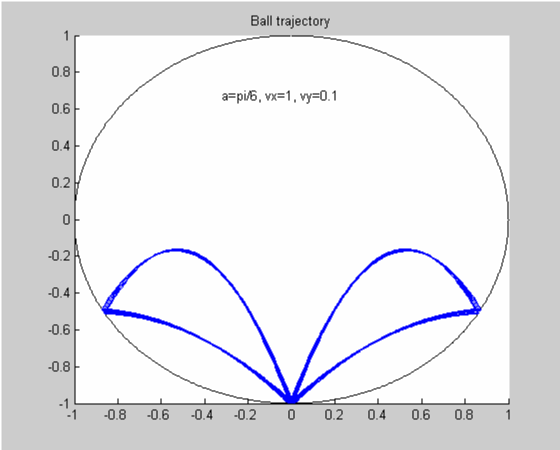

Рис. 8. Траектория движения шарика в трубе при начальных условиях $\alpha=\pi/6$, v_n=0.1, v_t=1.

```
if tau == 0,
  i
 error(' Polynomial have not     any real positive roots.');
  break;
end
t=(tau/100):(tau/100):tau;
xt=ones(100,1)*xp+t'*vp+(t').^2*acc/2;
x(((i-1)*100+1):(i*100))=xt(:,1);
y(((i-1)*100+1):(i*100))=xt(:,2);
vm=vp+acc*tau;
xm=xp+vp*tau+acc*tau^2/2;
if abs(norm(xm)-1) > 1.e-6,
  i
  disp(tau)
 error(' Model point is not on
 circle.');
  break;
end

xp=xm;
vp=vm-2*(vm*xm')*xm;
end
plot(x,y)
hold off
```

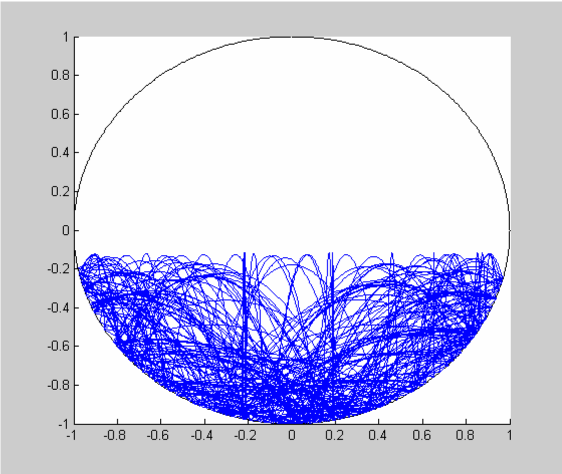

Рис. 9. Траектория движения шарика в трубе при начальных условиях $\alpha=\pi/6$, $v_n=0.15$, $v_t=1$, $n=300$.

Для вызова этой функции целесообразно создать файл сценария

```
Names={'Alpha','vn',’vt’,'Steps'};
Title='Initial data ';
LineNo=4;
DefAns={'pi/6','1',’0.1’,'100'};
while str2num(DefAns{4})>0,
      DefAns=inputdlg(Names,Title,LineNo,DefAns);
                      [x,y]=billiard(str2num(DefAns{1}), str2num(DefAns{2}),
            str2num(DefAns{3}), str2num(DefAns{4}));
end
```

На рис. 8 и 9 показаны результаты вычислений этой программы. Начальные условия решения, приведенного на рис. 8 соответствуют малой окрестности устойчивой циклической траектории. На рис. 9 приведена траектория соответствующая хаотическому поведению. При увеличении числа отскоков *n* траектории заполняют весь фазовый объём, определяемый уровнем энергии в системе.

4. ***Плоские модельные задачи спутниковой навигации.*** Используются при освоении функций решения нелинейных уравнений.

Слушателям предлагается решить ряд простых модельных задач спутниковой навигации. Задача определения координат приемника в результате обработки сигналов спутниковой навигационной системы является одной из наиболее сложных задач, требующих глубокого знания специальных математических методов обработки информации [6], [7].

