Инженерный вестник Дона, №1 (2016)
ivdon.ru/ru/magazine/archive/n1y2016/3511

Использование сетки конечных элементов, построенной программой GMesh, для вычислений в программе Mathcad

А.М. Филипов[1], *М.В. Хоменко*[2]

[1]*ВолгГАСУ, Волгоград*
[2] *Государственный университет по землеустройству, Москва*

Аннотация: в статье приводятся алгоритмы и листинги пользовательских функций для импортирования сеток элементов в программную среду комплекса Mathcad, для их последующего анализа численными методами. А так же приведено решение по экспорту полученных результатов в программу GMesh для последующей визуализации.
Ключевые слова: визуализация результатов расчета, импорт сеток элементов со сложной топологией, экспорт сеток элементов с произвольной топологией, GMesh, Mathcad.

При разработке численных алгоритмов [1-3], связанных с обработкой сложных двух- и трехмерных сеток и работ по их оптимизации существует необходимость в использовании относительно большеразмерных моделях со сложной топологией, а так же в предоставлении результатов в удобочитаемом виде – графиках, изополях, изоповерхностях [4,5]. Одной из достаточно простых и универсальных программ математического моделирования является Mathcad и его аналоги. Однако в их возможности обычно не входят средства построения 4-х мерных графиков и окраска поверхности и (или) отрезков в соответствии с заданными пользователем значениями.

Для достижения этих целей можно:

– применить сторонние готовые программные решения типа GMesh [6] и аналогичные (например, Array Visualizer позволяет отобразить узловые значения для 3-х мерной пространственной сетки без отображения самой сетки);
– самостоятельно разработать такое решение, используя специализированные библиотеки, как например MatFOR (разработка AnCAD), или общие, такие как OpenGL [7,8], GTK, QT, и др.

Однако второй вариант, имея наиболее оптимальный подход для задачи отображения результата, не решает проблемы с построением исходных сеток узлов и элементов для численного анализа.

В данной статье предлагается описание функций Mathcad'a (MathSoft, Inc.) для импортирования исходной сетки узлов и элементов и экспорта результата в программу GMesh. При этом поставленная задача может быть решена разными способами, перечислим их в порядке сложности:

– ручное преобразование исходных файлов;
– использование программ преобразования файлов;
– получение необходимых данных, а так же формирование данных для отображения стандартными функциями Mathcad'a [9];
– использование средств разработки userefi [9,10].

Из предложенных способов первые два не обеспечивают автоматизации, последний требует хороших знаний в области программирования на языке C++ и написания динамически подгружаемых библиотек. Соответственно, наиболее оптимальным решением будет использование стандартных средств системы Mathcad, о чем и пойдет речь далее в данной статье.

Для импортирования данных используется алгоритм, основаный на особенностях используемой среды:

– автоматическое размещение массивов;
– массивы могут содержать одновременно как строковые, так и численные данные;
– ограниченный набор функций работы с символьными строками.

Чтение данных выполняется функцией READFILE [9,10] с параметрами "fixed" и числом предполагаемых символов в строке (оно же число столбцов) равным 256 – это значение достаточно для считывания всех атрибутов элемента, имеющего до 30 узлов. Т.е. команда чтения будет иметь

вид: Atmp = READFILE(file_name,fixed,256). При этом функция передает вектор строк (см. рис. 1).

Atmp =

	1
1	"$MeshFormat"
2	"2.2 0 8"
3	"$EndMeshFormat"
4	"$Nodes"
5	181
6	"1 0 0 0"
7	"2 10 0 0"
8	...

Рис. 1 – фрагмент полученных из файла с сеткой элементов данных

Структура данных [6]:

1. Описание формата файла.
2. Описание атрибутов узлов сетки.
3. Описание атрибутов элементов.
4. Описание параметров результатов и данные результатов.

П.п. 4 является необязательным и может повторяться внутри одного файла.

В нашем случае интерес при импортировании представляют блоки 2 и 3 (при получении массива атрибутов элементов сетки) и, соответственно, блоки 2-4 при экспортировании сетки элементов.

Все блоки имеют начальную строку с заголовком блока, размер блока и конечную строку соответственно: **$<Имя блока>, <размер блока>** и **$End<Имя блока>**, (см. рис.1 строки 1, 3 и 4, 5). Это упрощает извлечение интересующих данных.

Рассмотрим подробнее функцию преобразования исходных данных в массив координат узлов и аналогичную, преобразующую исходные данные в массив атрибутов элементов (см. рис.2а). Т.к. функции аналогичные и отличаются только параметрами поиска интересующих блоков (выполняется

поиск слов «Nodes» и «Elements» соответственно), то ниже рассматривается первая из них.

