электронное научно-техническое издание
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС 77 - 30569. Государственная регистрация №0420900025. ISSN 1994-0408

Численные методы в моделировании сочленённых транспортных систем

 # 10, октябрь 2010
авторы: Баженов Е. Е., Буйначев С. К.

УДК 629.113.028

Уральский Федеральный университет
имени первого Президента России Б.Н.Ельцина»

Прогнозирование эксплуатационных свойств сочленённых транспортных и технологических машин предполагает этап имитационного моделирования сложной динамической системы. Одним из этапов моделирования является решение системы дифференциальных уравнений движения транспортной системы. Процесс вычисления связан со значительными затратами машинного времени и с уменьшением точности из-за накапливающейся погрешности на каждом этапе расчётов.

Целями борьбы за вычислительные ресурсы в моделировании определим:

во-первых, уменьшение размерности уравнений решаемой задачи;

во-вторых, увеличение скорости вычислений, в-третьих, облегчение составления структуры модели.

Уравнение движения системы:

$$\Theta\ddot{\vartheta} = F + R$$

в котором:

ϑ - канонические координаты действия сил инерции;

Θ - коэффициенты инерции в направлении канонических координат;

F - приведенные к каноническим координатам внешние силы;

R - приведенные внутренние силы;

представим в виде [1] трехдольного графа (рис.1).

Граф определяет структуру программы при интегрировании системы дифференциальных уравнений. В соответствии со структурой графа необходима обработка связей и вершин средней доли. Сканирование связей необходимо осуществлять сначала между средней и левой долей, затем между средней и правой долей.

Создание программы производится на объектно-ориентированном языке программирования *Python*.

Взаимодействие вершин средней доли производится в среде, описываемой левой долей графа. Силы внутреннего взаимодействия *R* представляют один общий класс

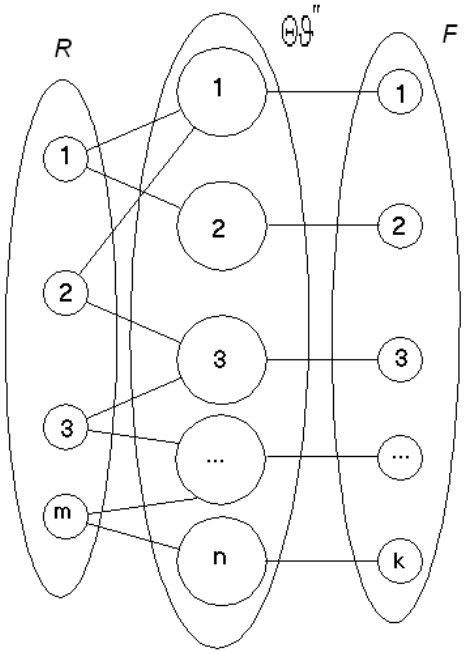

Рис.1. Трехдольный граф уравнения движения

```
class R:
 def_init_(self,script_i):
    """Вводится cript_i - сценарий внутреннего взаимодействия"""
    self,script_e=script_i
 def F(self):
    return self.script_i
```

Теперь создаем объекты взаимодействия

$$ri=R(script\_i),$$

и вставляем их в список связей левой и средней долей:

$$L=[(r1,m1,m2),(r2,m2,m3), ..., ...],$$

а затем обрабатываем, сканируя эти связи в цикле

```
for (r,m1,m2) in L:
  m1.f, m2.f=r.F()
```

Правая доля графа аналогична левой:

```
class F:
  def_init_(self,script_e):
    """Вводится сценарий внешнего нагружения"""
    self.script_e
  def F(self):
    return self.script_e
```

объекты внешнего воздействия:

$$fi=P(script\_e),$$

и список связей правой и средней долей:

$$W = [(f1,m1), (f2,m2), ..., ...,],$$

а его обработка

```
for (f,m) in L:
  mf=f.F()
```

Интегрант функционала (8) для каждой доли графа имеет свою составляющую [2]:

$T=\Theta\dot{\vartheta}/2$ - средняя доля;

$A_i = A_i(R)$ - левая доля;

$A_e = A_e(F)$ - правая доля;

$$B=\int(T+A_i+A_e)dt \rightarrow \min(\vartheta).$$

Силы внешнего и внутреннего воздействия на вершины средней доли пересчитываются между двумя отдельными шагами *dt* интегрирования дифференциального уравнения. Свойства состояний каждого объекта меняются после пересчета сил, интегрирования движения на шаге и смены положений.

Используем двухшаговый [3] метод численного интегрирования на интервале от 0 до 1 (рис. 2). Точка *D* соответствует решению на первом этапе по методу Ньютона. Производная в конечной точке конца интервала дает касательную семейства решений прямую – $FE \parallel AC$. На втором этапе используем метод трапеций, *AB* - гипотенуза угла $\angle DAC$. Точка *B* – решение на втором этапе интегрирования. Второй этап можно повторять как несколько итераций на шаге интегрирования, добиваясь требуемой точности.