Будем предполагать, что приемник спутниковой навигационной системы принимает одновременно сигналы нескольких спутников. Каждый сигнал, посланный спутником с номером k, содержит информацию о радиус-векторе $\vec{r}_k$ спутника S_k (точнее о его декартовых координатах x_k, y_k, z_k) и точном времени t_k отправления этого сигнала. Координаты радиус-вектора $\vec{r}$ приемника M обозначим через x, y, z соответственно, а через t обозначим время получения сигналов приемником.

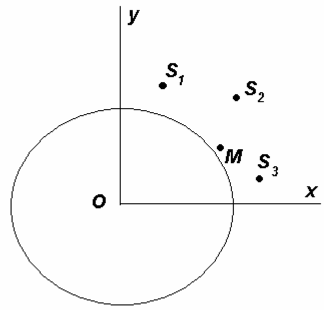

Рис. 10. Система координат в плоской модельной задаче спутниковой навигации.

Для плоской задачи определить координаты приемника можно в результате решения триангуляционной задачи. Для этого необходимо решить систему из двух уравнений

$$\|\vec{r} - \vec{r}_1\| = c(t - t_1) \qquad \|\vec{r} - \vec{r}_2\| = c(t - t_2)$$

Здесь c скорость распространения радиоволн, которую считаем постоянной.

Введем обозначение для дальностией $l_1=c(t-t_1)$ и $l_2=c(t-t_2)$ и запишем эту систему в координатном виде

$$(x - x_1)^2 + (y - y_1)^2 = l_1^2; \qquad (x - x_2)^2 + (y - y_2)^2 = l_2^2; \qquad (4.1)$$

Использование переменной l_k позволяет отказаться от требования одновременности получения спутниковых сигналов для приемника, неподвижного в системе координат Oxy. Выразим из второго уравнения неизвестную y

$$y = y_2 - \sqrt{l_2^2 - (x - x_2)^2} \qquad (4.2)$$

Решение первого уравнения относительно x с помощью функции *fzero* используем для отыскания координат приемника

X=fzero('gps1',xo,[],[],l1,l2,r1,r2)

Функция *gps1* имеет вид

```
function f=gps1(x,l1,l2,r1,r2)
% f=gps1(x,l1,l2,r1,r2)
% Рекомендуемое обращение  x=fzero('gps1',3,[],[], 2.5, 2,[1,7],[5,5])
% Для приведенных цифр 1соответствует 1000 километров

 y=r2(2)-sqrt(l2^2 - (x – r2(1))^2);
```

f=norm(r1-[x,y])-l1;

Недостатком этого метода вычислений является необходимость задания такого начального приближения *x0,* при котором выражение под знаком радикала в формуле (4.2) неотрицательно.

Более эффективным для решения поставленной задачи представляется использование процедуры *fsolve.* В этом случае решается система нелинейных уравнений (4.1) относительно переменных *x* и *y*. Эту задачу решает обращение

r=fsolve('gps2',r0,[],[],l1,l2,r1,r2);

вызывающая функцию

```
function f=gps2(r,l1,l2,r1,r2)
% f=gps2(r,l1,l2,r1,r2)
% Рекомендуемое обращение r=fsolve('gps2',[3,3],[],[], 2.5, 2,[1,7],[5,5])
% Для приведенных цифр 1 соответствует 1000 километров
 f(1)=norm(r-r1)-l1;
 f(2)=norm(r-r2)-l2;
```

Для предлагаемых в комментариях к функциям числовых значений начальных условий и первым и вторым способом отыскивается решение $\vec{r} = [3.1139,\ \ 5.6653]$. Между тем предлагаемая система уравнений имеет второе, решение $\vec{r}_* = [3.3361,\ \ 6.1098]$, ложное с точки зрения решения навигационной задачи. Это решение может быть получено, например, при начальном значении *r0=[3.3,7].* Отбросить это решение как ошибочное позволяет информация о том, что оно соответствует точке, находящейся на высоте ~500км над поверхностью Земли. Для исключения этого значения можно также использовать сигнал, полученный от третьего спутника.