Первоначально в цикле WHILE [9,10] выполняется поиск номера исходного массива, содержащего строку символов с интересующим заголовком блока, и вычисляются номера первого и последнего элемента, содержащего данные. Далее выбранный диапазон элементов исходного массива поэлементно передается пользовательской функции Str2NumVect(), выполняющей преобразование строки символов в числовой вектор. Листинг функции Str2NumVect() приведен на рис. 2б

Примечание: в приведенном листинге исходные данные содержит массив Atmp.

```
node(Atmp) := | fi ← (0  0)^T
              | i1 ← 0
              | while fi_1 = 0
              |     i1 ← i1 + 1
              |     if IsString[(Atmp^⟨1⟩)_i1] = 1
              |         fi_1 ← i1 + 2  if search[(Atmp^⟨1⟩)_i1, "Nodes", 0] > 0
              |         fi_2 ← (Atmp^⟨1⟩)_(fi_1 − 1) + fi_1 − 1  if fi_1 > 0
              | for i1 ∈ fi_1 .. fi_2
              |     nd ← Str2NumVect[(Atmp^⟨1⟩)_i1]
              |     for i2 ∈ 1 .. rows(nd)
              |       nt_(i1 − fi_1 + 1, i2) ← nd_i2
              | nt
```

а.

```
Str2NumVect(T) := | AT ← adjustl(T)
                  | AT ← LCompact(AT)
                  | i1 ← 1
                  | while search(AT, " ", 0) > 0
                  |     i2 ← search(AT, " ", 0)
                  |     line ← substr(AT, 0, i2)
                  |     num_i1 ← str2num(line)
                  |     i1 ← i1 + 1
                  |     AT ← lshift(AT, i2 + 1)
                  | num
```

б.

Рис.2 – Листинг функции преобразования исходных данных в массив координат узлов (а) и входящая в ее состав функция преобразования строки символов в числовой вектор (б)

Входящие в состав Str2NumVect() пользовательские функции adjustl(), LCompact() и lshift() выполняют выравнивание строки по левому краю, удаление повторяющихся пробелов и сдвиг элементов строки на указанное число позиций влево соответственно. Ввиду отсутствия в Mathcad'e возможности обратится к элементу строки без применения функции substr(),

обработка строк пользовательскими функциями осуществляется после их преобразования в последовательность ANSI-кодов и обратным преобразованием после (функции str2vec() и vec2str() [9,10] соответственно). Листинги приведены на рис.3а-в соответственно.

```
adjustl(T) := | AT ← str2vec(T)
              | a1 ← rows(AT)
              | a2 ← 1
              | while AT_a2 = 32
              |   a2 ← a2 + 1
              | for i1 ∈ 1.. a1 − a2 + 1
              |   AT_i1 ← AT_i1+a2−1
              | for i1 ∈ a1 − a2 + 1.. a1  if  a2 > 1
              |   AT_i1 ← 32
              | num ← vec2str(AT)
              | num
```

а.

```
LCompact(T) := | AT ← str2vec(T)
               | AT_rows(AT)+1 ← 32
               | i2 ← 1
               | for i1 ∈ 1.. rows(AT) − 1
               |   if AT_i1 ≠ 32 ∨ AT_i1+1 ≠ 32
               |     | ATT_i2 ← AT_i1
               |     | i2 ← i2 + 1
               | TC ← vec2str(ATT)
               | TC
```

б.

```
lshift(T, s) := | AT ← str2vec(T)
                | for i1 ∈ s + 1.. rows(AT)
                |   ATT_i1−s ← AT_i1
                | for i1 ∈ rows(AT) − s + 1.. rows(AT)
                |   ATT_i1 ← 32
                | l1 ← vec2str(ATT)
                | l1
```

в.

Рис.3 – листинги функций adjustl (а), LCompact(б), lshift(в)

Наличие функций adjustl() и LCompact() необязательно, если осуществляется обмен данными только между Mathcad'ом и GMesh'ем, в остальных случаях – желательно.

Формирование файла с результатами для отображения в GMesh осуществляется путем формирования массива символьных строк и его последующей записи в файл, аналогично приведенным в статье [11].

В состав пользовательской функции, формирующей массив содержащий образ записываемого файла, res() входят следующие элементы:

1. res_header – формирование блока описания формата файла;
2. res_node() – формирование блока описания узлов;
3. res_elem() – формирование блока описания атрибутов элементов;
4. res_result() – формирование блока описания узловых значений.

