

# **Автоматизированное моделирование и расчёт конструкций в ANSYS**

## **Одномерные модели**

*Учебное пособие*

**КАЗАНСКИЙ УНИВЕРСИТЕТ**

**2019**

**УДК 531/534, 004.942**  
**ББК 22.251\*018.2\*32.973**  
**A42**

*Печатается по рекомендации учебно-методической комиссии  
Института математики и механики им. Н.И. Лобачевского  
Казанского (Приволжского) федерального университета  
протокол № 2 от 28 октября 2018 года*

**канд. физ.-мат. наук О.А. Саченков,**  
**канд. физ.-мат. наук, доц. А.А. Саченков,**  
**П.В. Большаков, О.В. Герасимов**

**Научный редактор**  
**канд. физ.-мат. наук, доц. С.А. Кузнецов,**

**Рецензент**  
**канд. физ.-мат. наук Ф.Р. Шакирзянов**

**Автоматизированное моделирование и расчёт конструкций в ANSYS: одномерные модели:** учеб. пособие / О.А. Саченков, А.А. Саченков, П.В. Большаков, О.В. Герасимов. – Казань: Казан. ун-т, 2019. – 140 с.

Пособие предназначено для проведения лекционных, лабораторных и практических занятий по курсам «Механика упругих стержней», «Численные методы теории упругости», «Программные системы инженерного анализа», а также для проведения исследований в рамках курсовых и выпускных квалификационных работ. Рассматриваются вопросы программирования в среде APDL для создания автоматизированных решений на базе метода конечных элементов в программном обеспечении ANSYS. Помимо общего описания структуры и логики построения конечно-элементной модели приведены примеры решения ряда задач.

**УДК 531/534, 004.942**  
**ББК 22.251\*018.2\*32.973**

**© Саченков О.А., Саченков А.А., Большаков П.В., Герасимов О.В., 2019**

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
ГЛАВА 1. ЭЛЕМЕНТЫ ПРОГРАММИРОВАНИЯ.....	8
Макросы.....	8
Задание переменных.....	8
Массивы и таблицы.....	9
Ветвление.....	10
Циклы.....	11
Безусловный переход.....	12
ГЛАВА 2. ПРЕПРОЦЕССОР.....	13
Системы координат.....	14
Построение геометрии.....	14
Пример №1. Применение графических команд.....	18
Пример №2. Построение параболы $x=y^2/2$ .....	26
Пример №3. Построение функции $y=\text{Sin}(x)$ .....	27
Задание конечного элемента (Element type).....	29
Конечный элемент BEAM188.....	30
Конечный элемент LINK180.....	32
Конечный элемент CERIG.....	33
Задание материала (Material props).....	33
Задание постоянных элемента (Real constant).....	35
Задание сечения (Section).....	35
Привязка материала и сечения.....	38
Разбиение на сетку (Meshing).....	39
Изменение атрибутов элемента.....	43
Объединение объектов.....	44
ГЛАВА 3. ПРОЦЕССОР.....	46
Массовые силы.....	47
Граничные условия.....	47
Пример.....	51
ГЛАВА 4. ПОСТПРОЦЕССОР.....	58
Деформированное состояние и векторные поля.....	59
Контурные поля.....	61
Табличные данные и вывод.....	62
Анимация.....	67

Выгрузка данных .....	68
ГЛАВА 5. ДОПОЛНИТЕЛЬНЫЕ ИНСТРУМЕНТЫ.....	70
Выбор элементов геометрии и сетки .....	70
Компоненты и сборки.....	73
Извлечение данных *GET .....	76
Создание макросов .....	78
Работа с файлами .....	82
ГЛАВА 6. РАЗБОР ЗАДАЧ.....	86
Задача 1. Расчёт бруса переменного сечения на сжатие .....	86
Задача 2. Влияние на прогиб радиуса кривизны кривого бруса, нагруженного равномерным давлением.....	92
Задача 3. Расчёт рамы под действием неравномерной нагрузки.....	98
Задача 4. Определение массово-прочностных характеристик конструкции.....	109
Задача 5. Расчёт фермы из разномодульного материала .....	120
Задача 6. Расчёт рамы с трением в скользящей опоре.....	127
РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА .....	137
ПРИЛОЖЕНИЯ.....	138

## ВВЕДЕНИЕ

*ANSYS* — универсальный программный продукт конечно-элементного анализа, являющийся одним из «гигантов» в сфере автоматизированных инженерных расчётов. *ANSYS* позволяет проводить решение методом конечных элементов (МКЭ) линейных и нелинейных, стационарных и нестационарных пространственных задач механики деформируемого твёрдого тела и конструкций, жидкости и газа, теплопередачи и теплообмена, электродинамики, акустики, а также прочих задач механики. Система работает на основе геометрического ядра Parasolid. Программная система КЭ-анализа *ANSYS* разрабатывается американской компанией ANSYS, Inc.

В данном пособии рассматриваются одномерные модели. Структурно можно выделить следующие этапы в построении модели: препроцессор, процессор, постпроцессор. В препроцессоре создаётся геометрия и/или КЭ-сетка расчётной области, указываются граничные условия: статические, кинематические и др. В процессоре определяется тип решаемой задачи, параметры и настройки для расчета. В постпроцессоре реализуется вывод результатов. Интерфейс *ANSYS* представлен двумя типами: командным и графическим (см. рис. 0.1). Настоящее учебное пособие ориентировано на командный интерфейс. Необходимо отметить некоторые общие синтаксические правила:

1. В каждой строке один оператор. Допускается объединение операторов знаком \$, однако это ухудшает читабельность кода.
2. Не требуется описание типов переменных: все переменные по умолчанию воспринимаются как действительные.
3. Существует автоматическое распознавание переменных.
4. Для имён переменных используются **только** латинские буквы, регистр не различается;
5. В именах не допускается применение символов:

! @ # \$ % & ^ \* ( ) \_ - + = | \ { } [ ] “ ‘ / < > ~

6. Строчные комментарии реализуются с помощью символа:

!

7. Строка команды ограничена: максимум 255 символов.

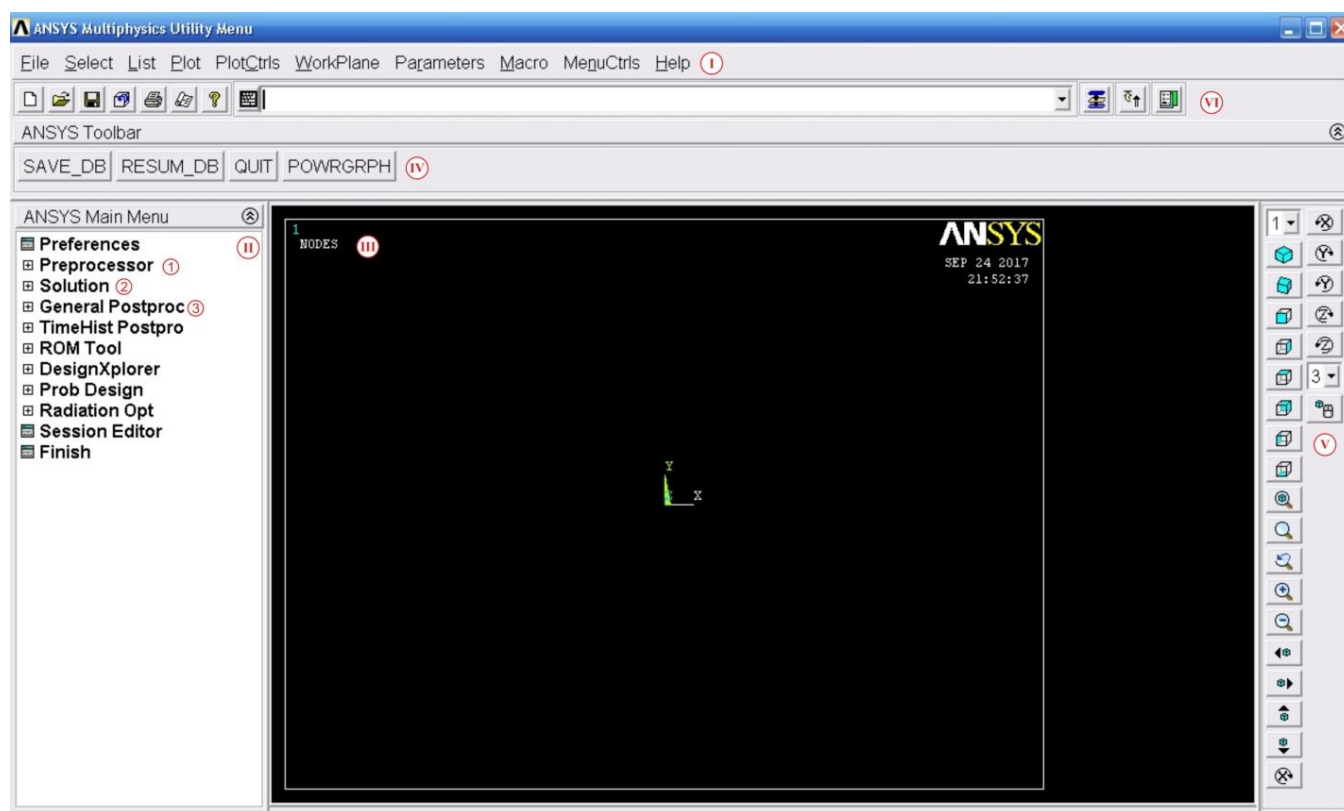


Рис. 0.1. Рабочее окно программы: I – строка меню, II – главное меню ANSYS, III – графическая область, IV – панель инструментов, V – панель инструментов графической области, VI – командная строка. 1 – препроцессор, 2 – процессор, 3 – общий постпроцессор.

Командный интерфейс в ANSYS состоит из команд APDL (ANSYS Parametric Design Language), а также команд Fortran. При использовании графического интерфейса командный аналог дублируется в файле истории команд, который можно просмотреть, используя меню:

*File → List → Log file,*

или открыв сам текстовый файл в директории проекта.

Далее по тексту примеры, реализованные на языке APDL, будут выделены шрифтом Courier New. Аргументы команд обозначаются по правилам

формального описания языка и выделяются угловыми скобками < >; необязательные аргументы в тексте выделены квадратными скобками [ ]. Для удобства работы с командными файлами авторы рекомендуют программу *Notepad++* с настройкой синтаксиса на язык Fortran (free form).

Также будут использованы следующие сокращения: МКЭ – метод конечных элементов, КЭ – конечный элемент/конечно-элементный, ГУ – граничные условия, СК – система координат, НДС – напряжённно-деформированное состояние.

В конце пособия приведен список рекомендованной литературы. В работе [1] дается строгое введение в МКЭ в механике деформируемых твердых тел, рассматриваются различные элементы, особенности их применения. Работа [2] более проста для понимания, в ней рассматриваются классические балочные, плоские и трехмерные элементы, в книге подробно разобраны примеры. В работах [3-5] рассматриваются вопросы построения КЭ моделей в *ANSYS*.

# ГЛАВА 1. ЭЛЕМЕНТЫ ПРОГРАММИРОВАНИЯ

## Макросы

Макрос – набор команд для *ANSYS*. Файл с командами может быть введен вставку из буфера в командную строку (см. рис. 0.1 п. IV), прочтён из файла с помощью меню:

*File → Read Input from,*

или определён командой */INPUT* (подробнее про работу с макросами смотрите главу 5).

## Задание переменных

Для задания переменных можно использовать оператор =:

**<Var> = <Value>**

например,

```
A=5
```

или команду:

**\*SET, <Var>, <Value>**

например,

```
*SET, A, 5
```

На практике удобнее использовать первый вариант.

Список активных переменных и их актуальные значения можно получить с помощью меню графического интерфейса:

*Parameters → Scalar Parameters.*

Для удаления переменной используется оператор:

**<Var> = <Nul>**

или

**\*SET, <Var>, < Nul >**

например:

```
*SET, A,
```



## Массивы и таблицы

Для создания массива можно использовать графический интерфейс:

*Parameters* → *Array Parameters* → *Define/Edit*

или команду

**\*DIM, <Name>, <Type>, <N1>, <N2>, <N3>, <V1>, <V2>, <V3>, <CSid>**

Описание аргументов команды:

**Name** – название набора данных;

**Type** – тип данных: TABLE – таблица, может быть 1D-, 2D- или 3D-размерности (подробнее про использование таблиц написано в главе 3); TAB4 – четырёхмерная таблица; TAB5 – пятимерная таблица; ARRAY – одно-, двух- или трёхмерный массив; ARR4 – четырёхмерный массив; ARR5 – пятимерный массив; CHAR – символьный (до 8 символов) массив; STRING – массив строк (до **N1** символов в строке);

**N1** – первая размерность;

**N2** – вторая размерность;

**N3** – третья размерность.

**V1** – переменная для первой размерности. Определяется для таблиц (TABLE), может принимать следующие значения: TIME – время, FREQ – частота, X – координата икс, Y – координата игрек, Z – координата зет (координаты определяются заданной системой координат – параметр CSid), TEMP – температура, VELOCITY – скорость, PRESSURE – давление, SECTOR – циклическая координата (подробнее о применении переменных – в главе 3);

**V2** – переменная для второй размерности;

**V3** – переменная для третьей размерности;

**CSid** – ссылка на номер системы координат (подробнее о системе координат описано в главе №2).

Приведем пример объявления массива и задания значений:

```
*DIM, Data, ARRAY, 3, 1  
Data(1,1)=20
```

```
Data(2,1)=40  
Data(3,1)=80
```

или

```
*DIM, Data, ARRAY, 3, 1  
Data(1,1)=20,40,80
```

Отметим специальный тип TABLE – массив со встроенной линейной интерполяцией данных. Рассмотрим пример:

```
*DIM, Data, TABLE, 3, 1  
Data(1,0) = 1  
Data(2,0) = 2  
Data(3,0) = 3  
Data(1,1) = 20  
Data(2,1) = 40  
Data(3,1) = 80  
Res1 = Data(1.5,1)  
Res2 = Data(2.5,1)
```

Так, после работы приведенного кода в Res1 будет храниться значение 30, а в Res2 – 60.

Получить информацию о значениях элементов массива можно командой **\*STAT**:

**\*STAT, [<Name>]**

где **Name** – имя массива (или переменной).

При вызове команды без аргументов будет выведен список всех переменных совместно с их типом.

## Ветвление

Оператор ветвления

**\*IF, <V1>, <O1>, <V2>, <B1>, <V3>, <O2>, <V4>, <B2>**

Описание аргументов команды:

**V1** – первое сравниваемое значение;

**O1** – операция сравнения (с точностью до  $10^{-10}$  для переменных типа REAL): EQ – равенство, NE – неравенство, LT – меньше, GT – больше, LE –

меньше или равно, GE – больше или равно, ABLT – меньше по модулю, ABGT – больше по модулю;

**V2** – второе сравниваемое значение;

**B1** – литера, обозначающая действие при выполнении предыдущего логического выражения (значения ИСТИНА): :label – переход по ссылке, STOP – останавливает работу ANSYS, EXIT – выход из цикла, CYCLE – переход в конец цикла, THEN – конструкция ветвления; также здесь могут быть логические операторы: AND – логическое И, OR – логическое ИЛИ, XOR – логическое исключающее ИЛИ;

**V3** – третье сравниваемое значение;

**O2** – операция сравнения;

**V4** – четвертое сравниваемое значение;

**B2** – литера, обозначающая действие при выполнении предыдущих логических выражений (значений ИСТИНА с логическим И).

Для наглядности приведем стандартную конструкцию ветвления:

**\*IF,<V1>,<O1>,<V2>,THEN**

**<BODY1>**

**\*ELSE**

**<BODY2>**

**\*ENDIF**

## **Циклы**

### Безусловный цикл

**\*DO,<Iter>,<Iter0>,<Iter1>,<Inc>**

**<Body>**

**\*ENDDO**

Описание аргументов команды:

**Iter** – переменная (счётчик) цикла;

**Iter0, Iter1** – начальное и конечное значения переменной цикла;

**Inc** – шаг переменной цикла;

**Body** – операторы (тело) цикла.

### Условный цикл

**\*DOWHILE,<Scalar>**

**<Body>**

**\*ENDDO**

Описание аргументов команды:

**Scalar** – скалярная величина – цикл выполняется, пока её значение больше нуля;

**Body** – тело цикла.

### **Безусловный переход**

Также существует команда безусловного перехода, которую не рекомендуется использовать с точки зрения качества кода

**\*GO, <Label>**

где **Label** – ссылка на позицию в коде, обозначенная в программе в виде **:label** или **STOP** (завершение работы программы).

## ГЛАВА 2. ПРЕПРОЦЕССОР

При подготовке модели, расчёте и обработке результатов программа ANSYS создаёт временные и/или заданные пользователем файлы и сохраняет их в рабочей папке, для определения которой необходимо выбрать:

*File → Change Directory*

или воспользоваться командой

**/CWD, <DirName>**

где **DirName** – путь и название папки.

Преппроцессор включается командой:

**/PREP7**

или открытием папки *Preprocessor* в главном меню ANSYS (см. рис.1.1 п. 1).

Рассмотрим некоторые команды интерфейса, работающие с отображением объектов. В строке меню (см. рис. 0.1 п. II) отметим следующие пункты: *List* , *Plot*, *PlotCtrls*.

*List* – список – содержит команды по выводу списка заданных объектов (*Keypoints*, *Lines* и т.д.);

*Plot* – график – содержит команды по отображению в графической области заданных (*Keypoints*, *Lines* и т.д.) или всех (*Multi-Plot*) объектов;

*PlotCtrls* –параметры графиков – содержат настройки отображения заданных объектов в графической области. Здесь стоит отметить пункт нумерации *Numbering*, который содержит настройки по отображению номеров графических объектов:

**/PNUM, <Label>, <KEY>**

Описание аргументов команды:

**Label** – тип объекта, нумерация которого будет отображена: **NODE** – узел, **ELEM** – элемент, **MAT** – номер материала, **KP** – key point, **LINE** – линия;

**KEY** – флаг: 0 – отображение отключено, 1 – отображение включено.

## Системы координат

Геометрические построения зависят от СК, определение которой происходит в меню *WorkPlane*. По умолчанию существуют три глобальные СК: декартовая (Cartesian), цилиндрическая (Cylindrical) и сферическая (Spherical). Изменить тип можно в меню:

*WorkPlane* → *Change Active CS to*

или командой

**CSYS, <NCS>**

где **NCS** – номер системы координат: 0 – декартовая (по умолчанию), 1 – цилиндрическая (ось вращения – ось Z), 2 – сферическая, 4 или WP – рабочая плоскость, 5 – цилиндрическая (ось вращения – ось Y), 11 или больше – заданная пользователем.

Стоит отметить, что для цилиндрической и сферической СК угол Y задается в градусах в диапазоне от 0° до 360°. В сферической СК угол Z задается в градусах в диапазоне от -90° до 90°.

Для создания пользовательской СК применяется инструментарий, расположенный в меню:

*WorkPlane* → *Local Coordinate System* → *Create Local CS*.

## Построение геометрии

В ANSYS реализованы основные геометрические примитивы:

- 1) *Keypoints* – точечные объекты (0D);
- 2) *Lines* – линейные объекты (1D);
- 3) *Areas* – плоские объекты (2D);
- 4) *Volumes* – объемные объекты (3D).

В пособии будут рассмотрены 0D- и 1D-геометрические примитивы:

### 1. Keypoint – Ключевая точка

Синтаксис:

**K, [<NKPoint>], <XKPoint>, <YKPoint>, <ZKPoint>**

Описание аргументов команды:

**NKPoint** – номер точки: можно не указывать – в этом случае номер присваивается автоматически;

**XKPoint, YKPoint, ZKPoint** – координаты точки.

## 2. *Lines – Линии*

### 2.1. *Straight Line – Линия через две точки*

Синтаксис:

**LSTR, <KeyPoint1>, <KeyPoint2>**

или:

**L, <KeyPoint1>, <KeyPoint2>**

Здесь **KeyPoint1, KeyPoint2** – номера точек, через которые должна быть проведена линия.

### 2.2. *Tangent to Line – Касательная к линии*

Синтаксис:

**LTAN, <NLine>, <KeyPoint>, [<XV3>, <YV3>, <ZV3>]**

Описание аргументов команды:

**NLine** – номер линии, касательную к которой необходимо построить, знак числа определяет направление линии;

**KeyPoint** – номер точки, через которую требуется провести линию.

**XV3, YV3, ZV3** – компоненты вектора направления.

### 2.3. *Normal to Line – Нормаль к линии*

Синтаксис:

**LANG, <NLine>, <KeyPoint>, <Angle>, [<PhiT>, <Locat>]**

Описание аргументов команды:

**NLine** – номер линии, нормаль к которой необходимо построить;

**KeyPoint** – номер точки, в которой должна оканчиваться линия;

**Angle** – угол к заданной линии (задается в градусах): для нормали значение равно 90°.

**PhiT** – номер точки пересечения с нормалью;

**Locat** – координаты точки пересечения с нормалью, в относительных координатах линии.

#### 2.4. *Arcs – Дуга*

Синтаксис:

**LARC, <KeyPoint1>, <KeyPoint2>, <KeyPointC>, <Radius>**

Описание аргументов команды:

**KeyPoint1, KeyPoint2** – номер точек начала и конца дуги;

**KeyPointC** – номер точки, определяющей центр кривизны;

**Radius** – радиус кривизны.

#### 2.5. *Segmented spline – Сплайн*

Синтаксис:

**BSPLIN, <P1>, <P2>, <P3>, <P4>, <P5>, <P6>, <XV1>, <YV1>, <ZV1>, <XV6>, <YV6>, <ZV6>**

Описание аргументов команды:

**P1, P2, P3, P4, P5, P6** – номера точек, через которые необходимо провести сплайн;

**XV1, YV1, ZV1** – компоненты касательной к сплайну в первой точке;

**XV6, YV6, ZV6** – компоненты касательной к сплайну в последней точке.

#### 2.6. *Divide Line by Line – Разделение линии линией*

Синтаксис:

**LSBL, <Line1>, <Line2>, [<Sepor>, <Keep1>, <Keep2>]**

Описание аргументов команды:

**Line1** – линия, которая должна быть разделена;

**Line2** – линия, которая будет разделять;



**Sepo** – настройка точек: при пустом параметре в области пересечения для каждой линии будут общие точки;

**Keep1** – настройка удаления линии Line1: Delete – удалить, Keep – сохранить (по умолчанию);

**Keep2** – настройка удаления линии Line2: Delete – удалить (по умолчанию), Keep – сохранить.

## *2.7. Divide into N Lines – Разделение линии на несколько*

Синтаксис:

**LDIV, <NLine>, <Ratio>, <KeyPDiv>, <NDiv>, <Keep>**

Описание аргументов команды:

**NLine** – линия, которая должна быть разделена;

**Ratio** – отношение, в котором требуется разделить линию (отсчет по ориентации линии): если разделение происходит по точке, то значение устанавливается в ноль;

**KeyPDiv** – точка, разделяющая линии: если указано отношение, то поле является необязательным;

**NDiv** – количество новых линий (значение по умолчанию – 2);

**Keep** – настройка входных объектов: 0 – старые линии изменяются и используются новые точки, 1 – старые линии не используются, а новые накладываются на старые.