В завершение рассмотрим еще одну задачу этого цикла. Учтем наличие ошибки синхронизации часов в приемнике. Полагаем, что все часы на спутниках синхронны (это достигается систематической коррекцией их), а часы на приемнике имеют систематическую ошибку Δt. В этом случае приемник определяет не истинную дальность до *к*-го спутника, а величину именуемую, псевдодальность $l_k^* = l_k + e$. Здесь $e = c\Delta t$. Для получения координат приемника в этом случае также потребуется третий спутник. Систему уравнений, для получения координат составим из трех уравнений

$$\|\vec{r} - \vec{r}_1\| = l_1^* - e \qquad \|\vec{r} - \vec{r}_2\| = l_2^* - e \qquad \|\vec{r} - \vec{r}_3\| = l_3^* - e$$

Рассмотрим две попарные разности этих уравнений

$$\|\vec{r} - \vec{r}_2\| - \|\vec{r} - \vec{r}_1\| = l_2^* - l_1^* \qquad \|\vec{r} - \vec{r}_3\| - \|\vec{r} - \vec{r}_1\| = l_3^* - l_1^*$$

В эти уравнения уже не входят слагаемые связанные с ошибками синхронизации часов. Таким образом координаты приемника можно определять в результате вызова функции *fsolve*

r=fsolve('gps3',r0,[],[],l1,l2,l3,r1,r2,r3);

Эта команда вызывает функцию

```
function f=gps3(r,l1,l2,l3,r1,r2,r3)
%  f=gps3(r,l1,l2,l3,r1,r2,r3)
% Рекомендуемое обращение
%                    r=fsolve('gps2',[3,3],[],[], 2.5, 2,3.92,[1,7],[5,5],[6,3])
%  Для приведенных цифр 1 соответствует 1000 километров
f(1)=norm(r2-r)-l2-norm(r1-r)+l1;
 f(2)=norm(r3-r)-l3-norm(r1-r)+l1;
```

Приведенные задачи легко переформулируются для трехмерного случая. Такое задание можно дать слушателям для самостоятельной работы.

5. Моделирование процессов вставания и приседания антропоморфного многозвенника. Используются при освоениии различных видов визуализации исследований.

Слушателям предлагается написать функцию, реализующую анимацию вставания и приседания человека. Численные значения координат различных точек на теле человека сведены предварительно в виде таблицы в файл, каждая строка которого соответствует фиксированному моменту времени. Интервал времени, прошедший между записью двух последовательных строк постоянен. Заданы номера столбцов, в которых помещены координаты точек на плече, корпусе, в районе тазобедренного сустава, на колене, на щиколотке и на пальцах стопы. Предлагается проектировать изображение на саггитальную плоскость (в профиль). Можно воспользоваться записями положений этих точек, полученными в результате эксперимента [8] либо численного моделирования типа [9]. В последнем случае анимация может оказаться полезной при оценке медиками правдоподобности математической модели.

Анимация может быть реализована одним из двух способов: Первый представляет собой канонический способ, использующий функции *movie*. На первом этапе, при таком подходе производится накопление фреймов изображения в массив, а на втором прокрутка анимации.

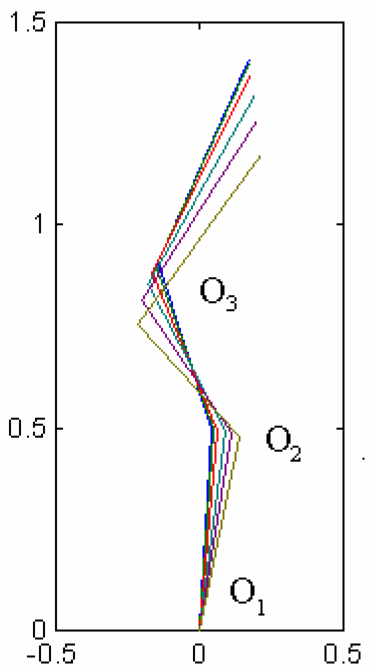

Рис. 11. Последовательность поз, отображаемая при мультипликации

Сценарий, реализующий представленную последовательность операций представлен ниже