Формирование блоков данных производится в соответствии с требованиями [6], а именно: разделитель значений – пробел, разделитель целой и дробной части – точка. Блок описания узловых значений имеет дополнительные 8 строк [6]. Листинги пользовательских функций приведены соответственно на рис.5а–д. Используемые переменные и массивы:

5. elem – массив с описанием элементов в соответствии с требованиями («№ элемента», «тип элемента», «кол-во дополнительных атрибутов», «атрибуты», «номера узлов»);
6. Name – имя графика (не поддерживаются символы кириллицы и спец.символы);
7. node – массив с координатами узлов сетки;
8. Time – позиция шкалы времени, для которой выводятся результаты;
9. Time_unit – единицы измерения шкалы времени результата;
10. Res – вектор узловых значений.

Функция r_out() нужна для записи числа в формате с плавающей точкой при его целом значении, т.е. если переменная равна «1», то ее строчная запись все равно будет иметь вид «1.00», листинг функции приведен на рис.5е.

```
res(N,E,Name,Tm,NRes) := stack(res_header,res_node(N),res_elem(E),res_result(Name,Tm,NRes))
```

```
C...\test_export.msh

res(node,elem,Name,Time,Res)
```

а.

```
res_header := ("$MeshFormat"  "2.2 0 8"  "$EndMeshFormat")^T
```

б.

```
res_node(N) := | AT ← ("$Nodes"  num2str(rows(N)))^T
               | for i1 ∈ 1..rows(N)
               |   AT_(2+i1) ← concat(num2str(N_(i1,1)),"  ",num2str(N_(i1,2)),"  ",num2str(N_(i1,3)),"  ",num2str(N_(i1,4)))
               | AT_(3+rows(N)) ← "$EndNodes"
               | AT
```

в.

```
res_elem(E) := | AT ← ("$Elements"  num2str(rows(E)))^T
               | for i1 ∈ 1..rows(E)
               |   | AT_(2+i1) ← concat(num2str(E_(i1,1)),"  ",num2str(E_(i1,2)),"  ",num2str(E_(i1,3)))
               |   | i3 ← E_(i1,3) + 3
               |   | for i2 ∈ 4..cols(E)
               |   |   AT_(2+i1) ← concat(AT_(2+i1),"  ",num2str(E_(i1,i2)))  if E_(i1,i2) ≠ 0 ∨ i2 ≤ i3
               | AT_(3+rows(E)) ← "$EndElements"
               | AT
```

г.

```
res_result(Name,Time,Res) := | t1 ← concat("""",Name,",","  (Time: ",r_out(Time),"  ",Time_unit,")"")
                             | AT ← ("$NodeData"  "1"  t1  "1"  r_out(Time)  "3"  "0"  "1")^T
                             | AT_9 ← num2str(rows(Res))
                             | for i1 ∈ 1..rows(Res)
                             |   AT_(9+i1) ← concat(num2str(i1),"  ",r_out(Res_(i1)))
                             | AT_(10+rows(Res)) ← "$EndNodeData"
                             | AT
```

д.

```
r_out(r) := | a1 ← num2str(r)
            | a1 ← concat(a1,".00")  if search(a1,".",0) < 0
            | a1
```

е.

Рис.5 – листинг функций формирования и записи файла для GMesh

Исходная сетка элементов в программе GMesh приведена на рис.5а. В качестве примера рассмотрена сетка элементов, содержащая одноузловые, линейные и плоские элементы. Узловые значения получены по формуле *F(x)= Cos(8x)* см. рис.5б.