## *2.8. Delete lines – Удалить линии*

Синтаксис:

**LDELE, <Line1>, [<Line2>, <Inc>, <KeyPointDel>]**

Описание аргументов команды:

**Line1** – номер линии, с которой удалять;

**Line2** – номер линии, по которую удалять;

**Inc** – счётчик по номерам линий;

**KeyPointDel** – опция для удаления точек, совместных с линиями: 0 – удалить только линии, 1 – удалить точки, привязанные только к удаляемым линиям.

### 3. Node – Узел

Синтаксис:

**N**, [**<NNode>**], **<XNode>**, **<YNode>**, **<ZNode>**

Описание аргументов команды:

**NNode** – номер узла: можно не указывать, в этом случае номер присваивается автоматически;

**XNode**, **YNode**, **ZNode** – координаты узла.

### 4. Element – Элемент

Синтаксис:

**E**, **<I>**, **<J>**, **<K>**, **<L>**, **<M>**, **<N>**, **<O>**, **<P>**

Здесь **I**, **J**, **K**, **L**, **M**, **N**, **O**, **P** – локальные номера узлов КЭ (количество аргументов команды и их смысл определяются типом КЭ).

## Пример №1. Применение графических команд

Построим четыре точки, расположенные по углам единичного квадрата:

```
/prep7
K,, 0, 0, 0
K,, 1, 0, 0
K,, 1, 1, 0
K,, 0, 1, 0
```

Для отображения точек можно воспользоваться командой *Plot → Key-Points* (или *Plot → Multi-Plots*).

Для отображения номеров точек в графической области в меню *PlotCtrls → Numbering* необходимо включить флажок *KP Keypoint numbers*.

Для отображения списка точек: *List → Keypoint → Coordinates Only*.

LIST ALL SELECTED KEYPOINTS. DSYS= 0

NO.	X, Y, Z LOCATION			THXY, THYZ, THZX ANGLES		
1	0.000000	0.000000	0.000000	0.0000	0.0000	0.0000
2	1.000000	0.000000	0.000000	0.0000	0.0000	0.0000
3	1.000000	1.000000	0.000000	0.0000	0.0000	0.0000
4	0.000000	1.000000	0.000000	0.0000	0.0000	0.0000

Рис. 2.1. Результат вывода списка точек.

Построим дугу между точками 1 и 3, радиусом 1 и центром кривизны в точке 2:

```
LARC, 1, 3, 2, 1
```

Для отображения только линий необходимо воспользоваться командой *Plot* → *Lines*, для отображения точек совместно с линиями: *Plot* → *Multi-Plots*.

Добавим дугу между точками 1 и 3, радиусом 2 и центром кривизны в точке 2:

```
LARC, 1, 3, 2, 2
```

Построим прямую линию между точками 1 и 3:

```
LSTR, 3, 1
```

Для отображения номеров линий в графической области *PlotCtrls* → *Numbering* необходимо включить флажок *LINE Line numbers*.

Для отображения списка линий: *List* → *Lines*. В этом случае требуется уточнить необходимые для отображения атрибуты: *Attribute format* – вывод атрибутов линий, *Radius format* – вывод радиусов кривизны линий, *Layer-mesh format* – вывод слоев сеток, *Orientation KP* – вывод ориентации по точкам.

LIST ALL SELECTED LINES.

NUMBER	KEYPOINTS	LENGTH	(NDIV)	(SPACE)	KYND	NDIV	SPACE	#NODE	#ELEM	MAT	REAL	TYP	ESYS
1	1	3	1.571	0	1.000	0		0	0	0	0	0	0
2	1	3	1.445	0	1.000	0		0	0	0	0	0	0
3	3	1	1.414	0	1.000	0		0	0	0	0	0	0

Рис. 2.2. Результат вывода списка атрибутов линий.

LIST ALL SELECTED LINES.

NO.	KEYPOINTS	RADIUS
1	1 3	1.0000
2	1 3	2.0000
3	3 1	0.0000

Рис. 2.3. Результат вывода списка радиусов кривизны линий.

Построим нормаль к первой линии (дуга радиуса 1):

```
LANG, 1, 2, 90,,
```

Стоит отметить, что в этом случае линия, к которой строится нормаль, будет разделена на две, то есть результатом построения будет изменение топологии:

- 1) будет построена новая точка 5;
- 2) линией под номером 1 будет дуга между точками 1 и 5;
- 3) появится линия под номером 4 – дуга между точками 5 и 3;
- 4) появится линия (нормаль) между точками 5 и 2.

Таким образом, сначала задаются координаты необходимой точки, после чего назначается новая точка, номер которой определяется как следующий по возрастанию.

Далее реализуется разделение существующих линий, правило которого определяется их ориентацией (порядок точек). Таким образом, в нашем при-

мере дуга образована порядком точек 1-3, между которыми расположена точка 5, поэтому разделение дуги будет происходить по правилу: 1-5, 5-3 – в таком случае первая дуга сохранит номер порождающей линии, а второй дуге будет присвоен первый свободный номер (так как на этот момент было создано 3 линии, то первым свободным номером окажется 4). После создания вспомогательных точек и линий происходит построение искомой нормали – линии между точками 5 и 2, которой будет присвоен свободный номер (так как на этот момент уже создано 4 линии, то свободным номером будет 5).

Теперь построим нормаль к третьей построенной линии (диагональная прямая):

```
LANG, 3, 2, 90, ,
```

В этом случае линия также будет разделена на две, но номер сохранит линия между точками 3 и 6, так как ориентация первоначальной прямой определена точками 3-1.

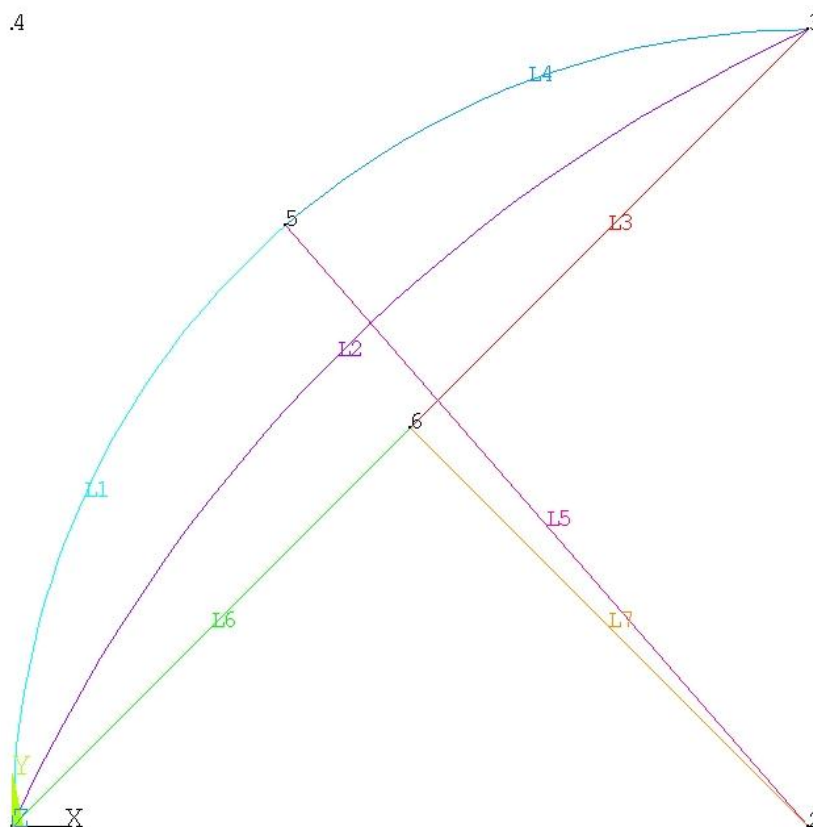


Рис. 2.4. Результаты построения.

Несмотря на то, что линия 2 (L2 на рис. 2.4) визуально пересекается с линией 5 (L5 на рис. 2.4), фактически, то есть в топологии ANSYS, этого пересечения нет. Построим это пересечение с сохранением разделяющей линии – для этого используем следующую команду:

```
LSBL, 2, 5, , , Keep
```

В результате выполнения этой команды в точке пересечения линий 2 и 5 появляется точка, после чего линия 2 исчезает, и на её месте появляются две линии – 8 и 9; линия 5, пересечение которой производится, остается без изменений.

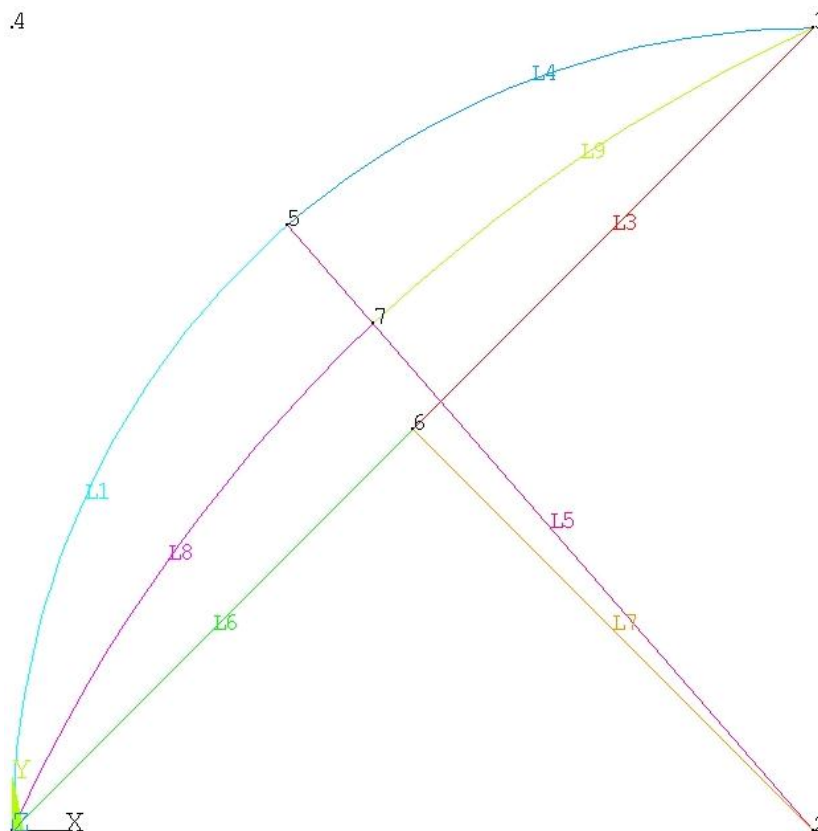


Рис. 2.5. Результат разделения линий.

Посмотрим ориентацию вновь построенных линий: для этого выведем список ориентации линий (см. рис. 2.6).

LIST ALL SELECTED LINES.

NO.	KEYPOINTS		KPB	KPE	SECT
1	1	5			0
3	3	6			0
4	5	3			0
5	5	2			0
6	6	1			0
7	6	2			0
8	1	7			0
9	3	7			0

Рис. 2.6. Ориентация полученных линий.

Разделим линию 5 по точке 7:

```
LDIV, 5, 0, 7
```

В этом случае на месте линии 5 появляются две: линия 5 (номер сохраняется по ориентации первоначальной линии: точки 5-2), вторая линия получает свободный номер: в нашем случае это номер 2.

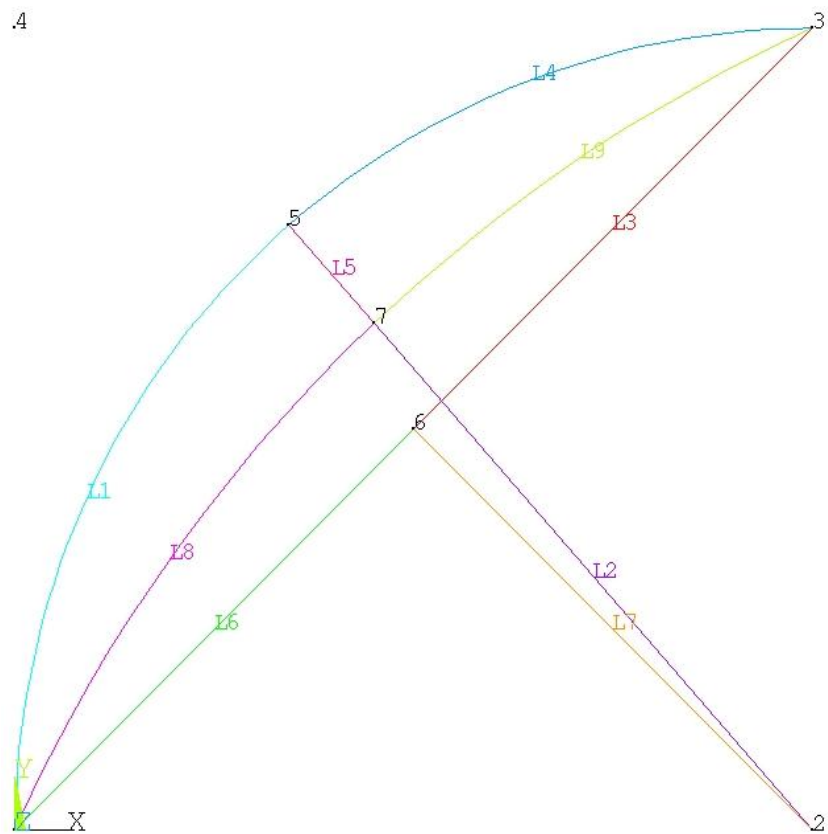
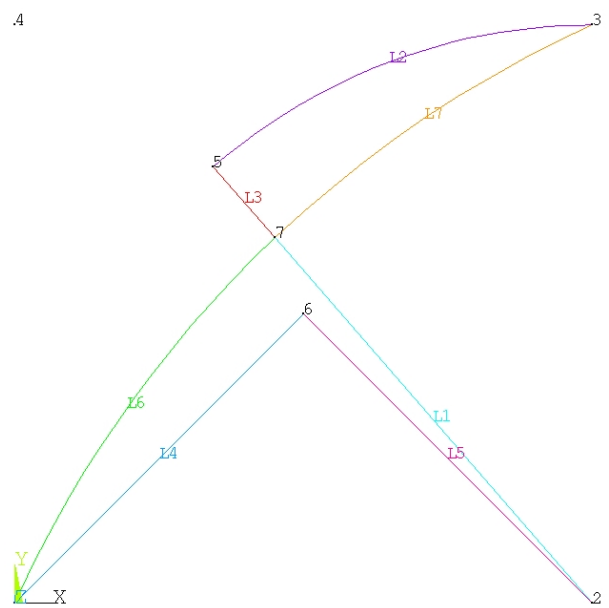
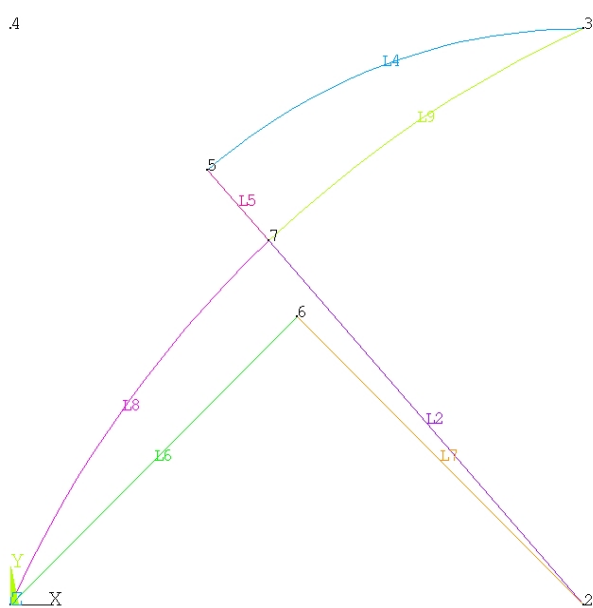


Рис. 2.7. Результат разделения линий.

Удалим линию 1 и 3:

```
LDELETE, 1
LDELETE, 3
```





а

б

Рис. 2.8. Результат: а – линии с нумерацией после удаления;  
б – линии с исправленной нумерацией.

Для восстановления нумерации линий можно использовать команду:

```
NUMCMP, LINE
```

Рассмотрим построение касательных к линии 2:

```
LTAN, -2, 1  
LTAN, 2, 7
```

Так как ориентация линии 2 определена точками 5-3, то первая команда определяет обратную ориентацию линии 3-5 (знак минус перед номером линии), поэтому касательная кривая выходит из точки 5, а вторая линия определяется прямой ориентацией и исходит из точки 3.

После удаления линий 7 и 3 получим фигуру на рис. 2.9:

```
LDELE, 7, , , 1  
LDELE, 3, , , 1
```

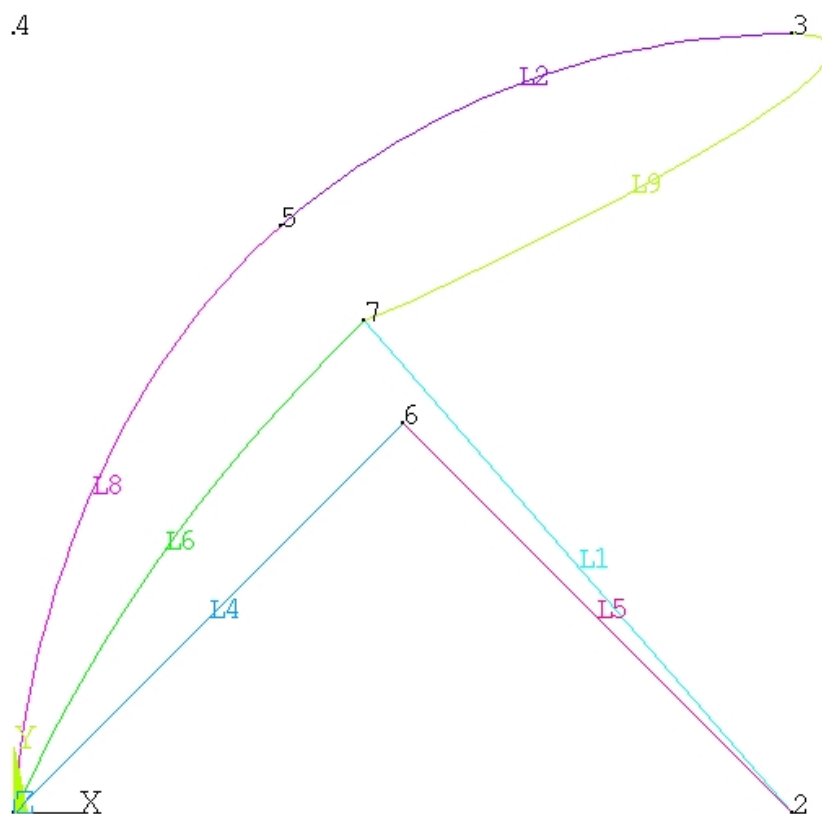


Рис. 2.9. Результат разделения линий.

Полный листинг программы:

```
/CWD, <DirName>
/prep7
K,, 0, 0, 0
K,, 1, 0, 0
K,, 1, 1, 0
K,, 0, 1, 0
LARC, 1, 3, 2, 1
LARC, 1, 3, 2, 2
LSTR, 3, 1
LANG, 1, 2, 90,,
LANG, 3, 2, 90,,
LSBL, 2, 5, , , Keep
LDIV, 5,0,7
LDELE, 1,,1
LDELE, 3,,1
NUMCMP,LINE
LTAN, -2, 1
LTAN, 2, 7
LDELE, 7,,1
LDELE, 3,,1
```

### Пример №2. Построение параболы $x=y^2/2$

Построим параболу  $x=y^2/2$  Вначале включаем препроцессор:

```
/prep7
```

Далее строим точки. Пусть имеется десять точек – в таком случае необходимо создать цикл на десять итераций. Координаты точек будут определены по формуле:

```
*DO,i,1,10
  K,,i*i/2,i,0
*ENDDO
```

Теперь необходимо соединить точки попарно: для этого создадим цикл на 9 итераций: линия будет соединять соседние точки по их номерам.

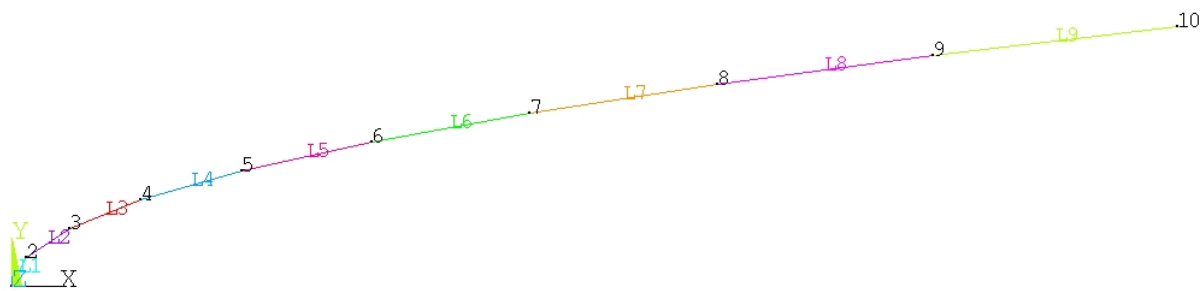


Рис. 2.10. Результат разделения линий.