% чтение данных из файла

```
load adc.txt
%Задание столбцов с координатами точек
s=[23, 20, 17, 14, 11, 8];
e=[25, 22, 19, 16, 13, 10];
hold on
h=gca;
%Инициализация массива фреймов
M=moviein(floor(length(adc)/30));
% Накопление фреймов изображения
for k=1:30:length(adc),
  cla
  x=[adc(k,s(1)),adc(k,s(2)),adc(k,s(3)),adc(k,s(4)),adc(k,s(5)),adc(k,s(6))];
  y=[adc(k,s(1)+1),adc(k,s(2)+1),adc(k,s(3)+1),adc(k,s(4)+1),adc(k,s(5)+1),
                 adc(k,s(6)+1)];
  plot(x,y)
  set(h,'XLim',[-600,0],'YLim',[0,1500])
  M(:,floor(k/30)+1) = getframe;
end
size(M)
cla
% Воспроизведение мультипликации
movie(M,1,1);
```

Простые изображения, отражающие получаемые непосредственно в процессе компьютерного моделирования, можно получить перерисовывая изображение на стандартном сером фоне графического окна пакета *MATLAB*. Пример функции реализующей анимацию такого типа представлен в виде *GUI,* приведенном на рис. 12. При нажатии в этом окне кнопки “Показать анимацию” вызывается последовательность команд

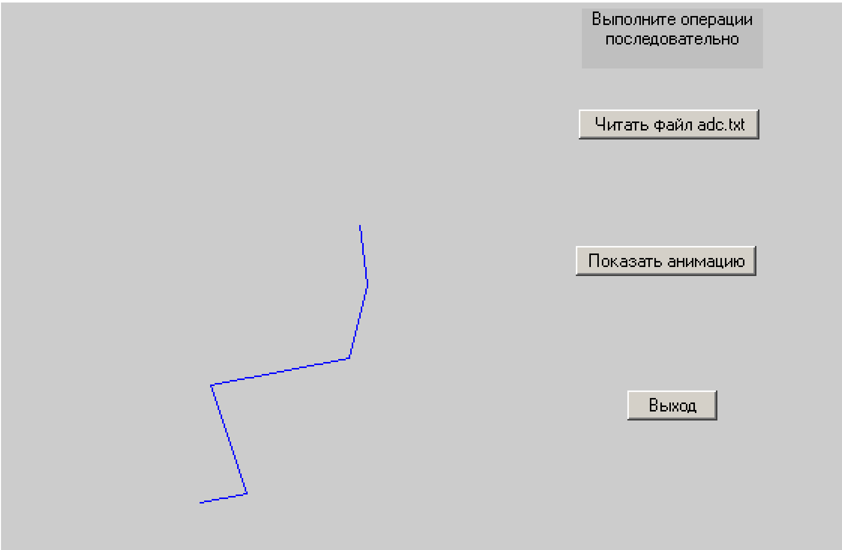

Рис. 12. Вид графического окна моделирования

```
x=adc(10,s)/100;
y=adc(10,s+1)/100;
hp=plot(x,y);
set(gca,'XLim',[-9,3],'YLim',[0,16],'Visible','off','LineWidth',1);
for i=2:N,
  x=adc(i*10,s)/100;
  y=adc(i*10,s+1)/100;
% Вместо двух предшествующих команд могут вызываться функции
% вычисления координат точек в рамках специализированной
%  математической модели
  set(hp,'XData',x,'YData',y);
  pause(0.1)
end
```

6. Моделирование собственных и вынужденных колебаний цепочки упруго соединенных звеньев, совершающих поступательное движение. Используется при освоении матричных операций, вычислении собственных чисел и собственных векторов, решении задачи Коши для однородной и неоднородной линейной системы уравнений.