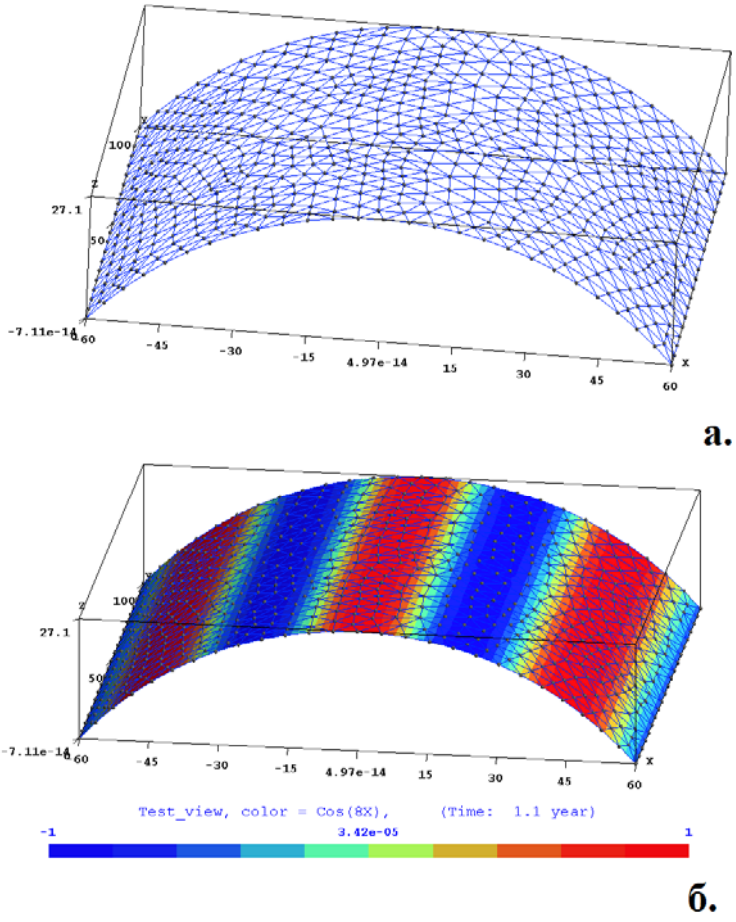

Рис.5 – внешний вид исходной импортируемой сетки (а) и результат, отображаемый в программе GMesh (б)

Литература

1. Варвак П. М., Варвак Л.П. Метод сеток в задачах расчета строительных конструкций. М.: Стройиздат, 1977. 154 с.

2. Сабоннадьер Ж.К., Кулон Ж.Л. Метод конечных элементов и САПР: Пер. с франц.. М.: Мир, 1989. 190 с.

3. Singiresu, S.R., 2011. The Finite Element Method in Engineering. Elsevier UK, 726 p.

4. Лахов А. Я. Программное обеспечение для стереовизуализации результатов конечно-элементного моделирования // «Инженерный вестник Дона». 2013. №1. URL: ivdon.ru/ru/magazine/archive/n1y2013/1501

5. Рачковская Г. С. Математическое моделирование и компьютерная визуализации сложных геометрических форм // «Инженерный вестник Дона». 2013. №1. URL: ivdon.ru/ru/magazine/archive/n1y2013/1498

6. Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities URL: gmsh.info/doc/texinfo/gmsh.html (дата обращения: 2.03.2016).

7. Евченко А. И. OpenGL и DirectX: программирование графики. Для профессионалов. СПб.: Питер, 2006. 350 с.

8. Lawrence, N., 2002. Compaq Visual Fortran: A Guide to Creating Windows Applications. Digital Press, 482 p.

9. Очков В.Ф. Mathcad 14 для студентов и инженеров русская версия. СПб.: БВХ-Петербург, 2009. 512 с.

10. Mathsoft, I., 2001. Mathcad: user's guide with reference manual : Mathcad 2001 professional. MathSoft, Inc., 528 p.

11. Филипов А. М. Подготовка файлов с результатами численного анализа для последующего графического отображения в программе Gmesh // Естественные и технические науки. 2014. №75. С. 4.

References

1. Varvak P. M., Varvak L.P. Metod setok v zadachakh rascheta stroitel'nykh konstruktsiy [The grid method in problems of design of structures]. M.: Stroyizdat, 1977. 154 p.

2. Sabonnad'er Zh.K., Kulon Zh.L. Metod konechnykh elementov i SAPR: Per. s frants. [Finite Element Method and CAD: Trans. from France]. M.: Mir, 1989. 190 p.

3. Singiresu, S.R., 2011. The Finite Element Method in Engineering. Elsevier UK, p. 726.

4. Lakhov A. Ya. Inženernyj vestnik Dona (Rus), 2013. №1, URL: ivdon.ru/ru/magazine/archive/n1y2013/1501.

5. Rachkovskaya G. S. Inženernyj vestnik Dona (Rus), 2013. №1. URL: ivdon.ru/ru/magazine/archive/n1y2013/1498.

6. Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities URL: gmsh.info/doc/texinfo/gmsh.html (Date Views 2.03.2016).

7. Evchenko A. I. OpenGL i DirectX: programmirovanie grafiki. Dlya professionalov [OpengL and DirektH: graphics programming. For professionals]. SPb.: Piter, 2006. 350 p.

8. Lawrence, N., 2002. Compaq Visual Fortran: A Guide to Creating Windows Applications. Digital Press, 482 p.

9. Ochkov V.F. Mathcad 14 dlya studentov i inzhenerov russkaya versiya [Mathsad 14 for students and engineers Russian version]. SPb.: BVKh-Peterburg, 2009. 512 p.

10. Mathsoft, I., 2001. Mathcad: user's guide with reference manual : Mathcad 2001 professional. MathSoft, Inc., 528 p.

11. Filipov A. M. Podgotovka faylov s rezul'tatami chislennogo analiza dlya posleduyushchego graficheskogo otobrazheniya v programme Gmesh. Estestvennye i tekhnicheskie nauki. 2014. №75. p. 4.