Код можно обобщить, введя переменную N, определяющую необходимое количество точек:

```
N=10
*DO, i, 0, N-1
  K, , i*i/2, i, 0
*ENDDO
*DO, i, 1, N-1
  LSTR, i, i+1
*ENDDO
```

### Пример №3. Построение функции $y=\sin(x)$

Попробуем построить график функции  $y=\sin(x)$ . Прямая подстановка функции  $\sin(i)$  в аргумент точки приведет к результату, но удобнее использовать буферную переменную:

```
N=20
*DO, i, 0, N-1
  y=sin(i)
  K, , i, y, 0
*ENDDO
*DO, i, 1, N-1
  LSTR, i, i+1
*ENDDO
```

Полученный график имеет ярко выраженный ломаный вид (рис. 2.11), далёкий от необходимого.

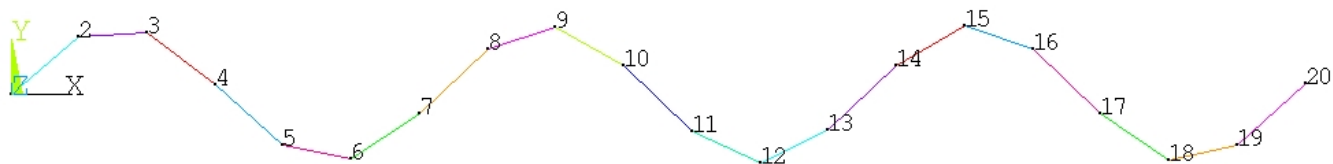


Рис. 2.11. Результат построения функции  $\sin(x)$  ломанными.

С целью получения большей гладкости воспользуемся функцией сплайна – будем проводить его через четыре точки. В таком случае код можно обобщить следующим способом:

```
BSPLIN, 1, 2, 3, 4
*DO, i, 1, N/4-1
  BSPLIN, 4*i, 4*i+1, 4*i+2, 4*i+3, 4*i+4
*ENDDO
```

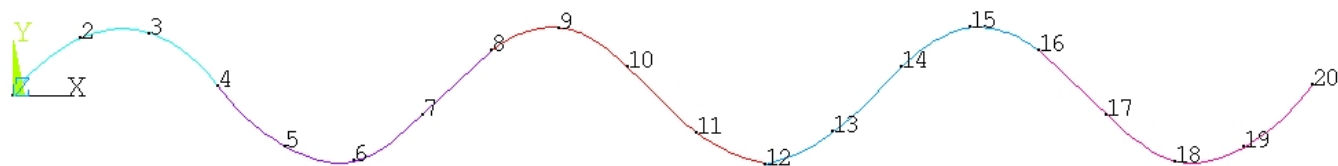


Рис. 2.12. Результат построения функции  $\sin(x)$  сплайнами.

Можно попробовать тот же код, но в другой СК, например, в цилиндрической. Для этого необходимо ввести команду:

```
CSYS, 1
```

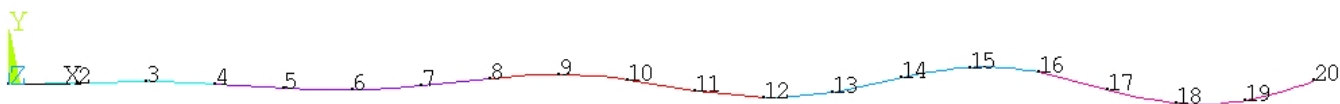


Рис. 2.13. Результат построения функции  $\sin(x)$  сплайнами в цилиндрической системе координат.

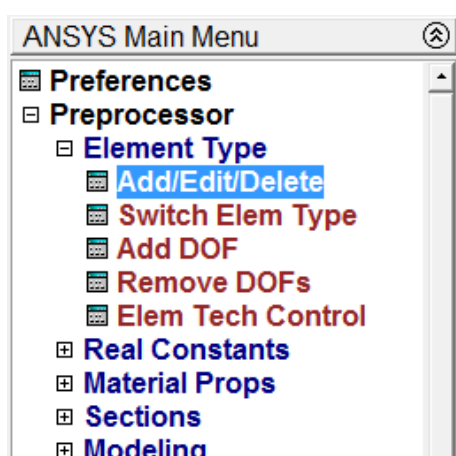
## Задание конечного элемента (Element type)

Добавление КЭ производится в главном меню препроцессора *Element Type*.

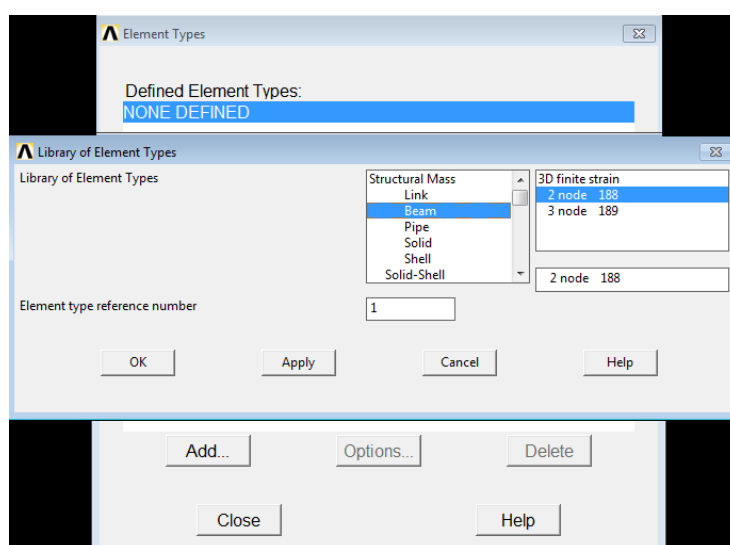
В возникающем меню нажатием клавиши *Add* (Добавить) можно вызвать список элементов (см. рис. 2.14).

Доступ через графический интерфейс:

*Preprocessor* → *Element Type* → *Add/Edit/Delete*.



а



б

Рис. 2.14. Параметры КЭ.

Такой же результат можно получить с помощью следующей команды:

**ET, <ETNum>, <ETName>**

Описание аргументов команды:

**ETNum** – порядковый номер типа КЭ: в последующем обращение к типу будет производиться по этому номеру;

**ETName** – имя КЭ согласно номенклатуре ANSYS.

Рассмотрим некоторые одномерные КЭ, представленные в ANSYS.

## Конечный элемент BEAM188

BEAM188 – одномерный балочный конечный элемент. Это двухузловой КЭ с линейной, квадратичной или кубической аппроксимацией. Каждый узел обладает шестью степенями свободы: три перемещения в направлениях  $x$ ,  $y$ ,  $z$  и три угла поворота вокруг осей  $Ox$ ,  $Oy$ ,  $Oz$ .

Командой этот КЭ добавляется так:

```
ET, 1, BEAM188
```

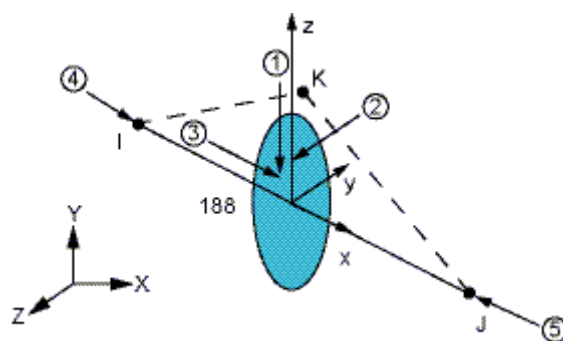
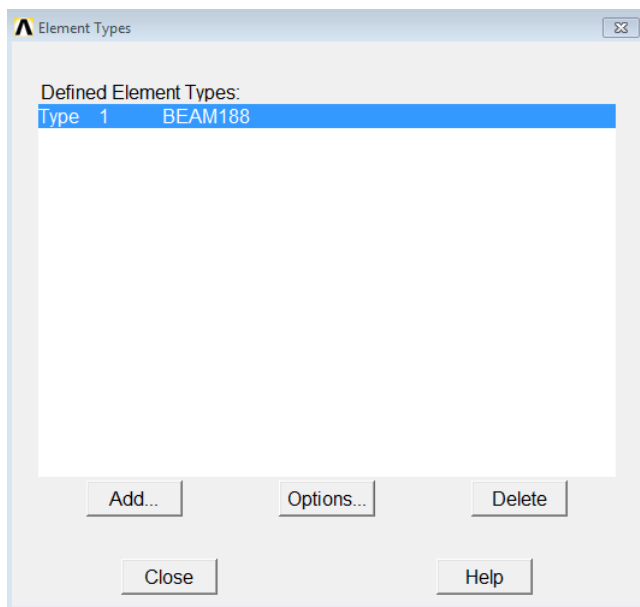


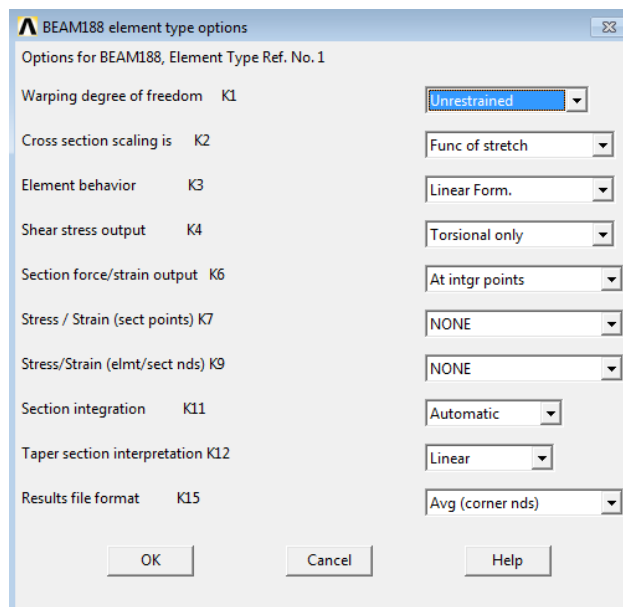
Рис. 2.15. Схема КЭ.

На рис. 2.15 приведена стандартная иллюстрация к геометрии КЭ (крайние узлы принято обозначать I, J) и локальная СК, связанная с ним. Узел K предназначен для определения ориентации локальной СК, таким образом, узлы IJK определяют плоскость  $xOy$ . Цифрами в кружках на рис. 2.15 указаны возможные направления прикладываемого давления.

Настройки КЭ можно вызвать в меню *Element Type* (Тип элемента): для этого требуется выбрать необходимый КЭ и нажать кнопку *Options* (Параметры) – см. рис. 2.16.



а



б

Рис. 2.16. Параметры КЭ.

Эти же параметры можно задать с помощью специальной команды

**KEYOPT, <ElemTypeNum>, <NumKeyOpt>, <ValKeyOpt>**

Описание аргументов команды:

**ElemTypeNum** – номер типа КЭ, для которого определяют настройки;

**NumKeyOpt** – номер параметра (KEYOPT);

**ValKeyOpt** – значение параметра (KEYOPT).

Все возможные настройки для КЭ описаны в документации к ANSYS.

Рассмотрим в качестве примера KEYOPT(3). На рис. 2.17 приведены данные из документации: KEYOPT(3) регулирует порядок функции формы по длине КЭ и имеет три возможных значения: 0 – первый порядок или линейная аппроксимация (по умолчанию), 2 - второй порядок или квадратичная аппроксимация, 3 - третий порядок или кубическая аппроксимация.

### KEYOPT(3)

Shape functions along the length:

0 -- Linear (default)

2 -- Quadratic

3 -- Cubic

Рис. 2.17. Параметры для KEYOPT(3).

Так как выбранный нами КЭ имеет порядковый номер 1, то это значение будет первым аргументом команды KEYOPT; вторым аргументом является номер параметра, – в нашем случае 3, и последний аргумент – значение параметра: 0, 2 или 3. Ниже приведены команды включения по очереди всех возможных значений:

```
KEYOPT,1,3,0  
KEYOPT,1,3,2  
KEYOPT,1,3,3
```

Для моделирования поведения механической системы с помощью этого КЭ необходимо задать материал, тип и характеристики сечения.

### Конечный элемент LINK180

КЭ состоит из двух узлов и имеет три степени свободы в каждом узле: перемещения в направлениях  $x$ ,  $y$ ,  $z$ . Данный тип КЭ используется для моделирования одноосного растяжения/сжатия. Может работать только на сжатие или только на растяжение, но для этого необходимо включение нелинейного итерационного решателя.

Настройка **KEYOPT(2)** – изменение поперечного сечения: 0 – включено (по умолчанию): работает при включенной нелинейной геометрии; 1 – выключено.



Входными данными для КЭ являются: площадь поперечного сечения (AREA), распределенная масса (ADDMAS) и опции работы КЭ: -1 – работа на сжатие, 1 – растяжение или совместное растяжение со сжатием – 0.

Вид работы включается через задание постоянных КЭ (см. параграф ниже – Задание постоянных элемента). Для моделирования поведения механической системы этим КЭ необходимо задать материал и постоянные элемента.

### Конечный элемент CERIG

Рассмотрим паучьи сетки (spider net): такие сетки состоят из мастер-узла (master node) и подчиненных узлов (slave nodes). Кинематические и статические ГУ накладываются на мастер-узел, а на подчиненные узлы перераспределяются согласно определённому правилу. Рассмотрим пример такого КЭ, предназначенный для передачи сил и моментов.

Синтаксис:

**CERIG, <MASTER>, <SLAVE>, <Ld>, [<Ld2>, <Ld3>, <Ld4>, <Ld5>]**

Описание аргументов команды:

**MASTER** – номер мастер-узла;

**SLAVE** – номера подчиненных узлов;

**Ld, ..., Ld5** – степени свободы, связывающие мастер-узел и подчиненные узлы.

Использование паучьих сеток удобно для передачи усилий или формирования кинематических ГУ, эквивалентных шарнирным соединениям.

### Задание материала (Material props)

Доступ через графический интерфейс:

*Preprocessor → Material Props → Material Models.*

Определение таблицы температур для свойств материалов реализуется с помощью команды

**MPTEMP, <StartLoc>, <T1>, <T2>, <T3>, <T4>, <T5>, <T6>**

Описание аргументов команды:

**StartLoc** – начальная позиция в таблице;

**T1, T2, T3, T4, T5, T6** – значения температур.

Для задания типов и значений механических характеристик используется следующая команда:

**MPDATA, <MatName>, <NMat>, <StartLoc>, <C1>, <C2>, <C3>, <C4>, <C5>, <C6>**

Описание аргументов команды:

**MatName** – имя характеристики;

**NMat** – номер материала: используется в дальнейшем во всех ссылочных контекстах;

**StartLoc** – начальная позиция в таблице для ввода значения;

**C1, C2, C3, C4, C5, C6** – значения характеристик.

Например, для задания линейно-упругого материала необходимо определить две характеристики: модуль Юнга и коэффициент Пуассона. С этой целью создаётся таблица, состоящая из одного элемента со значением температуры 0. Далее задаются типы и значения упомянутых физических свойств. Модуль Юнга имеет специальное имя EX, коэффициент Пуассона – PRXY.

Ниже приведен код, вводящий модуль Юнга =  $210 \cdot 10^3$  МПа, коэффициент Пуассона = 0.3:

```
MPTEMP, 1, 0  
MPDATA, EX, 1, , 210000  
MPDATA, PRXY, 1, , 0.3
```

Стоит отметить вопрос размерностей физических величин. В ANSYS в задачах определения НДС вводимые механические характеристики и геометрические размеры не имеют физической размерности. За этим требуется следить самостоятельно: можно вводить все параметры «метр – Ньютон – Паскаль», но на практике удобнее использовать систему «миллиметр – Ньютон – МегаПаскаль».

## Задание постоянных элемента (Real constant)

Доступ через графический интерфейс:

*Preprocessor* → *Real constant* → *Add/Edit/Delete*

или командой

**R, <NReal>, <Val1>, <Val2>, <Val3>**

Описание аргументов команды:

**NReal** – номер постоянных;

**Val1, ..., Val3** – значения постоянных.

Для КЭ LINK180 необходимо задать:

**Val1** – площадь поперечного сечения (AREA);

**Val2** – распределенная масса (ADDMAS);

**Val3** – параметры работы КЭ (TENSKEY).

## Задание сечения (Section)

Доступ через графический интерфейс:

*Preprocessor* → *Sections* → *Beam* → *Common Sections*.

Помимо механических свойств балочным КЭ необходимо определить поперечное сечение, его вид и геометрические характеристики. Для задания типа сечения необходимо воспользоваться командой

**SECTYPE, <NumSec>, <Type>, <SecType>, <Name>, <Refinekey>**

Описание аргументов команды:

**NumSec** – индикационный номер сечения;

**Type** – тип сечения, связанный с типом КЭ;

**SecType** – тип сечения для заданного типа КЭ;

**Name** – название сечения;

**Refinekey** – параметр КЭ сетки для тонкостенных сечений.

Для задания смещения сечения относительно нуля локальной СК элемента необходимо использовать команду

**SECOFFSET**, <Location>, <Offset1>, < Offset2>, <CG-Y>, <CG-Z>, <SH-Y>, <SH-Z>

Значения аргументов зависят от типа КЭ. Рассмотрим значения аргументов для балочного КЭ BEAM188 (на рис. 2.18 приведен пример сечения, связанный с СК).

Описание аргументов команды:

**Location** – расположение поперечного сечения: CENT – центр масс поперечного сечения лежит на продольной оси КЭ (по умолчанию), SHRC – центр сдвига поперечного сечения лежит на продольной оси КЭ, ORIGIN – нуль СК поперечного сечения лежит на продольной оси КЭ, USER – смещение сечения по заданным величинам;

**Offset1, Offset2** – задание величин смещения (если задан тип USER);

**CG-Y, CG-Z** – задание пользовательских величин центраоида;

**SH-Y, SH-Z** – задание пользовательских величин центра сдвига;

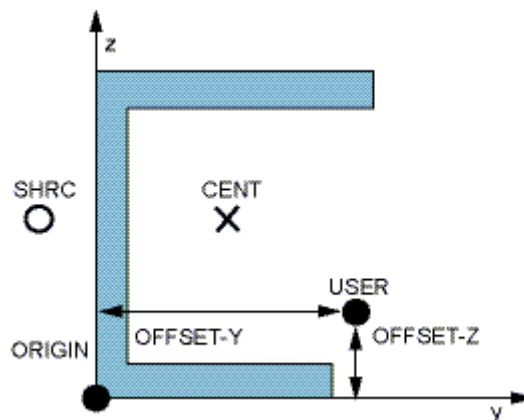


Рис. 2.18. Пример сечения и локальная ось конечного элемента.

Для задания геометрии поперечного сечения необходимо использовать команду

**SECDATA**, <VAL1>, <VAL2>, ..., <VAL11>, <VAL12>

Здесь значения **VAL1, ..., VAL12** определяют геометрию и КЭ-разбиение поперечного сечения, их смысл зависит от типа поперечного сечения.

Чтобы внести некоторую ясность, рассмотрим пример с прямоугольным поперечным сечением: в случае других типов рекомендуется изучение документации. На рис. 2.19 представлен вид прямоугольного сечения с параметрами из документации.

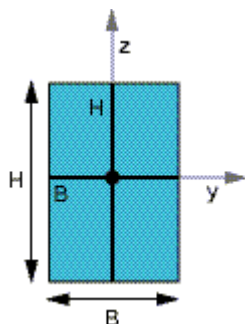


Рис. 2.19. Вид прямоугольного сечения.

Для этого типа сечения необходимо задать следующие параметры: **B, H, Nb, Nh**, где:

**B, H** – геометрические размеры: ширина и высота соответственно (см. рис. 2.19);

**Nb, Nh** – количество ячеек по соответствующим геометрическим размерам (по умолчанию равно 2).

Ниже приведен листинг, задающий тип сечения под номером 1 и именем MyName, тип КЭ, для которого задается сечение (BEAM), тип сечения (Rect): поперечное сечение определено центроидом, размер прямоугольника – 2 x 3, разбиение 5 x 5:

```
SECTYPE, 1, BEAM, RECT, MyName, 0  
SECOFFSET, CENT  
SECDATA, 2, 3, 5, 5
```

Для просмотра всех введенных сечений:

*Preprocessor* → *Sections* → *Beam* → *Plot Section*.

На рис. 2.20 приведена иллюстрация результата этих действий для введенного сечения. Справа от сечения приводятся его рассчитанные характеристики: площадь, компоненты тензора инерции и другие параметры.

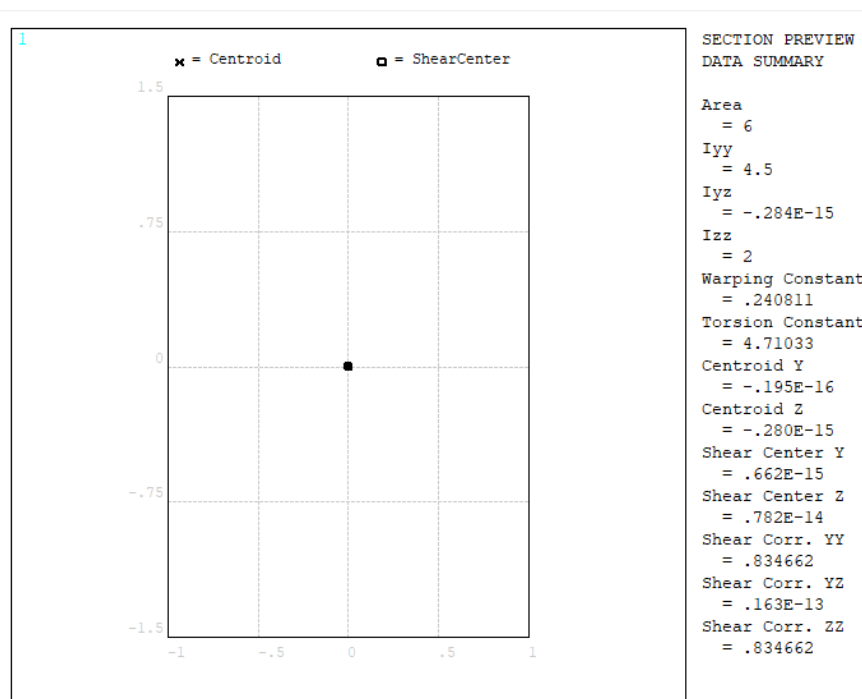


Рис. 2.20. Вид прямоугольного сечения и его разбиение.