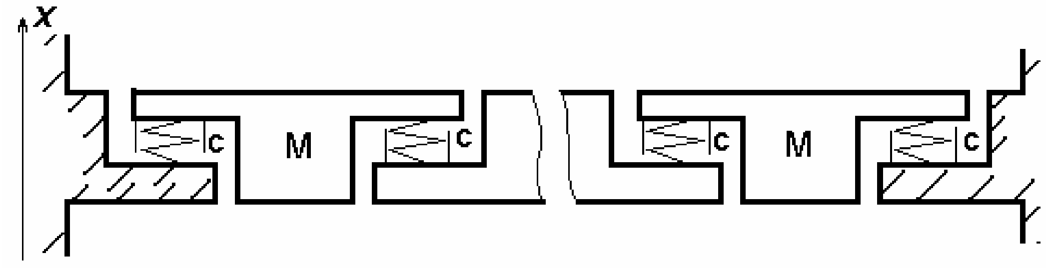

Рис. 13. цепочка упруго соединенных звеньев.

Слушателям предлагается исследовать собственные колебания цепочки звеньев, изображенной на рис. 13. Конструкция расположена в горизонтальной плоскости. Каждое звено этой цепочки имеет массу M и может совершать только поступательные движения вдоль оси x. Звенья соединены между собой упругими элементами жесткости c каждый. Требуется найти собственные частоты и собственные формы колебаний конструкции и промоделировать численно ее собственные колебания для случая, когда начальные условия соответствуют одной из собственных форм. Заметим, что обоснование математической модели такой структуры столь же просто, как и в случае продольно связанных осцилляторов [10], и столь же наглядно, как в традиционной дискретной модели струны (см. например [11]).

Уравнения собственных колебаний цепочки в матричном виде имеют вид

$\ddot{x} = Cx$

где $$C = \begin{pmatrix} -2\omega_o & \omega_o & 0 & \dots & 0 & 0 \\ \omega_o & -2\omega_o & \omega_o & \dots & 0 & 0 \\ 0 & \omega_o & -2\omega_o & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -2\omega_o & \omega_o \\ 0 & 0 & 0 & \dots & \omega_o & -2\omega_o \end{pmatrix}.$$

а $\omega_o = c/m$.

Собственные частоты колебаний цепочки отыскиваются как мнимые части собственных чисел матрицы

$$A = \begin{bmatrix} 0 & I \\ C & 0 \end{bmatrix}$$

Здесь I – единичная матрица. Собственные формы, соответствующие указанным собственным частотам, отыскиваются как подвекторы соответствующих собственных векторов.

Текст функции, проводящей соответствующие вычисления для произвольного числа звеньев приводится ниже

```
function [om,sv]=escalat1(m,c,N)
col=['y','b','r','g','m','k','c'];
Cm=zeros(N);
for k=1:N-1,
   Cm(k,k)=-2*c/m;
   Cm(k+1,k)=c/m;
   Cm(k,k+1)=c/m;
end
Cm(N,N)=-2*c/m;
A=[zeros(N),eye(N);Cm,
      zeros(N)];

[sv,Lam]=eig(A);

l=diag(Lam);
om=imag(l(1:2:2*N));
figure(1)
clf
hold on
for k=1:2:2*N,
   if abs(real(sv(N+1:2*N,k))) <= abs(imag(sv(N+1:2*N,k))),
      plot([0;imag(sv(N+1:2*N,k));0],col(rem((k+1)/2,7)+1))
   else
      plot([0;real(sv(N+1:2*N,k));0],col(rem((k+1)/2,7)+1))
   end
```

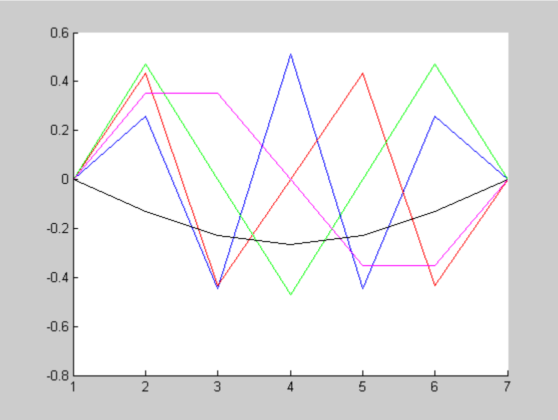

Рис. 14. Собственные формы колебаний цепочки из 5 звеньев.

```
end
hold offsys=ss(A,ones(length(A),1),eye(size(A)),zeros(length(A),1));
x0=real(sv(:,1));
[x,t]=initial(sys,x0);
figure(2)
clf
plot([0;x0(1:N);0]);
hold on
for i=1:30,
  plot([0,x(i,1:N),0]);
end
hold off
plot(t,x(:,1:2:end))
```