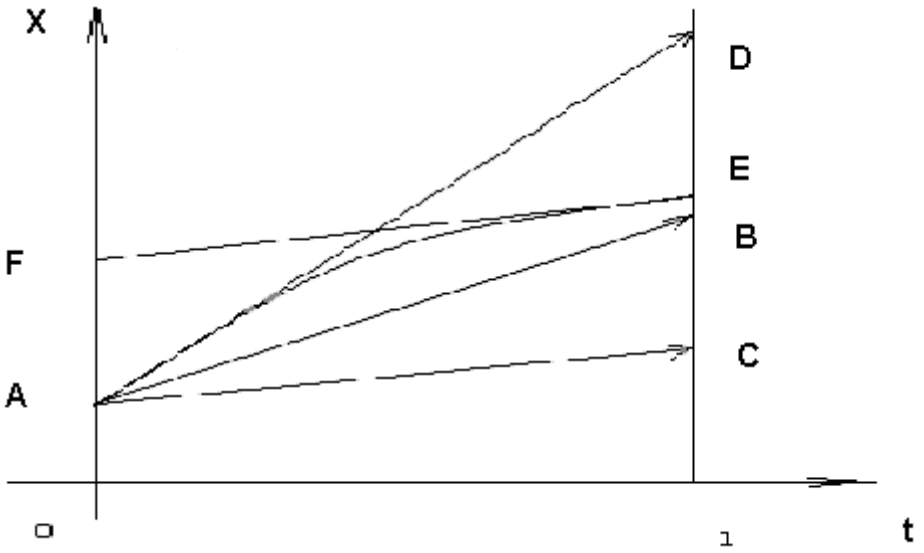

Рис. 2. Схема двухэтапного метода

Каждая вершина средней доли графа соответствует одной обобщенной координате, линейной или угловой:

$$am = F \quad \Rightarrow \quad \begin{Bmatrix} \dot{S} = V \\ \dot{V} = F/m \end{Bmatrix}$$

$$\varepsilon J = M \quad \Rightarrow \quad \begin{Bmatrix} \dot{\varphi} = \omega \\ \dot{\omega} = M/J \end{Bmatrix}.$$

Системы с переменными передаточными функциями и переменными инерционными коэффициентами включают дополнительное слагаемое:

$$\varepsilon J + \frac{1}{2} J_{\varphi} \omega^2 = M_c + M_D \quad \Rightarrow \quad \begin{Bmatrix} \Phi = \omega \\ \omega = \left(M_D + M_c - \frac{1}{2} J_{\varphi} \omega^2 \right) \Big/ J \end{Bmatrix}$$

Для всех уравнений, описывающих движения объекта используем следующий алгоритм интегрирования

$$V_1 = V_0 + f \, \delta t$$
$$S_1 = S_0 + \frac{1}{2} \left(V_0 + V_1 \right) \delta t \ ,$$

где *S,V* - вектор фазовых координат; *f* – ускорение или единичная сила; *dt* – шаг интегрирования.

Для всех динамических объектов достаточно одного класса.

```
class M:
  def __init__(self,m=0.0,x=[0.0,0.0,0.0],f=0.0): # Конструктор класса
    """ x=[положение, скорость, ускорение] """
    self.m, self.x, self.f =m,x,f
  def step(self,dt):
    if self.m == 0.0:
self.x=[0.0,0.0,0.0]
  return
else:          # Поведение объекта на шаге интегрирования
  self.x[2]=self.f/self.m
  self.x[0]+=0.5*self.x[1]*dt
  self.x[1]+=self.x[2]*dt
  self.x[0]+=0.5*self.x[1]*dt
```

Создание *i*-го динамического объекта с шестью степенями свободы производится в два этапа. На первом этапе создаются объекты

```
mxi=M(50000.0,[0.0,0.0,0.0],0.0),
myi=M(50000.0,[0.0,0.0,0.0],0.0),
mzi=M(50000.0,[0.0,0.0,0.0],0.0),
Jxi=M(1500.0,[0.0,0.0,0.0],0.0),
Jyi=M(2000.0,[0.0,0.0,0.0],0.0),
Jzi=M(700.0,[0.0,0.0,0.0],0.0),
```

На втором этапе вершины средней доли графа (рис. 1) «раскрашиваются»,

```
m=[mx1, my1, mz1, Jx1, Jy1, Jz1, mx2, my2, mz2, Jx2, Jy2, Jz2, ... , ... ,  ]
```

то есть создается список ссылок на вершины, которые сканируются при интегрировании.

```
for mi in m:
    mi.step(dt) .
```

- это результат перемещения системы в новое положение после приложения к элементам системы внешних и внутренних сил.

Сочленённые транспортные и технологические системы отличаются большим количеством упругих связей, элементов управления и саморегулирования, а также случайным характером сил внешнего внутреннего взаимодействия. Предлагаемый подход позволяет изменять шаг интегрирования для каждого отдельного дифференциального уравнения в зависимости от характера движения отдельно взятого объекта без увеличения количества используемых ресурсов, а значит существенно улучшить качество исследуемой модели.

Список литературы

1. Новиков. Ф.А. Дискретная математика для программистов. СПб.: Питер. 2002. 304с.
2. Динамика управляемых машинных агрегатов/Вейц В.Л., Коловский М.З., Кочура Е.А. М.: Наука. Главная редакция физико-математической литературы, 1984.352 с.
3. Березин И.С., Жидков Н.П. Методы вычислений. Т.2.М.: Физматгиз, 1962.