### Привязка материала и сечения

Для привязки атрибутов КЭ (материал, сечение и т.п.) к линии используется команда

**LATT, <MAT>, <REAL>, <ELEM>, -, <KB>, <KE>, <SECNUM>**

Описание аргументов команды:

**MAT, REAL, ELEM, SECNUM** – номера: материала, постоянных КЭ, КЭ, сечения;

**KB, KE** – начальная и конечная точки, определяющие линию и ориентацию сечения.

## Разбиение на сетку (Meshing)

Для разбиения геометрии на одномерные КЭ используется команда

**LMESH, <NLine1>, [<NLine2>, <Inc>]**

Описание аргументов команды:

**NLine1, NLine2** – номера линий, которые необходимо разбить на КЭ-сетку, можно также использовать зарезервированное слово **ALL**, которое обозначает все линии;

**Inc** – шаг счётчика, если указан диапазон линий.

При разбиении линий тип КЭ, материал, сечение, постоянные и т.п. будут использованы согласно привязкам. Если с линиями не связаны какие-либо параметры, то будут использованы активные настройки.

Рассмотрим следующие команды:

- задание типа КЭ

**TYPE, <NElem>**

где **NElem** – номер типа КЭ;

- задание типа материала

**MAT, <NMat>**

где **NMat** – номер материала;

- задание типа поперечного сечения

**SECNUM, <NSecNum>**

где **NSecNum** – номер поперечного сечения;

- задание постоянных КЭ

**REAL, <NRealNum>**

где **NRealNum** – номер постоянных КЭ;

- задание СК элементов

**ESYS, <NCS>**

где **NCS** – номер СК: 0 – используется правило ориентации СК элемента (по умолчанию), в противном случае – номер СК.

С точки зрения получаемых результатов принципиальным является качество КЭ сетки, определяемое типом и размерами КЭ. Для одномерных КЭ важнее размер. Для разметки линий используется команда

**LESIZE**, <NLine>, <Size>, <AngSize>, <Ndiv>, <Space>, <KFORC>, <Layer1>, <Layer2>, <KynDiv>

Описание аргументов команды:

**NLine** – номер линии или зарезервированное слово **ALL**;

**Size** – линейный размер КЭ;

**AngSize** – угловой размер (в градусах) КЭ;

**Ndiv** – количество КЭ;

**Space** – величина относительного изменения размера КЭ;

**KFORC** – настройка применения параметров разбиения для набора линий (когда задано зарезервированное слово **ALL**), может принимать значения: 0 – параметры разбиения применяются только к линиям, у которых не задана разметка; 1 – параметры разбиения применяются ко всем линиям; 2 – параметры разбиения применяются только к линиям, у которых заданная величина КЭ меньше актуальной; 3 – параметры разбиения применяются только к линиям, у которых заданная величина КЭ больше актуальной;

**Layer1, Layer2** – настройка разбиения слоев для двумерных КЭ;

**KynDiv** – 0, NO или OFF – авторазбиение не может изменять ручную настройку разбиения; 1, YES или ON – авторазбиение может изменять ручную настройку разбиения.

Настроим и разобьем на КЭ-сетку построенную линию: для этого привяжем тип КЭ, материал, поперечное сечение, СК, разметим разбиение линии на десять КЭ и проведем разбиение линии под номером 1:

```
TYPE, 1
MAT, 1
SECNUM, 1
ESYS, 0
LESIZE, 1, , , 10, , , , 0
```



```
LMESH, 1
```

Для большей наглядности создадим балку длиной 100 мм прямоугольного поперечного сечения 10 мм на 5 мм, материал – сталь, модуль Юнга –  $210 \cdot 10^3$  МПа, коэффициент Пуассона – 0.3.

Листинг программы:

```
/CWD,<SomeDir>  
/prep7  
CSYS,0
```

```
! зададим параметры: L – длина балки;  
! a, b – размеры поперечного сечения  
! N – количество КЭ  
N=10  
L=100  
a=10  
b=5
```

```
! Зададим точки и проведем линию  
K,,0,0,0  
K,,L,0,0  
LSTR,1,2
```

```
!выберем тип КЭ и его настройки  
ET,1,BEAM188  
KEYOPT,1,3,0
```

```
! зададим свойства материала  
MPTEMP,1,0  
MPDATA,EX,1,,210000  
MPDATA,PRXY,1,,0.3
```

```
! зададим тип и размеры поперечного сечения  
SECTYPE, 1, BEAM, RECT, MyName, 0  
SECOFFSET, CENT  
SECDATA,10,5,3,3
```

```
! привяжем тип КЭ, материал, поперечное сечение  
! разметим линию для разбиения и разобьем ее  
TYPE, 1  
MAT, 1
```

```

SECNUM, 1
ESYS, 0
LESIZE, 1, , , N, , , , , 0
LMESH, 1

```

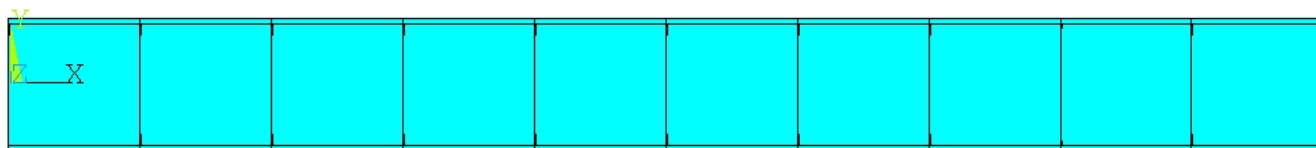
Результат разбиения линии приведен на рисунке 2.21. Для отображения поперечного сечения необходимо в *PlotCtrls* → *Style* → *Size and Shape* установить флажок в поле *[/ESHAPE] Display of element* или использовать команду: */ESHAPE, 1.0*. Пример такого отображения приведен на рис. 2.21в.



а



б



в

Рис. 2.21. Разбиение линии, с указанием номеров узлов:

а - номера узлов, б – номера КЭ, в – поперечное сечение.

Рассмотрим подробнее команду

**/ESHAPE, <SCALE>, [<KEY>]**

Описание аргументов команды:

**SCALE** – масштаб отображения: 0 – не отображать (по умолчанию);

**KEY** – настройка для оболочных КЭ (см. документацию).

## Изменение атрибутов элемента

Ориентация сечения определяется локальной СК, для изменения которой можно воспользоваться графическим интерфейсом:

*Preprocessor* → *Modeling* → *Move/Modify* → *Elements* → *Modify Nodes*

или командой

**EMODIF, <nElem>, <nNode>, <I1>, [<I2>, ... , <I8>]**

Описание аргументов команды:

**nElem** – номер КЭ;

**nNode** – номер узла в локальной нумерации КЭ или тип атрибута КЭ:

MAT – материал, TYPE – тип КЭ, ESYS – СК, или SECNUM – сечение;

**I1** – номер узла в глобальной нумерации, на который необходимо заменить заданный локальный номер узла или номер указанного атрибута, который необходимо задать;

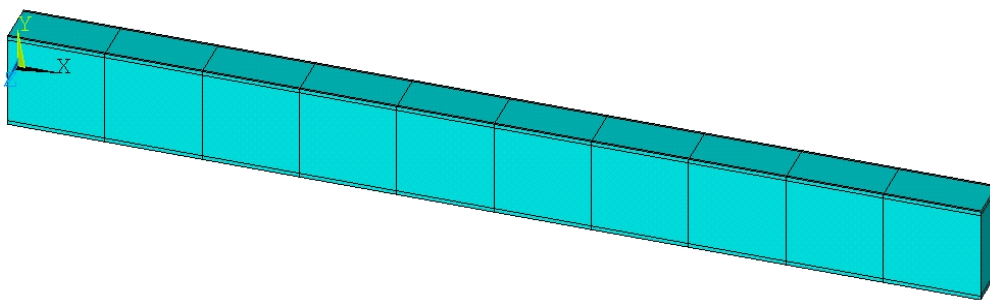
**I2, ..., I8** – номер узла в глобальной нумерации, на который необходимо заменить следующие заданные локальные номера узлов (каждый параметр увеличивается на единицу, начиная с **I1**; если изменению подвергается атрибут, эти аргументы не используются).

Например, можно поменять ориентацию локальной СК элемента и, как следствие, сечения. Для этого создадим удалённый узел и определим его как узел К (см. рис. 2.15). На рис. 2.22а приведена иллюстрация результата.

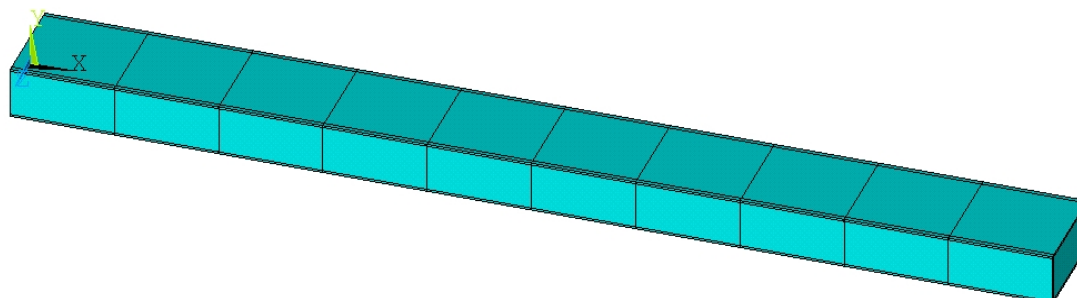
```
N, 500, 0, 0, 900  
EMODIF, ALL, 3, 500
```

Можно создать еще один узел, который, вернет локальную СК в первоначальную. На рис. 2.22б приведена иллюстрация результата:

```
N, 600, 0, 900, 0  
EMODIF, ALL, 3, 600
```



а



б

Рис. 2.22. Ориентация сечения.

### Объединение объектов

Зачастую возникает задача объединения объектов, которые структурно являются разными, но геометрически или физически близки, например, два узла или точки с одними и теми же координатами. Для решения таких задач предназначена команда

**NUMMRG, <Label>, <TOLER>, <GTOLER>, <Action>, <Switch>**

Описание аргументов команды:

**Label** – тип объекта: NODE – узел, ELEM – элемент, KP – точка, MAT – материал, TYPE – тип элемента, REAL – постоянные элемента, CP – набор кинематических пар, CE – выражение связей, ALL – все объекты;

**TOLER** – точность для критерия объединения: для объектов NODE и KP оценивается наибольшая разница по координатам, по умолчанию точность –  $10^{-4}$ ; для объектов MAT, REAL, CE оценивается относительное отличие между значениями, по умолчанию –  $10^{-7}$ ;

**GTOLER** – точность для объемных тел: используется для точек, присоединенных к линиям;

**Action** – определяет объединять или выбирать объекты: **SELE** – выбирает объекты; <пустое> – объединяет объекты в один;

**Switch** – правило нумерации для объединенного объекта: **LOW** – присваивается меньший номер (по умолчанию), **HIGH** – присваивается больший номер.

Рассмотрим пример: создадим четыре точки, соединенные попарно линиями. Пусть две из них имеют одинаковые координаты:

```
/PREP7  
K,,0,0,0  
K,,1,0,0  
K,,1,0,0  
K,,2,0,0  
L,1,2  
L,3,4
```

Объединим центральные точки в одну:

```
NUMMRG, KP
```

В этом случае вместо двух точек 2 и 3 останется только одна с номером 2 (правило нумерации по умолчанию). Стоит отметить, что в этом случае обе линии будут иметь общую точку.

## ГЛАВА 3. ПРОЦЕССОР

В первую очередь необходимо выбрать тип решаемой задачи в графическом интерфейсе:

*Solution* → *Analysis Type* → *New Analysis*

или задать командой

**ANTYPE, <SolType>**

где **SolType** – число, определяющее тип решаемой задачи:

0 – *Static* – статическая задача;

2 – *Modal* – задача на собственные значения;

3 – *Harmonic* – гармонический анализ (ответ на гармоничное воздействие);

4 – *Transient* – переходные задачи (задачи динамики);

8 – *Spectrum* – спектральный анализ (анализ реакции на воздействие);

1 – *Eigen Buckling* – задача потери устойчивости;

7 – *Substructuring* – подконструкция (суперэлементный метод).

Тип решаемой задачи накладывает условия на необходимые типы материалов, ГУ и т.п. Разные типы задач имеют различные методы их решения и соответствующие наборы параметров, для задания которых можно воспользоваться графическим интерфейсом:

*Solution* → *Analysis Type* → *Analysis Options/Sol'n Controls*

(название меню может варьироваться для различных типов задач). Одной команды, которая регулирует эти настройки, нет. Приводить весь список здесь нецелесообразно: при необходимости следует самостоятельно изучить документацию. Стоит отметить, что в зависимости от типа решаемой задачи возникает необходимость в задании определенных свойств (моделей) материалов. Так, например, если в задаче определения НДС при отсутствии массовых сил плотность не нужна, то при определении собственных форм колебаний

конструкции задание плотности обязательно. В рамках настоящего пособия рассматриваются статические линейные задачи механики твердого деформированного тела.

### Массовые силы

Для задания массовых сил необходимо в первую очередь задать плотность материала, после чего указать направление и величину ускорения. Для задания ускорения в графическом интерфейсе выбрать

*Preprocessor → Loads → Define Loads → Apply → Structural → Inertia* или *Solution → Define Loads → Apply → Structural → Inertia*, где будет приведен список возможных нагрузок: угловая скорость (OMEGA), ускорение (DOMEGA), гравитация (ACEL) и прочее.

### Температура

Приложение температуры на линиях через графический интерфейс:

*Preprocessor → Loads → Define Loads → Apply → Structural → Temperature → On Lines* или *Solution → Define Loads → Apply → Structural → Temperature → On Lines*

или командой

**BFL, <NLine>, <Lab>, <Val1>, [<Val2>, <Val3>, <Val4>]**

Описание аргументов команды:

**NLine** – номер линии;

**Lab** –тип нагрузки: в случае приложения температуры – TEMP;

**Val1, ..., Val4** – значения прикладываемой температуры.

Стоит отметить, что для появления температурных деформаций необходимо указать в свойствах материала коэффициент температурного расширения.

### Граничные условия

Граничные условия могут быть заданы как в препроцессоре, так и в процессоре и допускают изменение в ходе решения. Они могут быть приложены

как к геометрическим объектам (точка, линия, площадь) так и к узлам КЭ-сетки. В действительности любые ГУ, прикладываемые к геометрическим объектам, перед решением пересчитываются программой и прикладываются к узлам КЭ-сетки (которые относятся к заданному геометрическому объекту). На практике зачастую удобнее прикладывать ГУ к геометрическим объектам, что позволяет не привязываться к номерам узлов, которые могут быть изменены при переопределении параметров КЭ-сетки или геометрии.

### Кинематические граничные условия

Кинематические ГУ накладывают ограничения на перемещения.

Определение кинематических ГУ в *точке* через графический интерфейс:

*Preprocessor* → *Loads* → *Define Loads* → *Apply* → *Structural* → *Displacement* → *On Keypoints* или *Solution* → *Define Loads* → *Apply* → *Structural* → *Displacement* → *On Keypoints*

или командой

**DK**, <KPNum>, <Lab>, <Val>, <Val2>, <KExp>, <Lab2>, <Lab3>, <Lab4>, <Lab5>, <Lab6>

Описание аргументов команды:

**KPNum** – номер точки или ALL – все точки из выборки;

**Lab** – допустимая ссылка на тип степени свободы (например, UX, UY, ROTZ и т.п.);

**Val** – значение для выбранной степени свободы: если значение определяет массив, то имя необходимо выделять символами % (например, %dimValue%);

**Val2** – второе значение для выбранной степени свободы в случае, когда это допустимо типом решаемой задачи и степенью свободы;

**KExp** – расширенная настройка: 0 – кинематические ограничения применяются к узлу в заданной точке, 1 – отметить точку для определения кинематических ГУ;



**Lab2, ..., Lab6** – дополнительные ссылки на типы степеней свободы.

Определение кинематических ГУ в узле через графический интерфейс:

*Preprocessor → Loads → Define Loads → Apply → Structural → Displacement → On Nodes* или *Solution → Define Loads → Apply → Structural → Displacement → On Nodes*

или командой

**D, <Nnode>, <Lab>, <Val>, <Val2>, <Nend>, <Ninc>, <Lab2>, <Lab3>, <Lab4>, <Lab5>, <Lab6>**

Описание аргументов команды:

**Nnode** – номер узла или ALL – все узлы из выборки;

**Lab** – допустимая ссылка на тип степени свободы (например, UX, UY, ROTZ и т.п.);

**Val, Val2** – первое и второе (если допустимо) значения для выбранной степени свободы: если значение определяет массив, то имя необходимо выделять символами %;

**Nend, Ninc** – определяет последний номер узла и шаг счётчика (если необходимо выбрать узлы, пронумерованные по порядку);

**Lab2, ..., Lab6** – дополнительные ссылки на типы степеней свободы.

#### Статические граничные условия

Статические ГУ представляют собой поверхностные нагрузки: сила, момент или давление.

Определение статических ГУ в точке через графический интерфейс:

*Preprocessor → Loads → Define Loads → Apply → Force/Moment → On Keypoints* или *Solution → Define Loads → Apply → Force/Moment → On Keypoints*

или командой

**FK, < KPNum>, <Lab>, <Val>, <Val2>**

Описание аргументов команды:

**KPNum** – номер точки или ALL – все точки из выборки;

**Lab** – допустимая ссылка на тип силы (например, FX, MY, HEAT и т.п.);

**Val, Val2** – первое и второе (если допустимо) значения для выбранной степени свободы: если значение определяет массив, то имя необходимо выделять символами %.

Определение статических ГУ в узле через графический интерфейс:

*Preprocessor → Loads → Define Loads → Apply → Force/Moment → On Nodes* или *Solution → Define Loads → Apply → Force/Moment → On Nodes*  
или командой:

**F, <NNode>, <Lab>, <Val>, <Val2>, <Nend>, <Ninc>**

Описание аргументов команды:

**NNode** – номер узла или ALL – все узлы из выборки;

**Lab** – допустимая ссылка на тип силы (например, FX, MY, HEAT и т.п.);

**Val, Val2** – первое и второе (если допустимо) значения для выбранной степени свободы: если значение определяет массив, то имя необходимо выделять символами %;

**Nend, Ninc** – определяет последний номер узла и шаг счётчика (если необходимо выбрать узлы, пронумерованные по порядку).

Приложение давления в статических ГУ к КЭ через графический интерфейс:

*Preprocessor → Loads → Define Loads → Apply → Pressure → On Elements*  
или

*Solution → Define Loads → Apply → Pressure → On Elements*)  
или командой

**SFBEAM, <NElem>, <LKey>, <Lab>, <ValI>, <ValJ>, <Val2I>, <Val2J>, <IOFFST>, <JOFFST>, <LENRAT>**

Описание аргументов команды:

**NElem** – номер КЭ или ALL – все КЭ из выборки;

**LKey** – число, определяющее направление прикладываемого давления (определяется типом КЭ);

**Lab** – ссылка на тип нагрузки (в задачах механики – PRES);

**ValI, ValJ** – значения в I-ом и J-ом узле (если для J-ого узла не задано значение, то используется значение в I-ом);

**ValI2, ValJ2** – второе значение, не используется в последних версиях;

**IOFFST, JOFFST** – смещение прикладываемого давления от узла I к узлу J и наоборот соответственно;

**LENRAT** – настройка смещения: 0 – смещение в размерности длины (по умолчанию), 1 – смещение в долях от длины КЭ.

### Пример

Построим модель балки, жестко закрепленной по краям и нагруженной посередине поперечной силой (схема изображена на рис. 3.1). Геометрию и свойства возьмем из предыдущего примера.

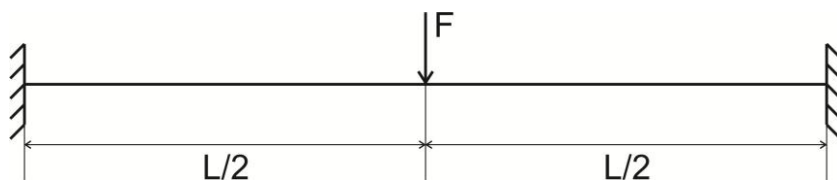


Рис. 3.1. Модель балки.

Рассмотрим некоторые варианты построения этой задачи. На первый взгляд кажется очевидным, что геометрически необходимо построить одну линию. Реализуем это построение и проанализируем особенности такого подхода подробнее. Ниже приведен код, включающий в себя построение геометрии, задание типа КЭ, свойств материала и поперечного сечения.

```
/PREP7  
N=10  
L=100  
a=10
```

```
b=5
```

```
K,,0,0,0  
K,,L,0,0  
LSTR,1,2
```

```
ET,1,BEAM188  
KEYOPT,1,3,0
```

```
MPTEMP,1,0  
MPDATA,EX,1,,210000  
MPDATA,PRXY,1,,0.3
```

```
SECTYPE, 1, BEAM, RECT, MyName, 0  
SECOFFSET, CENT  
SECDATA,10,5,3,3
```

```
TYPE, 1  
MAT, 1  
SECNUM, 1  
ESYS, 0  
LESIZE, 1, , , N, , , , 0  
LMESH, 1
```

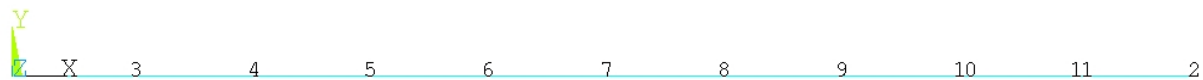
Теперь приложим кинематические ГУ:

```
DK,1, , , ,0,ALL, , , , ,  
DK,2, , , ,0,ALL, , , , ,
```

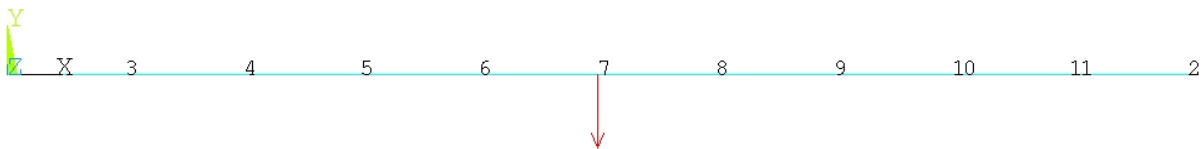
Особенность возникает при приложении поперечной силы. В месте приложения силы нет точки, но там может быть узел, так как его наличие в середине балки зависит от разбиения: при чётном количестве КЭ – нечетное количество узлов и есть узел в середине балки, при нечетном количестве – узла в середине балки нет.

Стоит также отметить особенности нумерации узлов сетки в ANSYS: в первую очередь нумеруются крайние точки линии, а только потом участок между этими точками (рис. 3.2а). Поэтому воспользуемся следующей формулой для определения номера узла в середине в случае произвольного разбиения:

```
NF=NINT ( (N+1) /2+1 )
F,NF,FY,-100
```



а



б

Рис. 3.2. Разметка.