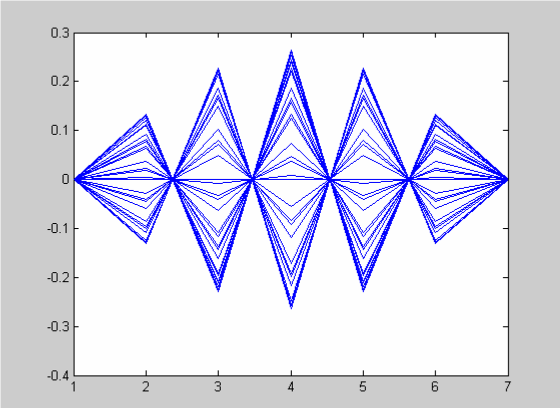

Рис. 15. Положения центров звеньев цепочки при собственных колебаниях по одной из собственных форм.

Функция *initial* в этом примере использована для получения решений систем линейных дифференциальных уравнений с постоянными коэффициентами. В равной степени для этого можно использовать функции решателей серии *ode.*

Литература

1. Болотин Ю.В., Голован А.А., Кручинин П.А., Парусников Н.А., Тихомиров В.В., Трубников С.А. Задача авиационной гравиметрии. Алгоритмы. Некоторые результаты испытаний.// Вестник МГУ. Математика. Механика. 1999. N 2. C. 36-41.
2. Хардле В. Прикладная непараметрическая регрессия.// М.: Мир. 1993. 349 с.
3. Черноусько Ф.Л., Болотник Н.Н., Градецкий В.Г. Манипуляционные роботы.// М., Наука, 1989, 368 с.
4. Devjanin E.A., Budanov V.M. Motion Control for the Six-Legged Walking Machine.- Proceedings European Mechanics Colloquium Euromech 375. Biology and Technology of Walking.// Germany, Munich, 1998, pp. 101-107.
5. Beletsky V.V., Kasatkin G.V., Starostin E.L. The Pendulum as e Dynamical Billiard // Chaos, Solitons & Fractals, 1996, v. 7, No. 8, pp. 1145-1178.
6. Leick A. GPS satellite surveying. // New York Chichester Brisbane Toronto Singapore, Wiley, 1995.
7. Вавилова Н.Б., Голован А.А., Парусников Н.А., Трубников С.А. Математические модели и алгоритмы обработки измерений спутниковой навигационной системы GPS. Стандартный режим. // М., Издательство центра прикладных исследований при механико-математическом факультете МГУ, 2001, 116 с.
8. Mourey F., Grishin A., d'Athis P., Pozzo T., Stapley P.Standing up from a chair as a dynamic equilibrium task: A comparison between young and elderly subjects.// Journal of Gerontology. A. Biol. Sci. Med. Sci. 2000. V. 55. N 9. PP 425-431.
9. И.В.Новожилов, П.А.Кручинин, И.А.Копылов, А.М.Журавлев, А.А.Гришин, П.П.Демин, С.В.Куликовский, Е.М.Моисеева Математическое моделирование сгибательно-разгибательных движений нижних конечностей при изменении вертикальной позы человека. // Издательство механико-математического факультета МГУ, Москва, 2001 г. 52 с.
10. Рабинович М.И., Трубецков Д.И. Введение в теорию колебаний и волн.// Москва Ижевск. НИЦ “Регулярная и хаотическая динамика”, 2000, 560 с.
11. Крауфорд Ф. Волны // М., Наука, 1984, (Берклеевский курс физики)- 512с.