Результат приложения силы приведен на рис. 3.2б. Очевиден вопрос – можно ли просто выбрать необходимый узел по координате? Ответ будет подробнее рассмотрен в главе 5, а пока попробуем исправить модель с помощью имеющегося багажа знаний. Чтобы упростить задачу определения точек приложения силы, изменим геометрию модели. Создадим три точки: две в области заделки и одну в месте приложения силы. Проведем через них линии попарно:

```
K,,0,0,0
K,,L/2,0,0
K,,L,0,0
LSTR,1,2
LSTR,2,3
```

Теперь задаём кинематические ГУ в точках 1 и 3, а нагрузку – в точке 2:

```
DK,1,, , ,0,ALL, , , , ,
DK,3,, , ,0,ALL, , , , ,
FK,2,FY,-100
```

Остальная часть кода не изменится, отличия возникают лишь при разбиении на КЭ сетку – разбиваем все линии:

```
LMESH, All
```

После этого необходимо задать тип решаемой задачи:

```
ANTYPE, 0
```

и запустить на решение:

```
SOLVE
```

Приведем полный листинг программы:

```
/PREP7
```

```
N=5
```

```
L=100
```

```
a=10
```

```
b=5
```

```
K, , 0, 0, 0
```

```
K, , L/2, 0, 0
```

```
K, , L, 0, 0
```

```
LSTR, 1, 2
```

```
LSTR, 2, 3
```

```
ET, 1, BEAM188
```

```
KEYOPT, 1, 3, 0
```

```
MPTEMP, 1, 0
```

```
MPDATA, EX, 1, , 210000
```

```
MPDATA, PRXY, 1, , 0.3
```

```
SECTYPE, 1, BEAM, RECT, MyName, 0
```

```
SECOFFSET, CENT
```

```
SECDATA, 10, 5, 3, 3
```

```
TYPE, 1
```

```
MAT, 1
```

```
SECNUM, 1
```

```
ESYS, 0
```

```
LESIZE, All, , , N, , , , , 0
```

```
!LMESH, All
```

```
LMESH, ALL
```

```
/SOL
```

```
DK, 1, , , , 0, ALL, , , , ,
```

```
DK, 3, , , , 0, ALL, , , , ,
```

```
FK, 2, FY, -100
```

```
ANTYPE, 0
```

## SOLVE

Рассмотрим на примере приложение силы по табличным данным: для этого рассмотрим балку, нагруженную силой по заданному закону (рис. 3.3), где  $F_1=10$  Н,  $F_2 = -9$  Н. В этом случае необходимо задать табличные данные, определяющие величину силы и тип аргумента, от которого она зависит.

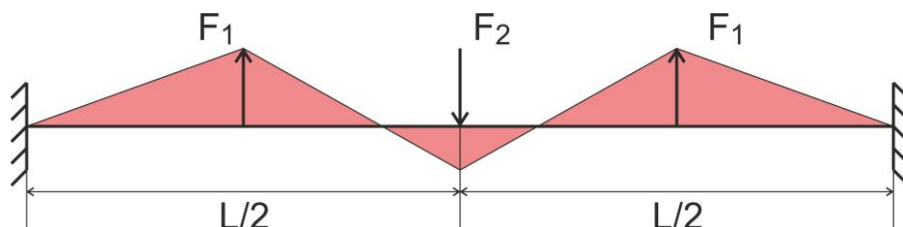


Рис. 3.3. Схема нагружения.

Создадим для этого таблицу (TABLE) с именем myForces из пяти КЭ. Также надо указать тип аргумента (в нашем случае это будет координата X):

```
*DIM,myForces,TABLE,5,1,1,X,
```

В первом столбце будут храниться данные о величине силы, в нулевом – координата X:

```
myForces(1,0)=0, 2, 5, 8,10  
myForces(1,1)=0,10,-9,10, 0
```

На рис. 3.4 приведён график функции:

```
*VPLOT,myForces(1,0),myForces(1,1)
```

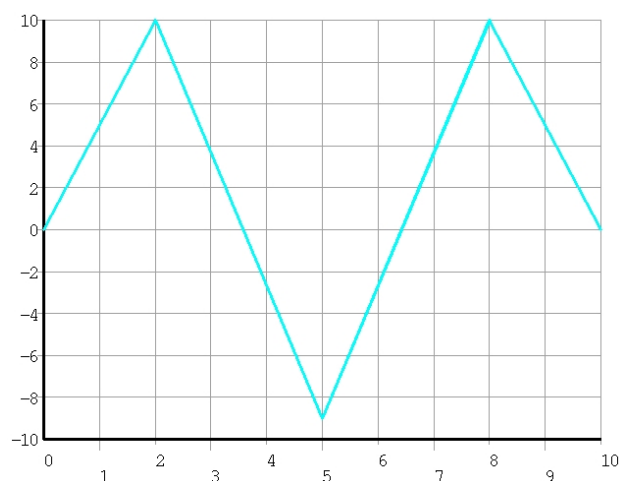
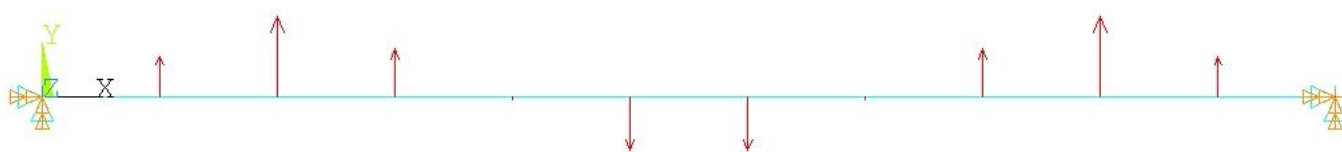
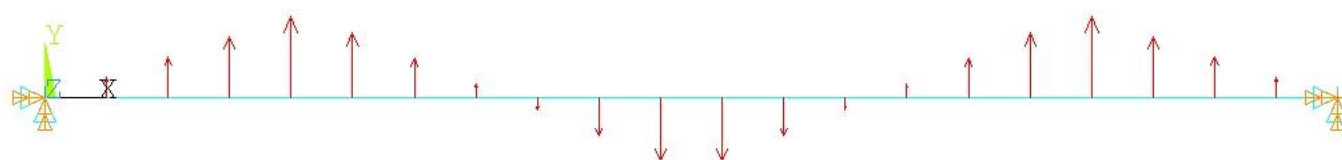


Рис. 3.4. График табличного представления нагрузки.

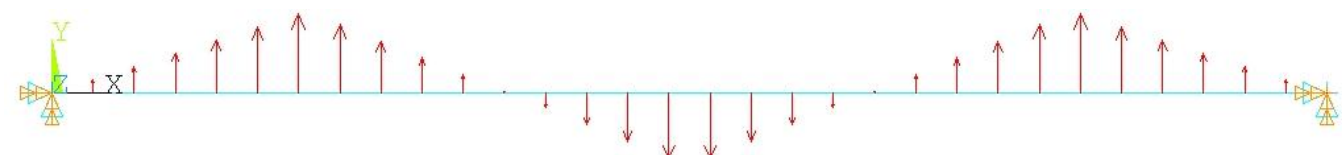
Приведенный код определяет табличную функцию, зависящую от координаты  $X$  (в активной СК); значения в промежуточных точках аппроксимацией по линейному закону.



а



б



в

Рис. 3.5. Распределение сил по узлам: а – для 11 узлов, б – для 21 узла, в – для 31 узла.



Для приложения силы к узлам по описанному закону используем команду:

```
F,ALL,FY,%myForces%
```

На рис. 3.5 показан результат приложения нагрузки при различной плотности КЭ сетки.

Приведем полный листинг программы:

```
NN=31
*DIM,myForces, TABLE, 5, 1, 1, X

myForces(1,0)=0, 2, 5, 8, 10
myForces(1,1)=0, 10, -9, 10, 0

/PREP7
ET,1,BEAM188
MPTEMP,1,0
MPDATA,EX,1,,2E11
MPDATA,PRXY,1,,0.3
SECDATA,0.2,0.2

K,,0,0,0
K,,10,0,0
L,1,2

LESIZE, All, , , NN, , , , 0
LMESH,ALL

/SOL
DK,1, , , , , , ALL, , , , ,
DK,2, , , , , , ALL, , , , ,

F,ALL,FY,%myForces%

/STATUS,SOLU
SOLVE
```

## ГЛАВА 4. ПОСТПРОЦЕССОР

Постпроцессор вызывается командой

### /POST1

В общем случае может быть определён некоторый набор решений. Чтение заданного решения через графический интерфейс:

*General Postporc* → *Read Results*

или командой

**SET**, <Ls>, [<Sb<, <Fact>, <IMG>, <TIME>, <ANG>, <NSET>, <ORDER>]

Описание аргументов команды:

**Ls** – номер шага нагружения, который должен быть прочтён: N – номер шага, FIRST – первый шаг, LAST – последний шаг, NEXT – следующий шаг, PREVIOUS – предыдущий шаг, NEAR – ближайшей по времени шаг, LIST – список шагов;

**Sb** – номер подшага (по умолчанию последний подшаг активного шага): исключение для BUCKLE или MODAL – номер подшага суть номер мода;

**Fact** – масштаб результатов (по умолчанию 1.0): применяется только к перемещениям и напряжениям,

**IMG** – настройка для комплексных результатов: 0 или REAL – действительная часть решения (по умолчанию); 1, 2 или IMAG – мнимая часть решения; 3 или AMPL – амплитуда; 4 или PHAS – сдвиг по фазе (в градусах от -180 до +180);

**TIME** – шаг определяется заданным временем (для гармонического анализа – частотой, для задач устойчивости – коэффициентом нагрузки) и используется, если в номере шага нагружения было задано;

**ANG** – величина угла (для гармонического анализа);

**NSET** – номер набора данных для чтения;

**ORDER** – настройка сортировки результатов гармонического анализа.

После загрузки необходимых данных решения, можно приступить к их графическому представлению.

### Деформированное состояние и векторные поля

Отображение деформированного состояния через графический интерфейс:

*General Postproc → Plot Results → Defomed Shape*

или командой

**PLDISP, <OPT>**

Здесь **OPT** – способ вывода: 0 – только деформированное состояние, 1 – начальное и деформированное состояния.

Масштаб деформированного состояния определяется командой

**/DSCALE, <WN>, <DMULT>**

Описание аргументов команды:

**WN** – номер окна для вывода (по умолчанию 1);

**DMULT** – AUTO или 0 – расчет масштаба из условия, что максимальные перемещения отображаются как 5% от максимального габарита (по умолчанию), 1 – нет масштаба, **FACTOR** – масштаб определяется численным значением **FACTOR**.

Результат отображения начального и деформированного состояний решаемой задачи (см. приложение 1) из прошлой главы приведен на рис. 4.1а.

Рассмотрим, как оценить количественно деформированное состояние. Для вывода векторных и скалярных величин предназначена команда меню:

*General Postproc → Plot Results → Contour Plot*

Результаты при этом могут быть выведены для узлов, КЭ и таблиц: *Nodal Solu* – команда **PLNSOL**, *Element Solu* – команда **PLESOL**, *Elem Table* – команда **PLETAB**, *Line Elem Res* – команда **PLLS**. Не будем заострять внимание на командах вывода результатов в узлах и КЭ – при необходимости информа-

цию можно почерпнуть из документации. На рис 4.1б приведена иллюстрация результатов в контурном представлении.

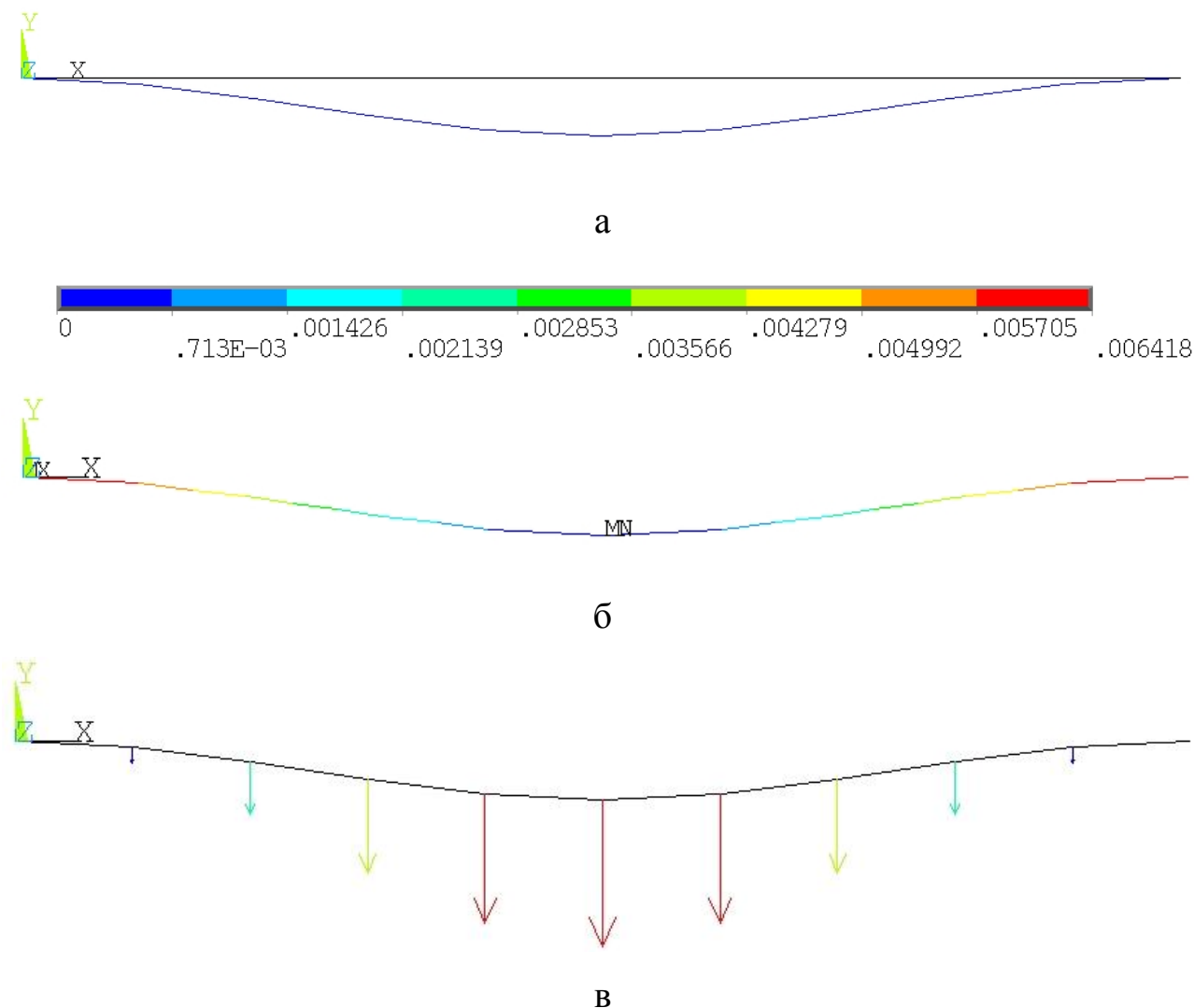


Рис. 4.1. Отображение состояния решаемой задачи: а - начальное и деформированное состояния, б – контурное представление модуля вектора перемещений, в – векторное представление вектора перемещений.

Для вывода векторных величин используется команда:

*General Postproc → Plot Results → Vector Plot.*

Для одномерных КЭ к таким величинам относятся только перемещения и углы поворота. На рис 4.1в приведены результаты расчётов в векторном виде.

## Контурные поля

Для графического вывода на экран значений тензоров напряжений и деформаций необходимо включить отображение поперечного сечения конструкции. При выводе результатов аппроксимация производится в узлах КЭ сетки, при отображении результатов в элементе – в каждом КЭ в зависимости от его типа.

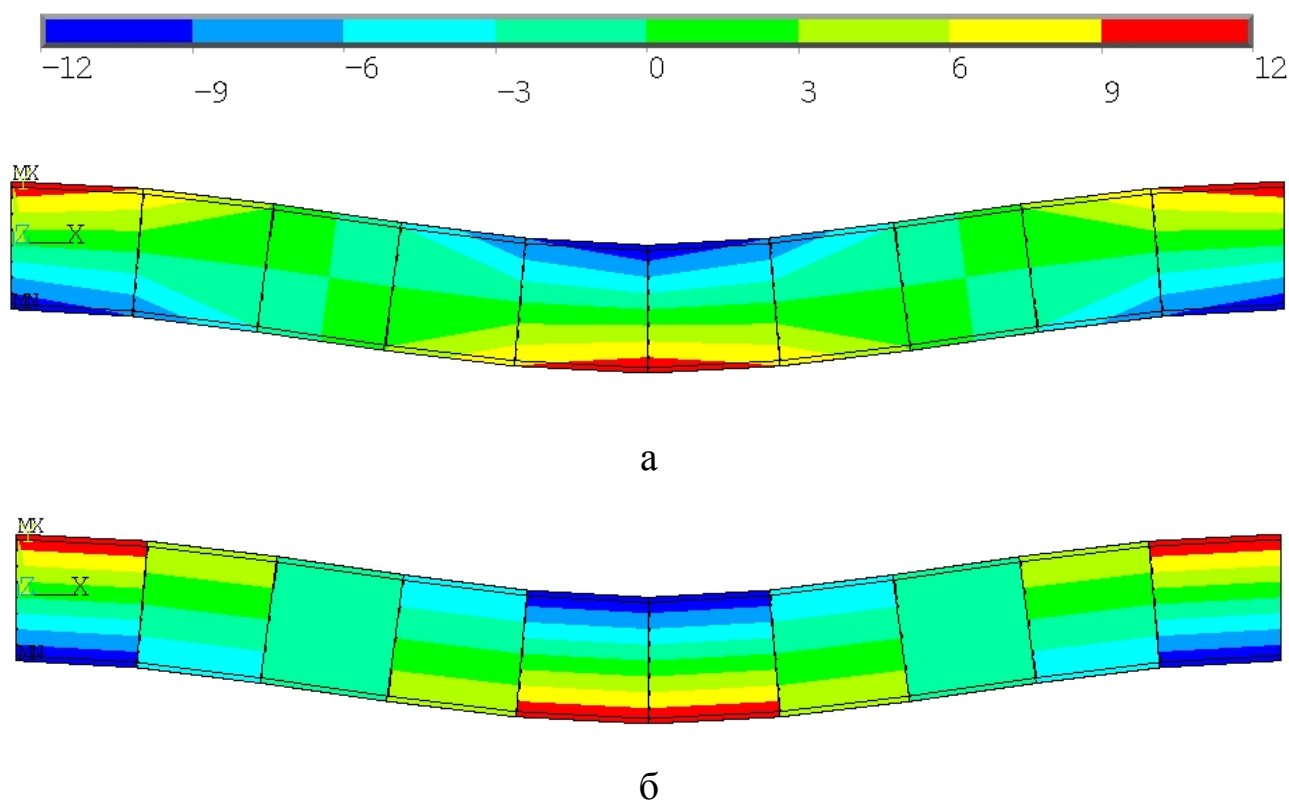


Рис. 4.2. Распределение нормальных напряжений: а – в узлах, б – в элементах.

На рис. 4.2а в каждом КЭ производится линейная аппроксимация напряжений по оси  $Oy$  – результаты на сопряженных гранях, вообще говоря, не совпадают.

При необходимости изменения цветовой кодировки надо щёлкнуть правой кнопкой мыши на шкале. В появившемся меню выбрать *Contour Properties* и далее перейти на закладку *Range*. Есть два формата шкалы: однородная и неоднородная. Однородная шкала делит интервал от минимального значе-

ния до максимального на 8 частей, для неоднородной шкалы можно задать свои интервалы (исключая минимальное значение). На рис. 4.3а приведены результаты первого главного напряжения в однородной шкале, а на рис. 4.3б – в неоднородной; при этом стоит отметить, что значения, превышающие наибольшее значение на шкале, отображаются серым цветом.

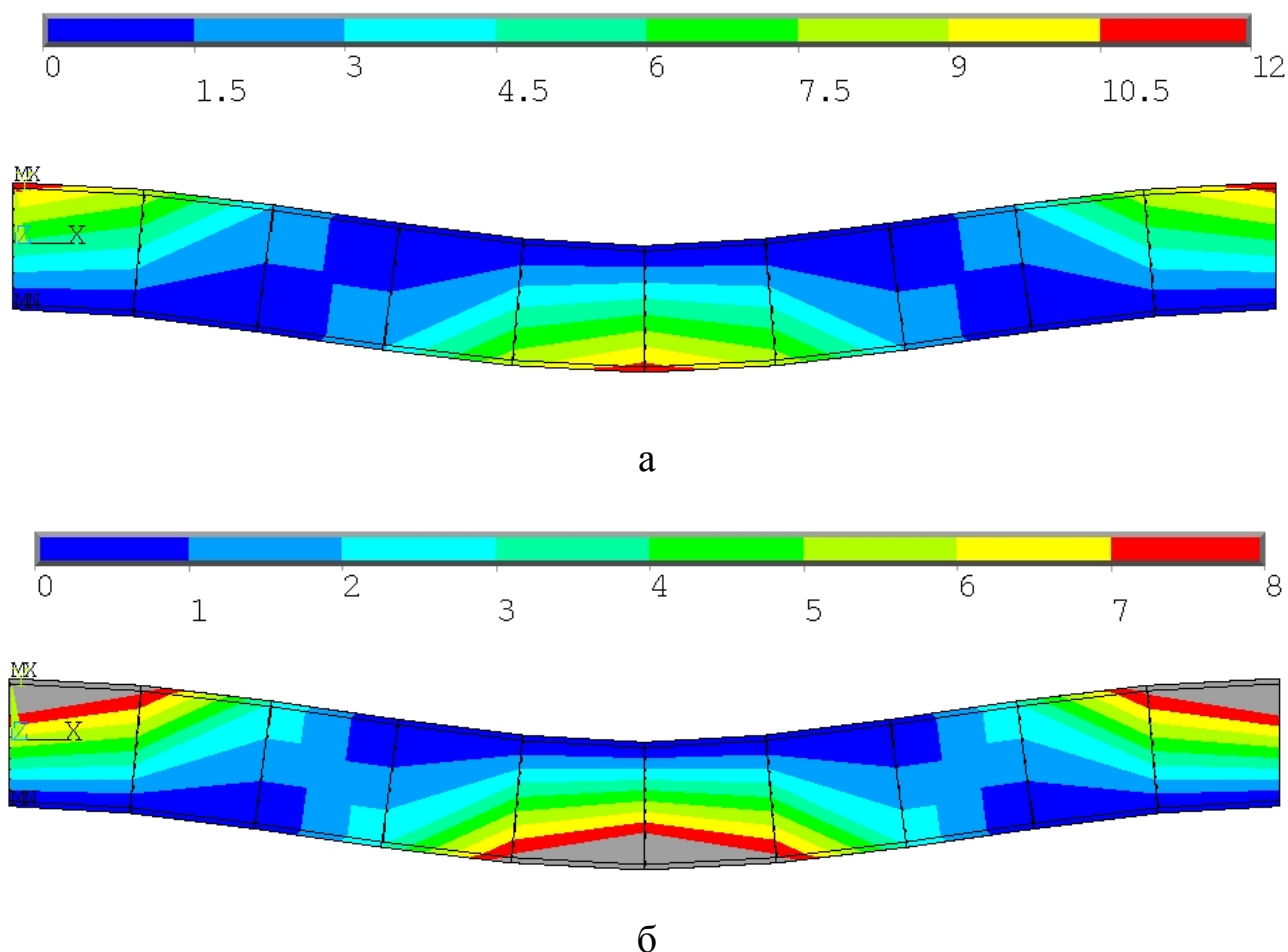


Рис. 4.3. Распределение первого главного напряжения:

а - однородная шкала, б – неоднородная шкала.

### Табличные данные и вывод

Для отображения результатов можно записать набор данных в таблицу, а затем вывести её на экран графически. В частности, для ряда данных одномерных КЭ возможен вывод только через извлечение данных. Например, ве-

личина изгибающего момента в I-ом или J-ом узлах (полный список извлекаемых данных приведён в документации для КЭ). Определить данные можно через графический интерфейс:

*General Postproc → Element Table → Define Table*

или командой

**ETABLE, <Lab>, [<Item>, <Comp>, <Opt>]**

Описание аргументов команды:

**Lab** – имя набора данных: REFL – обновляет данные всех существующих таблиц, STAT – выводит на экран данные таблиц, ERAS – удаляет таблицы;

**Item** – метка, определяющая извлекаемые данные: если значение равно ERAS, то будет удалена таблица под именем Lab;

**Comp** – компонента извлекаемых данных;

**Opt** – опция для хранения извлекаемых данных: MIN – сохраняет минимальное значение в КЭ, MAX – сохраняет максимальное значение, AVG – сохраняет среднее значение.

Согласно документации, данные о величине изгибающего момента около оси  $z$  (в локальной СК элемента) для КЭ BEAM188 хранятся по метке SMISC, компонент №3 отвечает за величину изгибающего момента в узле I, а №16 – в узле J. Зададим соответствующие таблицы:

```
ETABLE, MzI, SMISC, 3  
ETABLE, MzJ, SMISC, 16
```

Для графического отображения можно использовать путь:

*General Postproc → Element Table → Plot Elem Table* или *General Postproc → Plot Results → Elem Table*

после чего выбрать отображаемую таблицу или команду **PLETAB**. Пример получаемого изображения приведен на рис. 4.4а.

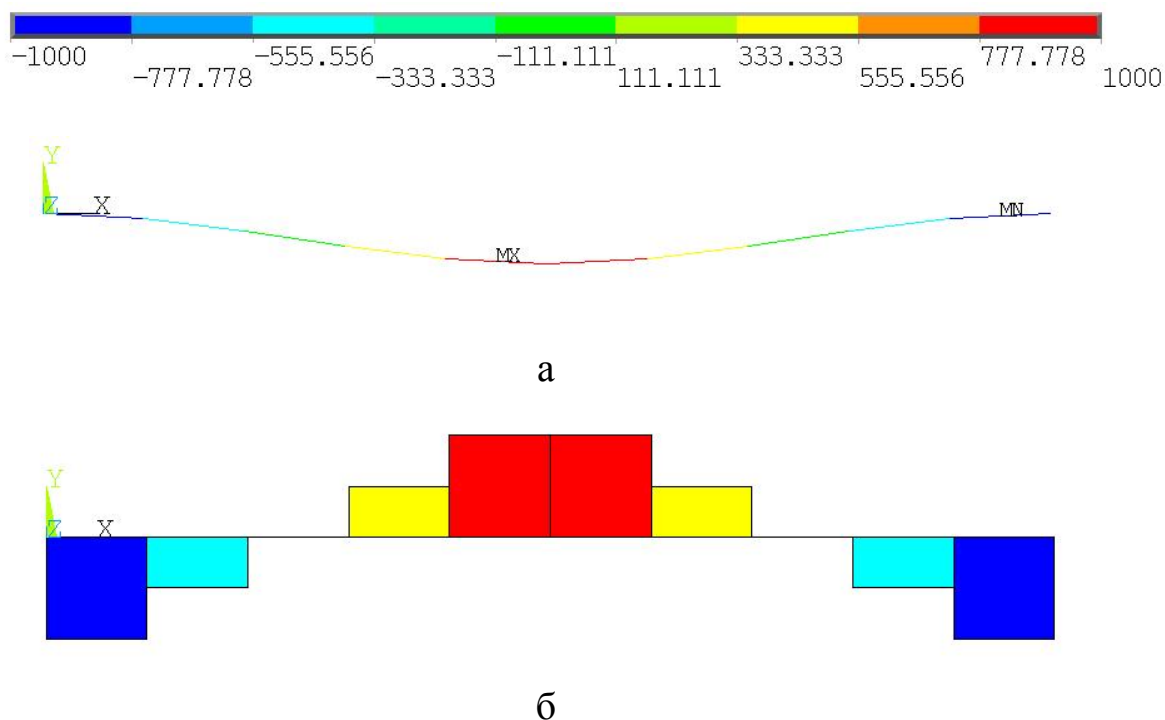


Рис. 4.4. Графическое изображение изгибающего момента:  
а - контурное представление, б – ступенчатое представление.

Другой способ отображения – это линейная аппроксимация табличных данных. Для вызова необходимо воспользоваться графическим интерфейсом:

*General Postproc → Plot Results → Line Elem Table*

или командой **PLLS**. Данные будут отображены цветной площадью поэлементно. При этом необходимо указывать два набора данных: для I-ого и J-ого узлов. Пример получаемого изображения приведен на рис. 4.4б. Команда для такого отображения:

```
PLLS, MzI, MzJ, 1, 0
```

Очевидно, что результаты расчётов и гладкость получаемых изображений будут зависеть от количества КЭ. Рассмотрим это детальнее на задаче об изгибе шарнирно опертой балки длиной 100 мм с действующей посередине силой 100 Н. (см. приложение 2).



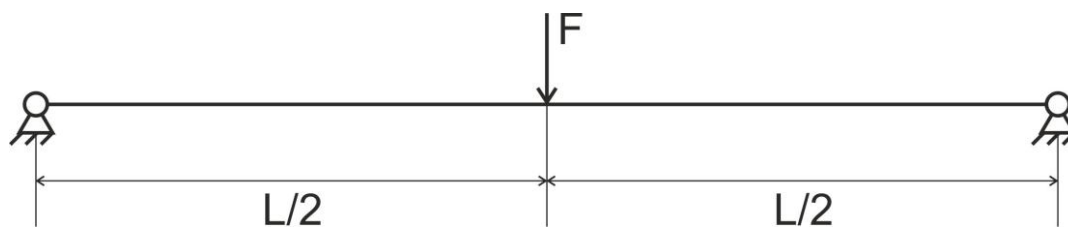
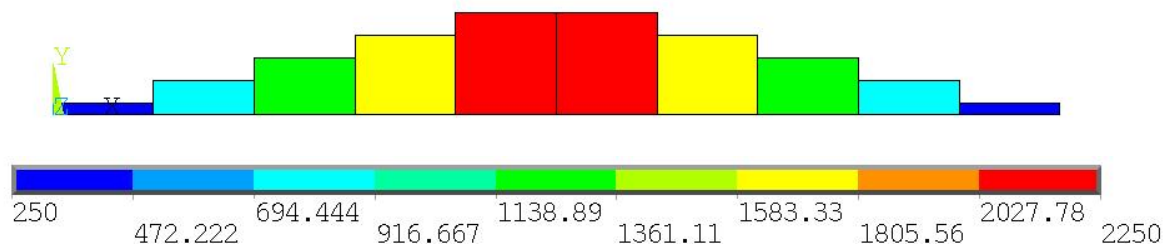


Рис. 4.5. Модель балки.

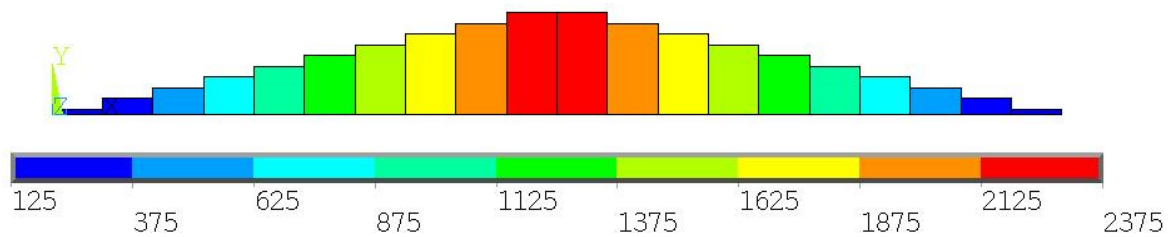
Эпюра изгибающего момента для этой задачи известна – это треугольник: на краях значение изгибающего момента равно нулю, максимум достигается в точке приложения силы и равен  $2500 \text{ Н}\cdot\text{мм}$  (рис. 4.5).

На рис. 4.6 приведены эпюры изгибающих моментов при различной плотности разбиения. Так как известно аналитическое решение, то можно оценить относительную погрешность для наибольшей величины изгибающего момента. На рис. 4.7 приведен график рассчитанной относительной погрешности для разного количества КЭ.

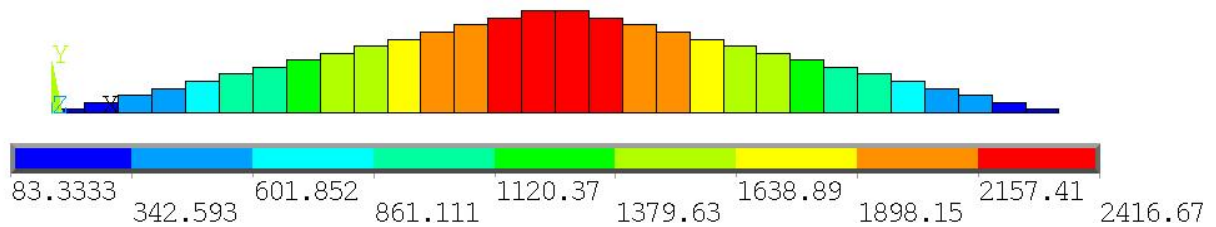
Для повышения точности решения необходимо поменять в настройках КЭ функцию формы на квадратичную ( $\text{KEYOPT}(3)=2$ ). В этом случае уже на малой КЭ-сетке (10 элементов) мы получаем величину наибольшего изгибающего момента в точности совпадающую с аналитическим решением. Расчётная эпюра изгибающих моментов приведена на рис. 4.8.



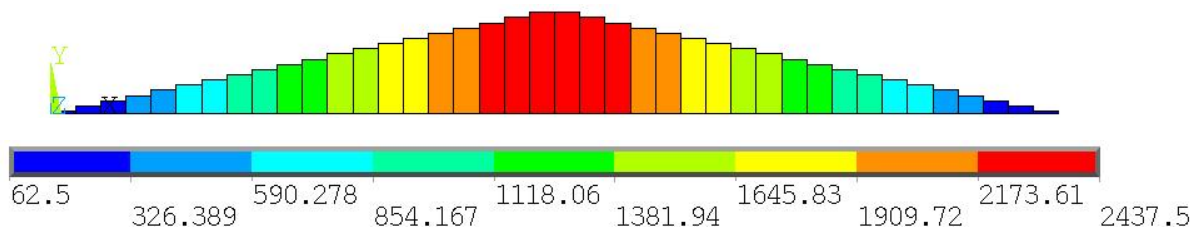
а



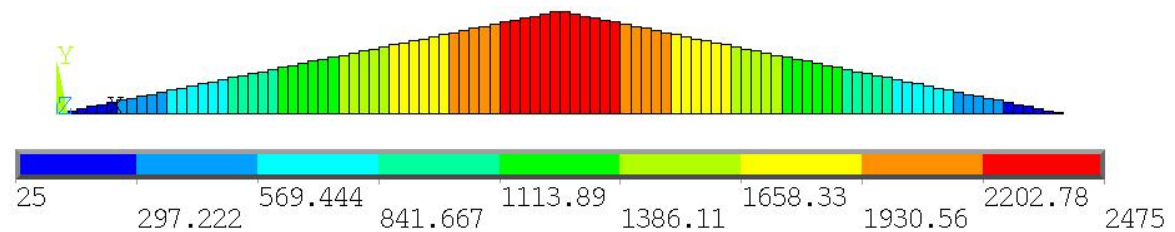
б



в



г



д

Рис. 4.6. Эпюры изгибающих моментов для разбиения на: а – 10 элементов, б – 20 элементов, в – 30 элементов, г – 40 элементов, д – 100 элементов.

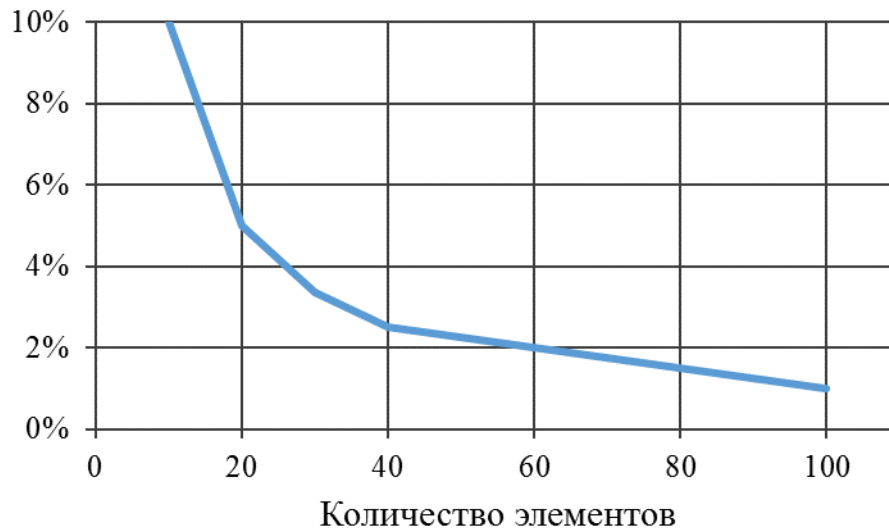


Рис. 4.7. Сходимость максимального изгибающего момента при сгущении КЭ-сетки.

Таким образом, можно заметить, что добиться необходимой точности можно несколькими способами: увеличивая размерность КЭ-сетки или повышая степень функций формы (если это позволяют настройки КЭ).

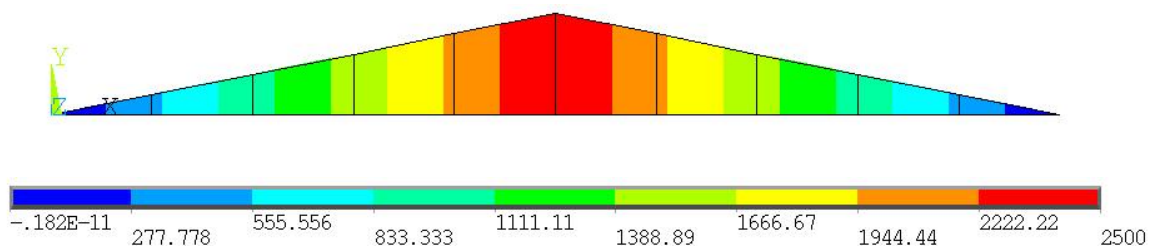


Рис. 4.8. Эпюра изгибающих моментов для разбиения на 10 КЭ.

## Анимация

Для настройки анимации результатов предназначен набор команд меню *PlotCtrls* → *Animate*.

Отображение результатов определения НДС через графический интерфейс:

*PlotCtrls → Animate → Deformed Results,*

*PlotCtrls → Animate → Isosurfaces,*

*PlotCtrls → Animate → Deformed Results.*

При настройке анимации необходимо указать способ вывода (количество фреймов и время задержки между ними) и что именно выводить: компоненты напряжений или поле перемещений (рис. 4.9).

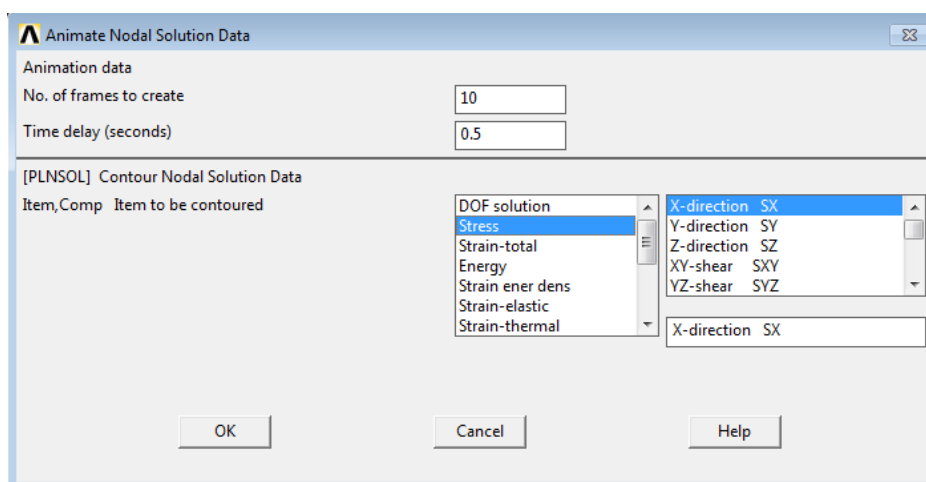


Рис. 4.9. Интерфейс настройки анимации.

После настройки анимации можно посмотреть результат и сохранить его в файл. Для этого предназначена команда меню

*PlotCtrls → Animate → Save Animation.*

### Выгрузка данных

Для вывода результатов в виде списка существует набор команд, расположенный в меню:

*General Postproc → List Results*

Отметим ряд пунктов, полезных для анализа, например, для вывода результатов расчета в узлах:

*General Postproc → List Results → Nodal Solution* или *General Postproc → List Results → Vector Data*

для вывода результатов расчета в элементах:

*General Postproc → List Results → Element Solution*

Подробнее автоматизация получения данных и работа с файлами описана в главе 5.

## ГЛАВА 5. ДОПОЛНИТЕЛЬНЫЕ ИНСТРУМЕНТЫ

### Выбор элементов геометрии и сетки

На практике часто возникает необходимость выбрать подмножество узлов или КЭ, связанное с частью геометрии, которая может быть определена топологией, расположением в пространстве или иным критерием. Для этих целей в ANSYS предусмотрен удобный инструмент выбора: *Select* → *Entities*.

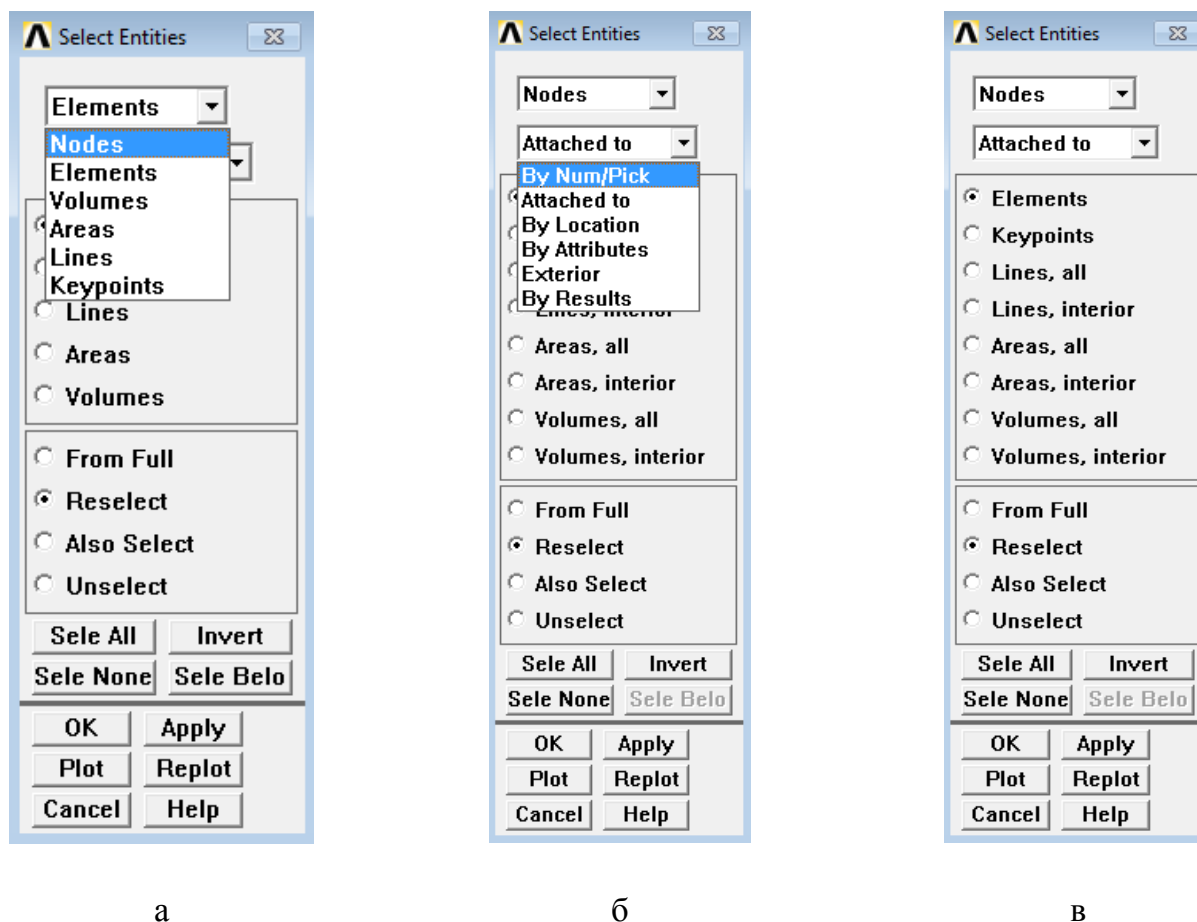


Рис. 5.1. Интерфейс выбора объектов.

Для выбора всех однотипных объектов используется команда

**ALLSEL, <LabT>, <Entity>**

Описание аргументов команды:

**LabT** – тип выбора: ALL – выбор всех объектов из всего или заданной выборки, BELOW – выбор объектов (того же уровня или ниже), связанных с активной выборкой;

**Entity** – тип объектов для выбора (перечисление приведено по убыванию в иерархии объектов): ALL – все объекты, VOLU – объемы, AREA – площади, LINE – линии, KP – точки, ELEM – элементы, NODE – узлы.

Например, для выбора всех объектов синтаксис команды будет выглядеть следующим образом:

`ALLSEL, ALL`

Выборка определяется типом объекта: Nodes (Узел), Elements (Конечный элемент), Volumes (Объемное тело), Areas (Поверхностное тело), Lines (Линейный объект), Keypoints (Точки); а также критерием, который может изменяться в зависимости от типа объекта.

Не будем заострять внимание на описании всех возможных команд для выбора объектов: при необходимости рекомендуется сначала использовать графический интерфейс, а потом из файла истории извлекать необходимые команды. Некоторые команды, упоминаемые в примерах, будут описаны, но в краткой форме. Отметим лишь общие возможности инструмента выбора объектов.

Табл. 5.1.

Объект	Основные команды	Смешанные команды
Nodes	NSEL	NSLE, NSLK, NSLL, NSLA, NSLV
Elements	ESEL	ESLN, ESLL, ESLA, ESLV
Keypoints	KSEL	KSLN, KSLI
Hard Points	KSEL, ASEL, LSEL	–
Lines	LSEL	LSLA, LSLK
Areas	ASEL	ASLL, ASLV
Volumes	VSEL	VSLA
Components	CMSEL	–

В таблице 5.1, указаны все возможные объекты и их «основные» и «смешанные» команды. Первый тип команд позволяет сделать непосредственный выбор объекта без его связи с другим объектом, второй – на основании связи одного объекта с другим иного типа, например, выбрать все узлы на заданной линии – NSLL.

В таблице 5.2 приведены все возможные виды выбора объектов.

Табл. 5.2.

Nodes	Elements	Volumes	Areas	Lines	Keypoints
Общие критерии					
By Num/Pick	By Num/Pick	By Num/Pick	By Num/Pick	By Num/Pick	By Num/Pick
Attached to	Attached to	Attached to	Attached to	Attached to	Attached to
By Location		By Location	By Location	By Location	By Location
By Attributes	By Attributes	By Attributes	By Attributes	By Attributes	By Attributes
Exterior			Exterior	Exterior	Exterior
By Results	By Results		By Hard Points	By Hard Points	By Hard Points
Специальные критерии					
	By Elem Name		Concatenated	By Length/Rad	
	Live Elem's			Concatenated	
	Adjacent to				

С точки зрения логики создания выборки существуют следующие поля:

**From Full** – выбрать заданное подмножество объектов из всего множества объектов,

**Reselect** – выбрать заданное подмножество объектов из активного подмножества объектов,

**Also Select** – выбрать заданное подмножество объектов и добавить его к активному подмножеству объектов,



**Unselect** – выбранное заданное подмножество объектов удаляется из активного подмножества объектов.

При работе с выборками объектов в APDL существует набор специальных команд. Рассмотрим некоторые из них.

**NSL (NUM), ESEL (NUM), KSEL (NUM), LSEL (NUM), ASEL (NUM), VSEL (NUM)** – проверка статуса узла, КЭ, точки, линии, поверхности и объема по заданному номеру; функции возвращают число: -1 – объект не выбран, 0 – не определен, 1 – объект выбран.

**NDNEXT (NUM), ELNEXT (NUM), KPNEXT (NUM), LSNEXT (NUM), ARNEXT (NUM), VLNEXT (NUM)** – функции возвращают номер следующего после заданного аргументом объекта (узел, элемент, точка, линия, поверхность, объем).

**CENTRX (E), CENTRY (E), CENTRZ (E)** – функции возвращают соответствующую координату центра КЭ (X, Y, Z) в активной СК по заданному номеру.

**NX (N), NY (N), NZ (N)** – функции возвращают координату узла (X, Y или Z) в активной СК по заданному номеру.

**KX (K), KY (K), KZ (K)** – функции возвращают координату точки (X, Y или Z) в активной СК по заданному номеру.

**NODE (X, Y, Z)** – функции возвращают номер ближайшего узла по заданным координатам.

**KP (X, Y, Z)** – функции возвращают номер ближайшей точки по заданным координатам.

### **Компоненты и сборки**

Для удобства можно сохранять сгруппированные объекты в компоненты (*Components*) и сборки (*Assemblies*). Компоненты – это именованный однотипный набор объектов (например, набор узлов, лежащих в заданной обла-

сти). Сборка – это именованный набор компонент или сборок, при этом допускается вложение до 5 уровней сборок.

Для создания компоненты требуется выбрать необходимые объекты и использовать команду **CM** или в графическом меню:

*Select → Comp/Assembly → Create Component.*

Синтаксис команды:

**CM, <CName>, <ObjType>**

Описание аргументов команды:

**CName** – имя компоненты;

**ObjType** – тип объектов для сохранения (VOLU, AREA, LINE, KP, ELEM, NODE).

Для создания сборки необходимо использовать команду **CMGRP** или в графическом меню:

*Select → Comp/Assembly → Create Assembly*

и выбрать в открывшемся окне необходимые компоненты и/или сборки.

Синтаксис команды:

**CMGRP, <AName>, <Nam1>, [<Nam2>, ..., <Nam8>]**

Описание аргументов команды:

**AName** – имя сборки;

**Nam1, ..., Nam8** – имена компонент или сборок, которые необходимо включить в заданную сборку (от одного до восьми имен).

Для активации заданной компоненты или сборки необходимо в графическом меню выбрать:

*Select → Comp/Assembly → Select Comp/Assembly*

или использовать команду **CMSEL** (выбирает заданную компоненту или сборку по её имени):

**CMSEL, <Type>, <Name>, <ObjType>**

Описание аргументов команды:

**Type** – логический тип выбора: S – задать как новую выборку, R – выбрать из активной выборки, A – добавить в активную выборку, U – удалить из активной выборки, ALL – выбрать все компоненты, NONE – не выбирать ни одну компоненту;

**Name** – имя компоненты или сборки (если в предыдущем параметре заданы S, R, A или U);

**ObjType** – тип объекта. Вводится, если не было введено значение **Name**, и может принимать следующие значения: VOLU, AREA, LINE, KP, ELEM, NODE.

В процессе работы существующие компоненты/сборки могут быть изменены командой **CMEDIT** (*Select → Comp/Assembly → Edit Assembly*), удалены командой **CMDELE** (*Select → Comp/Assembly → Delete Comp/Assembly*), выведены списком командой **CMLIST** (*Select → Comp/Assembly → List Comp/Assembly*). Для удобной ориентации в логике компонент/сборок предназначен менеджер: *Select → Component Manager*.

Компоненты и сборки автоматически обновляются: при изменении первоначальной причины выбора объектов происходит их автоматическое удаление из компонентов сборки; если в компоненте не остаётся объектов, она также автоматически удаляется. Например, пусть имеются две линии, разбитые на КЭ-сетку, и две компоненты: Elem1 хранит в себе набор элементов первой линии, Elem2 – первой и второй линии. Если удалить КЭ-сетку на первой линии, то компонента Elem1 будет удалена, так как окажется пустой, а в компоненте Elem2 останутся только элементы из второй линии. Если после этого опять разбить первую линию на КЭ-сетку, компонента Elem2 останется без изменения, то есть будет содержать все те же элементы второй линии, а необходимые новые элементы первой линии потребуется добавить самостоятельно.

## Извлечение данных \*GET

Для извлечения данных из различных объектов предусмотрена команда **GET**

**\*GET, <Var>, <Obj>, <ObjNum>, <IT1>, <ITN1>, [<IT2>, <ITN2>]**

Описание аргументов команды:

**Var** – переменная для хранения извлекаемого результата;

**Obj** – тип объекта;

**ObjNum** – номер или ссылка, описывающие конкретный объект;

**IT1** – номер или ссылка, описывающие данные, извлекаемые для заданного объекта;

**ITN1** – номер или ссылка, описывающие данные, извлекаемые для заданного объекта при заданном **IT1**;

**IT2, ITN2** – второй набор номеров или ссылок.

По типам извлекаемых данных можно выделить:

- общие данные;
- данные препроцессора;
- данные процессора;
- данные постпроцессора;
- данные вероятностных моделей.

Документация по команде структурирована по описанному выше разделению. Далее в каждом параграфе идет разделение по типу объекта. Для каждого заданного типа приведена таблица, содержащая все возможные для извлечения данные и набор команд для обращения к ним.

Данные, извлекаемые для одного и того же объекта, могут отличаться в зависимости от этапов: от построения модели до анализа результатов решенной задачи.

Так, например, для узлов (NODE) это могут быть данные о номере, координатах или приложенных силах на этапе препроцессинга (\*GET

Preprocessing Items, Entity = NODE), а на этапе постпроцессинга, очевидно, это могут быть компоненты вектора перемещений, напряжений и т.п. (\*GET Postprocessing Items, Entity = NODE).

Табл. 5.1.

<b><i>Entity = NODE, ENTNUM = N (node number) for averaged nodal results based on selected elements:</i></b>			
<b>*GET, Par, NODE, N, Item1, IT1NUM, Item2, IT2NUM</b>			
<b>Item1</b>	<b>IT1NUM</b>	<b>Description</b>	<b>Перевод</b>
<b>S</b>	X, Y, Z, XY, YZ, XZ	Component stress.	Компоненты тензора напряжения
	1, 2, 3	Principal stress.	Главные напряжения
	INT, EQV	Stress intensity or equivalent stress.	Интенсивность напряжений и напряжения по Мизесу
	MAXF	Maximum stress, failure criterion.	Максимальные напряжения, критерий разрушения.
	TWSI	Tsai-Wu strength failure criterion.	Критерий Тсай-Бу.
	TWSR	Inverse of Tsai-Wu strength ratio index failure criterion.	Обратный критерий Тсай-Бу, коэффициент запаса.

Рассмотрим подробнее устройство документации и правила работы с командой. Исследуем часть таблицы из документации для данных, извлекаемых для узла на этапе постпроцессинга, – так в таблице 5.3 приведены данные о

напряжениях. В заголовке таблицы указаны необходимые настройки для команды

### ***\*GET, Par, NODE, N, Item1, IT1NUM***

Описание аргументов команды:

**Par** – переменная, в которой будут сохранены данные;

**NODE** – тип объекта (узел – NODE);

**N** – номер узла.

Чтобы получить данные о рассчитанных напряжениях в узле, необходимо задать  $Item1 = S$ , а  $IT1NUM$  – в зависимости от цели: например, чтобы получить данные о нормальных напряжениях по оси  $X$  –  $IT1NUM = X$ .

Приведем конкретные примеры использования команды:

```
*GET, Var4S, NODE, 1, S, X
```

в переменную **Var4S** будет записана величина нормальных напряжений по оси  $Ox$  из узла под номером 1;

```
*GET, PrinMax, NODE, 15, S, 1
```

в переменную **PrinMax** будет записана величина первых главных напряжений из узла под номером 15.

## **Создание макросов**

Как уже отмечалось выше, макрос – это набор команд. На практике при написании программ с целью повышения удобства можно разделить исполняемый код на набор файлов. В ряде случаев, например, при чтении из файла, это необходимо.

Для создания макроса средствами APDL используется команда

**\*CREATE, <FName>, [<Ext>]**

**<BODY>**

**\*END**

Описание аргументов команды:

**FName** – название файла (с путём, при необходимости; по умолчанию файлы сохраняются в рабочей папке проекта): по нему в последующем реализуется вызов макроса;

**Ext** – расширение файла (до восьми символов);

**BODY** – набор команд (тело макроса).

Для вызова макроса есть две команды **\*USE** и **/INPUT**. Как уже было отмечено выше, макрос записывается в файл – подключение файла реализуется командой:

**\*ULIB, <Fname>, <Ext>**

Описание аргументов команды:

**Fname** – название файла (с путем, при необходимости);

**Ext** – расширение файла (до восьми символов).

Рассмотрим команды вызова подробнее.

**\*USE, <Name>, [<ARG1>, ... , <AR18>]**

Описание аргументов команды:

**Name** – название макроса;

**ARG1, ..., AR18** – список аргументов (при этом синтаксис ARG и номер являются зарезервированными).

**/INPUT, <FName>, <Ext>, [<-> , <LINE>, <LOG>]**

Описание аргументов команды:

**Fname** – название файла (с путем, при необходимости);

**Ext** – расширение файла (до восьми символов);

- – неиспользуемое поле.

**LINE** – номер линии или название метки, с которой необходимо начать чтение файла макроса: пустое, 0 или 1 – чтение с начала, номер строки или :метка – чтение с заданной строки или метки;

**LOG** – настройки для файла истории: 0 – в файле будет сохранена только команда **/INPUT**, 1 – будут сохранены команды из читаемого файла.

Рассмотрим подробнее примеры использования макросов. Создадим файл с названием *geom.mcr*, в который запишем следующий текст:

```
K, , 0, 0, 0  
K, , 1, 0, 0  
K, , 2, 0, 0  
LSTR, 1, 2  
LSTR, 2, 3
```

Теперь создадим следующий файл, где *DirPath* – путь к файлу:

```
/CWD, <DirPath>  
/prep7  
/INPUT, geom, mcr
```

При вызове файла *geom.mcr* командой */INPUT* записанный в файле код будет выполнен в программе ANSYS. Рассмотрим следующий пример, но уже с параметрами. Изменим файл:

```
K, , 0, 0, 0  
K, , L1, 0, 0  
K, , L2, 0, 0  
LSTR, 1, 2  
LSTR, 2, 3
```

В программе выполним следующий код:

```
/CWD, <DirPath>  
/prep7  
L1=3  
L2=4  
/INPUT, geom, mcr
```

При обращении к макросу *geom.mcr*, поскольку определены величины параметров *L1* и *L2*, будет получен результат построения геометрии. При этом стоит отметить, что указанные параметры формально не являются аргументами, а значит, при вызове должны точно совпадать с параметрами в исходном файле макроса.

Рассмотрим теперь команду *\*USE*. Создадим файл *geom2.mcr* со следующим содержанием:

```
K, , 0, 0, 0  
K, , ARG1, 0, 0
```



```
K,,ARG2,0,0  
LSTR,1,2  
LSTR,2,3
```

Отличие от прошлого файла только в именах параметров: теперь вместо L1 и L2 используются зарезервированные идентификаторы ARG1 и ARG2. Применяя команду \*USE, эти аргументы можно указать, как в стандартной процедуре:

```
/CWD,<DirPath>  
/prep7  
*USE, geom2.mcr,1,3
```

или использовать другие переменные:

```
/CWD,<DirPath>  
/prep7  
L1=1  
L2=5  
*USE,geom.mcr,L1,L2
```

Теперь рассмотрим пример создания макросов через команду:

```
/CWD,<DirPath>  
! Создаем макрос  
*CREATE, Cgeom1  
K,,0,0,0  
K,,L1,0,0  
K,,L2,0,0  
LSTR,1,2  
LSTR,2,3  
*END  
! основной код  
/prep7  
L1=3  
L2=4  
/INPUT,Cgeom1
```

Команда CREATE не только создаст макрос как набор команд, но и определит файл Cgeom1 в рабочей папке, в котором будут все команды из тела CREATE. Создание макроса с аргументами проводится аналогично:

```
/CWD,<DirPath>  
*CREATE, Cgeom2
```

```

K, , 0, 0, 0
K, , ARG1, 0, 0
K, , ARG2, 0, 0
LSTR, 1, 2
LSTR, 2, 3
*END
/prep7
L1=1
L2=10
*USE, Cgeom2, L1, L2

```

Описанный инструментарий позволяет упрощать и оптимизировать программный код.

### Работа с файлами

Для работы с файлами в первую очередь необходимо определить связь команды ввода/вывода с заданным файлом. Для этого предназначена команда

**\*CFOPEN, <Fname>, <Ext>, -, <Loc>**

Описание аргументов команды:

**Fname** – имя файла при необходимости с путём (по умолчанию все размещается в рабочей папке);

**Ext** – расширение файла;

- – пустое поле;

**Loc** – тип перезаписи файла: <пустое> - перезапись, APPEND – дописывать в конец файла.

После вызова этой команды весь ввод и вывод будет осуществляться в указанный файл. Для отключения связи с заданным файлом используется команда

**\*CFCLOS**

Рассмотрим теперь команды ввода/вывода. Команды такого рода обязательно должны содержаться в отдельном файле/макросе.

Команда вывода (запись в файл)

**\*VWRITE, <Par1>, [ ... , <Par19>] //**

## <FORMAT>

Описание аргументов команды:

**Par1, ... , Par19** – список переменных для вывода в файл;

¶ – символ, обозначающий конец строки;

**FORMAT** – формат вывода данных (в синтаксисе Fortran или C для строковых данных длиной более 8 символов).

Разделителем между окончанием команды **\*VWRITE** и форматами вывода является только символ конца строки. Это означает, что не допускается табуляция перед форматом вывода.

Правило записи массивов в файл отличается. В этом случае необходимо указать номер элемента с которого будет начинаться вывод данных. Весь массив будет выведен построчно. Для вывода только заданных строк или столбцов используются команды **\*VLEN** или **\*VMASK**. При работе с многомерными массивами вывод всегда производится построчно, при необходимости вывода по столбцам используется команда **\*VOPER**.

Команда ввода (чтение из файла)

**\*VREAD, <Var>, <Fname>, <Ext>, -, <Label>, <n1>, <n2>, <n3>, <NSKIP> ¶**  
**<FORMAT>**

Описание аргументов команды:

**Var** – переменная, в которую сохраняются данные;

**Fname** – имя файла при необходимости с путём (по умолчанию все размещается в рабочей папке);

**Ext** – расширение файла;

- – пустое поле;

**Label** – порядок заполнения массива: IJK (по умолчанию), IKJ, JIK, JKI, KIJ, KJI;

**n1, n2, n3** – размерность массива согласно правилу заполнения (по умолчанию n2 и n3 равны 1);

**NSKIP** – номер строки, с которой необходимо начать чтение файла (по умолчанию 0);

**¶** – символ, обозначающий конец строки;

**FORMAT** – формат чтения данных (в синтаксисе Fortran или C для строковых данных длиной более 8 символов).

Приведем пример формата:

**KAN** – текстовые данные, K – количество повторений в строке, N – количество символов;

**KFN.M** – действительные числа с фиксированной точкой, K – количество повторений в строке, N – количество позиций для числа, M – количество позиций в дробной части;

**KGN.M** – действительные числа двойной точности, K – количество повторений в строке, N – количество позиций для числа, M – количество позиций в дробной части;

**KX** – пробел, K – количество пробелов;

**K/** – конец строки, K – количество строк.

Пример использования формата:

```
/CWD, <DirName>
```

```
! Данные для вывода
```

```
vLine = ' | '
```

```
*dim,nNodes,,5
```

```
*dim,nValues,,5
```

```
nNodes(1) = 1, 4, 5, 6, 7
```

```
nValues(1) = -12.54, 12.874, -4.5, 612, 0.577
```

```
! Макрос для записи в файл
```

```
*CREATE, OutSave
```

```
*cfopen,out.txt
```

```
*VWRITE,'Name','Node','Value','Comm.'
```

```
(A4,2X,A6,4X,A6,A15)
```

```
*VWRITE,'Temp: ',nNodes(1),nValues(1),vLine, 'answer:  
,42
```

```
(A6,F3.0,2X,G16.8,A3,A10,F5.2)
```

```
*cfclose  
*END
```

```
/INPUT, OutSave
```

В выходной файл запишется:

Name	Node	Value	Comm.
Temp:	1.	-12.540000	answer: 42.00
Temp:	4.	12.874000	answer: 42.00
Temp:	5.	-4.5000000	answer: 42.00
Temp:	6.	612.00000	answer: 42.00
Temp:	7.	0.57700000	answer: 42.00

Рис. 5.2. Выходной файл.

## ГЛАВА 6. РАЗБОР ЗАДАЧ

### Задача 1. Расчёт бруса переменного сечения на сжатие

Рассмотрим задачу о сжатии стального ступенчатого бруса. На рис. 6.1.1 приведена схема бруса и ГУ. В каждом поперечном сечении брус прямоугольный, соотношение сторон поперечного сечения – 2:1. Линейные размеры бруса – в таблице 6.1.1.

Табл. 6.1.1.

Номер сечения	1	2	3	4	5
Длина, мм	20	40	20	60	40
Поперечное сечение, мм	8x16	6x12	4x8	3x6	5x10

Сжимающая нагрузка  $F = 500$  Н. Механические свойства материала: модуль Юнга – 210 ГПа, коэффициент Пуассона – 0.3.

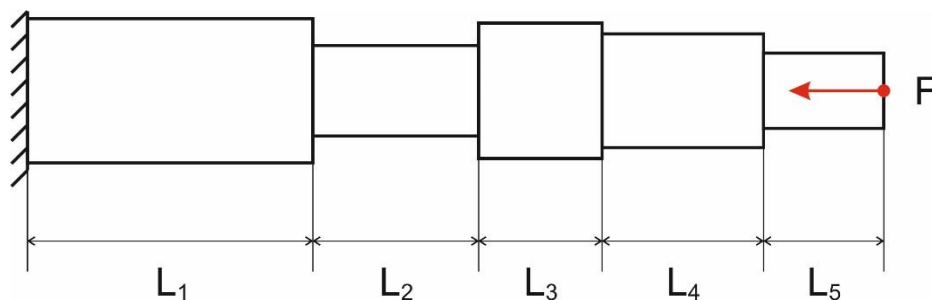


Рис. 6.1.1. Схема нагружения ступенчатого бруса.

#### *Ввод исходных данных*

В первую очередь зададим параметр, который будет определять величину нагрузки  $F$ :

```
Load=500
```

Для описания геометрии введем двумерный массив `body`. В первом и втором столбцах будут храниться длина и размер поперечного сечения  $i$ -го участка соответственно, так как один размер поперечного сечения по условию

задачи больше второго в два раза, то для хранения достаточно одного числа. Такой подход позволяет оптимизировать процесс построения модели и является предпочтительным при увеличении количества ступеней бруса.

```
*DIM, body, array, 5, 2
body (1,1) = 20, 40, 20, 60, 40
body (1,2) = 8, 6, 4, 3, 5
```

Перейдем к построению геометрии, зайдём в препроцессор и создадим опорную точку в начале СК:

```
/PREP7
K,,0,0,0
```

Для создания остальных точек введем переменную *hy*: перед итерациями она равна 0, а на каждой итерации она увеличивается на длину соответствующего участка. Таким образом, наращивая значение в переменной *hy*, мы будем получать координату *y* соответствующего участка, и размещать там точку:

```
hy=0
*DO,i,1,5
hy = hy + body(i,1)
K,,0,hy,0
*ENDDO
```

Полученные шесть точек необходимо соединить их линиями. Так как линии соединяют точки попарно, достаточно реализовать в цикле проведение линий между *i*-й и (*i*+1)-й точками:

```
*DO,i,1,5
LSTR,i,i+1
*ENDDO
```

Аналогично реализуется задание типа и размера поперечного сечения балки на разных участках:

```
*DO,i,1,5
SECTYPE, i, BEAM, RECT, , 0
SECOFFSET, CENT
```

```
SECDATA, 2*body(i, 2), body(i, 2)
*ENDDO
```

Теперь определим тип КЭ и свойства материала. Так как линейные размеры были заданы в миллиметрах, то модуль Юнга введём в МПа:

```
ET, 1, BEAM188      ! зададим элемент BEAM188
MPTEMP, 1, 0        ! зададим свойства материала
MPDATA, EX, 1, , 210000
MPDATA, PRXY, 1, , 0.3
```

Зададим на линиях необходимое количество КЭ:

```
Nmesh = 10
LESIZE, All, , , Nmesh, , , , 0
```

Теперь необходимо задать элементам на каждом участке соответствующие значения поперечного сечения. Это можно реализовать двумя способами.

*Первый способ.* Свяжем линии на каждом участке с заданным типом КЭ, материалом и поперечным сечением. Для этого можно воспользоваться командой **LATT**. В этом случае в цикле выбираем поочередно каждую линию и связываем её с заданным набором свойств, а затем разбиваем на КЭ-сетку все линии:

```
*DO, i, 1, 5
  LSEL, S, , , i
  LATT, 1, , 1, , , i
*ENDDO
ALLSEL, ALL
LMESH, All
```

*Второй способ.* В этом случае разбиение каждой линии происходит по очереди, но предварительно перед этим задается номер поперечного сечения, которое необходимо использовать при разбиении на КЭ-сетку:

```
TYPE, 1
MAT, 1
*DO, i, 1, 5
  SECNUM, i
  LMESH, i
*ENDDO
```



Для отображения сечений можно воспользоваться командой `/ESHAPE, 1.0` (на рис. 6.1.2. приведена иллюстрация ожидаемого результата):

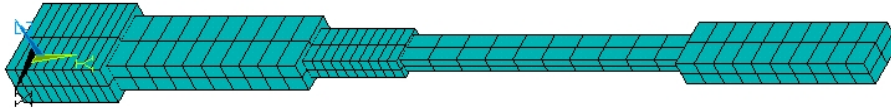


Рис. 6.1.2. КЭ сетка бруса.

Реализация ГУ в этой задаче проста: в первой точке необходимо зафиксировать все степени свободы, то есть три перемещения и три угла поворота, в последней точке, в нашем случае шестой, необходимо приложить сжимающую силу:

```
DK, 1, , , , 0, All, , , , , ,  
FK, 6, FY, -Load
```

После этого зададим тип решаемой задачи (задача статики) и запустим процесс решения:

```
/SOL  
/STATUS, SOLU  
SOLVE
```

Построим эпюру нормальных напряжений. Для этого создадим две таблицы, содержащие значение нормальных напряжений в I-ом и J-ом узлах КЭ. Согласно документации, для КЭ BEAM188 это параметр SDIR и его значения хранятся в SMISC 31 и 36. После сохранения этих результатов в таблицах выведем значения на экран (на рис. 6.1.3 приведена иллюстрация ожидаемого результата).

```
/POST1  
ETABLE, , SMISC, 31  
ETABLE, , SMISC, 36  
PLLS, SMIS31, SMIS36, 1, 0
```

Значения нормальных напряжений, приведенных на рис. 6.1.3, в МПа.

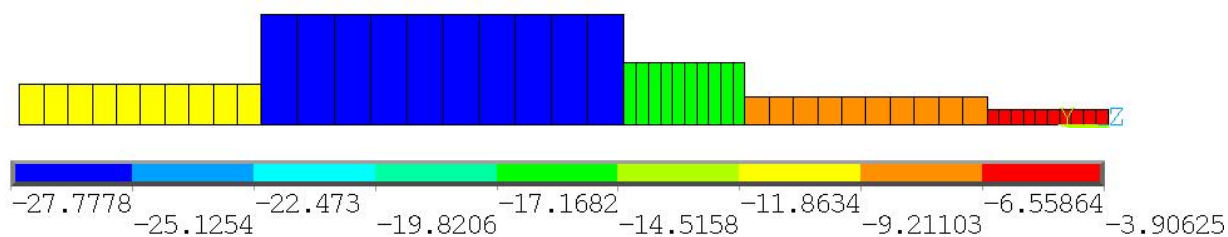


Рис. 6.1.3. Эпюра нормальных напряжений для бруса.

### Полный листинг программы

```
Nmesh = 10 ! Количество элементов
Load=500 ! Нагрузка
*DIM, body, ARRAY, 5, 2 ! Массив размеров
! Зададим размеры для бруса
body (1,1) = 20, 40, 20, 60, 40
body (1,2) = 8, 6, 4, 3, 5
/PREP7
! Построение точек
K,,0,0,0
hy=0
*DO,i,1,5
hy = hy + body(i,1)
K,,0,hy,0
*ENDDO
! Построение линий
*DO,i,1,5
LSTR,i,i+1
*ENDDO
! КЭ
ET, 1, BEAM188
! зададим свойства материала
MPTEMP,1,0
MPDATA,EX,1,,210000
MPDATA,PRXY,1,,0.3
! зададим тип и размеры поперечных сечений
*DO,i,1,5
SECTYPE, i, BEAM, RECT,, 0
SECOFFSET, CENT
SECDATA,2*body(i,2),body(i,2)
```

```

*ENDDO
! Разбиение на сетку
TYPE, 1
MAT, 1
LESIZE, All,,,Nmesh,,,,,0
*DO,i,1,5
    SECNUM, i
    LMESH, i
*ENDDO
! Граничные условия
DK,1,,,,0,All,,,,,
FK,6,FY,-Load
! Решение
/SOL
/STATUS,SOLU
SOLVE
! Эпюры
/POST1
! Построение эпюры
ETABLE,,SMISC, 31
ETABLE,,SMISC, 36
PLLS,SMIS31,SMIS36,1,0

```

## **Задача 2. Влияние на прогиб радиуса кривизны кривого бруса, нагруженного равномерным давлением**

Рассмотрим кривой брус, нагруженный внешним давлением. На рис. 6.2.1 приведена схема балки и ГУ. Поперечное сечение кривого бруса – прямоугольник со сторонами 5 мм и 2.5 мм. Допустимый диапазон радиуса кривизны – от  $R_0 = 50$  мм до  $R_1 = 52.25$  мм. Для построения зависимости вертикального перемещения левого края балки от радиуса кривизны необходимо провести некоторое число расчётов при разных радиусах кривизны, сохраняя значения радиуса кривизны и величину перемещения. Ручная реализация этой задачи трудоёмка и продолжительна по времени, тем более, если итерационный шаг по радиусу кривизны мал. Рассмотрим вариант автоматизированной реализации этой процедуры с помощью командных и программных средств.

Табл. 6.2.1.

Номер итерации	1	2	...	9	10
Радиус кривизны, мм.	50	50.25	...	52	52.25

Нагрузка  $p = 50$  МПа. Механические свойства материала: модуль Юнга – 210 ГПа, коэффициент Пуассона – 0.3.

В данном примере нас будут интересовать перемещения в скользящей опоре при изменении радиуса кривизны.

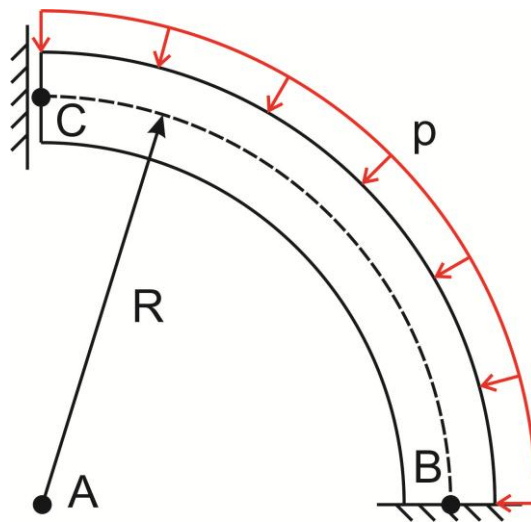


Рис. 6.2.1. Схема нагружения кривого бруса.

### *Ввод исходных данных*

В первую очередь зададим параметры: величина нагрузки  $P$ , радиус кривизны  $R$ , количество КЭ на линии Nmesh и количество итераций NStep:

```
P=50
R=50
Nmesh=20
NStep=10
```

Введем одномерный массив UYRes, размерность которого будет зависеть от количества итераций NStep ( $i$ -ое значение массива будет хранить перемещение в скользящей опоре на соответствующей итерации):

```
*DIM,UYRes,array,NStep
```

Перейдем к построению геометрии. Для этого зайдём в препроцессор и создадим три точки: первая точка в начале СК будет центром кривизны, вторые две точки будут началом и концом кривого бруса.

```
/PREP7
K,, 0, 0, 0
K,, R, 0, 0
K,, 0, R, 0
```

Теперь определим тип КЭ и свойства материала. Так как линейные размеры были заданы в миллиметрах, то модуль Юнга введём в МПа:

```
ET,1,BEAM188      ! зададим элемент BEAM188
MPTEMP,1,0        ! зададим свойства материала
MPDATA,EX,1,,210000
MPDATA,PRXY,1,,0.3
```

Зададим тип и размеры поперечного сечения бруса:

```
SECTYPE, 1, BEAM, RECT, , 0
SECOFFSET, CENT
SECDATA,2.5,5.0,0,0,0,0,0,0,0,0,0,0
```

Реализация ГУ: во второй точке необходимо зафиксировать все степени свободы: три перемещения и три угла поворота; в третьей – перемещения по осям  $Ox$  и  $Oz$  и три угла поворота:

```
DK,2, , , ,0,All, , , , ,
DK,3, , , ,0,UX, , UZ, ROTX, ROTY, ROTZ,
```

Для автоматизации решения задачи используем оператор цикла. В теле цикла будет происходить перестроение геометрии и КЭ-сетки, а также запись интересующих результатов – вертикальных перемещений в третьей точке. Параметр цикла  $ii$  будет меняться от 0 до  $NStep-1$ :

```
*DO,ii,0,NStep-1
```

Актуальный радиус кривизны может быть подсчитан по величине  $ii$ . Шаг изменения радиуса кривизны равен 0.5% от начального радиуса кривизны, то есть 0.25 мм. Для построения дуги воспользуемся командой LARC:

```
LARC, 2, 3, 1, R*(1+ii/200)
```

Зададим на линиях необходимое количество КЭ, материал и поперечное сечение:

```
TYPE, 1
MAT, 1
LESIZE, All, , ,Nmesh, , , ,0
SECNUM, 1
LMESH, All
```

Для отображения сечений используем команду /ESHAPE,1.0 (на рис. 6.6.2. приведена иллюстрация ожидаемого результата).

Нагрузка в виде давления прикладывается на всю линию:

```
SFBEAM,ALL,2,PRES,-P,,,,,0
```

После этого зададим тип решаемой задачи (задача статики) и запустим процесс решения:

```
/SOL  
/STATUS,SOLU  
SOLVE
```

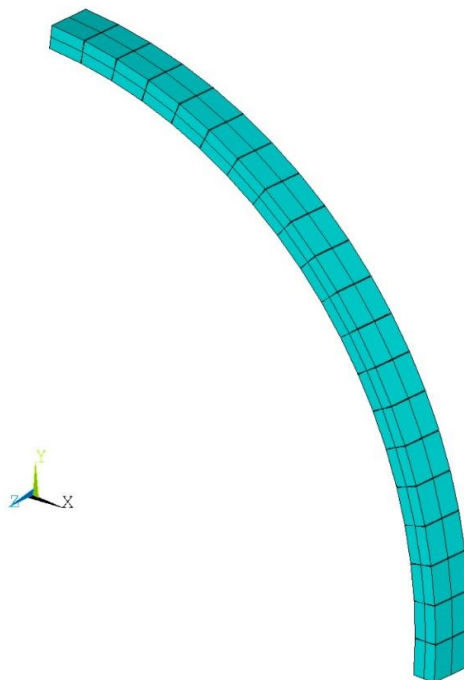


Рис. 6.2.2. КЭ сетка балки.

Для записи перемещений в интересующей нас точке, в данном случае третьей, воспользуемся командой **\*GET**. Для этого необходимо определить номер узла, который расположен в указанной точке. Так как нумерация узлов в общем случае нам неизвестна, определим этот узел с помощью выборки. Для этого сначала выберем третью точку, а затем узел, который привязан к выбранной точке:

```
KSEL,S,,,3  
NSEL,ALL  
NSLK,R
```

Для определения вертикального перемещения в заданном узле также воспользуемся командой **\*GET**. Определим переменную для записи результата – элемент массива UYRes, тип объекта – узел (NODE). Номер узла определим с помощью команды NDNEXT(0): в этом случае из выборки будет выбран узел, номер которого идет следующим после нулевого. Тип извлекаемых данных: перемещения (U) по вертикальной оси (Y). Запишем команду:

```
*GET, UYRes (ii+1) , Node, NDNEXT (0) , U, Y
```

После того как результат сохранен, необходимо подготовить модель к следующей итерации, а именно: выбрать всю модель, очистить КЭ-сетку и удалить дугу:

```
ALLSEL, ALL
/PREP7
LCLEAR, ALL
LDELETE, All, , , 0
```

После этого можно закрыть цикл:

```
*ENDDO
```

### *Полный листинг программы*

```
R=50
Nmesh=20
NStep=10
*DIM, UYRes, array, NStep
```

```
/PREP7
K, , 0, 0, 0
K, , R, 0, 0
K, , 0, R, 0
```

```
! КЭ
ET, 1, BEAM188
```

```
! зададим свойства материала
MPTEMP, 1, 0
MPDATA, EX, 1, , 210000
MPDATA, PRXY, 1, , 0.3
```

```
! зададим тип и размеры поперечных сечений
```



```
SECTYPE, 1, BEAM, RECT, , 0  
SECOFFSET, CENT  
SECDATA, 2.5, 5.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
DK, 2, , , , 0, All, , , , ,  
DK, 3, , , , 0, UX, , UZ, ROTX, ROTY, ROTZ,
```

```
*DO, ii, 0, NStep-1
```

```
  LARC, 2, 3, 1, R*(1+ii/200)
```

```
  TYPE, 1  
  MAT, 1  
  LESIZE, All, , , Nmesh, , , , 0  
  SECNUM, 1  
  LMESH, All
```

```
  /ESHAPE, 1.0
```

```
  SFBEAM, ALL, 2, PRES, -50, , , , , 0
```

```
  /SOL  
  /STATUS, SOLU  
  SOLVE
```

```
  KSEL, S, , , 3  
  NSEL, ALL  
  NSLK, R
```

```
  *GET, UYRes(ii+1), Node, NDNEXT(0), U, Y  
  ALLSEL, ALL
```

```
  /PREP7  
  LCLEAR, ALL  
  LDELETE, All, , , 0  
*ENDDO
```

```
*DIM, PlotRes, table, NStep, 2  
PlotRes(1, 2) = UYRes
```

### Задача 3. Расчёт рамы под действием неравномерной нагрузки

Рассмотрим раму под действием давления, заданного по определённому закону. На рис. 6.3.1 приведена схема рамы и ГУ. Поперечное сечение балок АВ, ВС, DE, EF – прямоугольник со сторонами 5 мм и 3 мм. Поперечное сечение балки CD – двутавр высотой 7 мм, толщиной стенки 3 мм, шириной полки 5 мм, толщиной полки 2 мм. Длина  $L_1 = 40$  мм,  $L_2 = 30$  мм. Радиус кривизны R кривого бруса CD равен 10 мм.

Распределенная нагрузка по синусоидальному закону действует на брус CD по закону:

$$F = -P \cdot \sin(\alpha)$$

где  $\alpha$  – угол, определяющий точку на дуге.

Максимальная вертикальная нагрузка составляет:  $F_{\max} = 2000$  Н. Рассмотрим вариант автоматизированной реализации приложения нагрузки, распределенной по раме по синусоидальному закону, с помощью командных и программных средств.

Механические свойства материала: модуль Юнга – 210 ГПа, коэффициент Пуассона – 0.3.

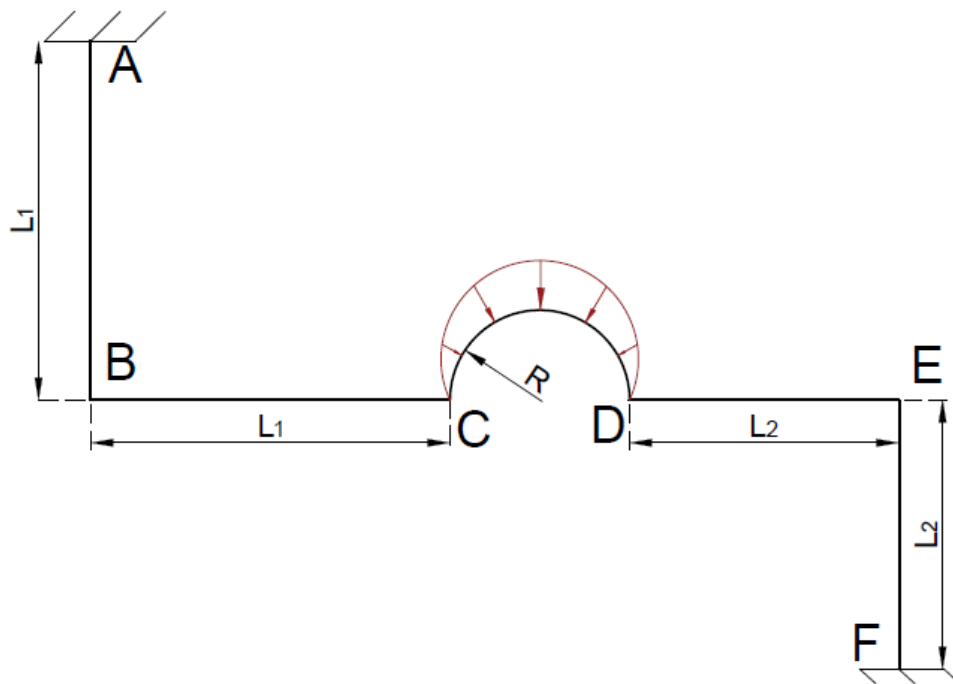


Рис. 6.3.1. Схема нагружения рамы.

### *Ввод исходных данных*

В первую очередь зададим параметры: величина максимальной вертикальной нагрузки Load и количество КЭ на линии Nmesh:

```
Nmesh=20
Load=2000
```

Перейдем к построению геометрии: для этого зайдём в препроцессор и создадим восемь точек. Первая точка в начале СК – центр кривизны балки CD:

```
/PREP7
K,, 0, 0, 0
K,, 10, 0, 0
K,, 0, 10, 0
K,, -10, 0, 0
K,, 40, 0, 0
K,, 40, -30, 0
K,, -30, 0, 0
K,, -30, 40, 0
```

Для построения дуги воспользуемся командой **LARC**:

```
LARC, 2, 3, 1, 10  
LARC, 3, 4, 1, 10
```

Затем соединим все точки линиями:

```
LSTR, 2, 5  
LSTR, 5, 6  
LSTR, 4, 7  
LSTR, 7, 8
```

Следующим шагом определим тип КЭ. Укажем вторую степень для функций формы KEYOPT(3)=2, а также зададим свойства материала: так как введенные линейные размеры были в миллиметрах, то модуль Юнга введём в МПа:

```
ET, 1, BEAM188  
KEYOPT, 1, 3, 2  
MPTEMP, 1, 0  
MPDATA, EX, 1, , 210000  
MPDATA, PRXY, 1, , 0.3
```

Зададим тип и размеры поперечного сечения бруса:

```
SECTYPE, 1, BEAM, I, , 0  
SECOFFSET, CENT  
SECDATA, 5, 5, 7, 2, 2, 3, 0, 0, 0, 0, 0, 0
```

```
! Прямоугольник  
SECTYPE, 2, BEAM, RECT, , 0  
SECOFFSET, CENT  
SECDATA, 3, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

Рассмотрим другой способ задания поперечного сечения линиям, для этого создадим отдельно компоненты ArcL и BeamL: ArcL – для хранения линий с двутавровым поперечным сечением, BeamL – для прямоугольного сечения.

```
LSEL, S, , , 1, 2  
CM, ArcL, Lines
```

```
LSEL, S, , , 3, 6  
CM, BeamL, Lines
```

Свяжем линии на каждом участке с заданным типом КЭ, материалом, поперечным сечением и построим КЭ сетку. На данном этапе с помощью команды **LATT** ориентируем элементы двутавра по первой точке.

```
CMSEL, S, ArcL
LESIZE, All, , , Nmesh/2, , , , 1
LATT, 1, , 1, , 1, 1, 1
```

```
CMSEL, S, BeamL
LESIZE, All, , , Nmesh, , , , 1
LATT, 1, , 1, , , , 2
```

```
ALLSEL, ALL
LMESH, All
```

На рис. 6.3.2 приведена иллюстрация разбиения конструкции на КЭ сетку с заданными сечениями.

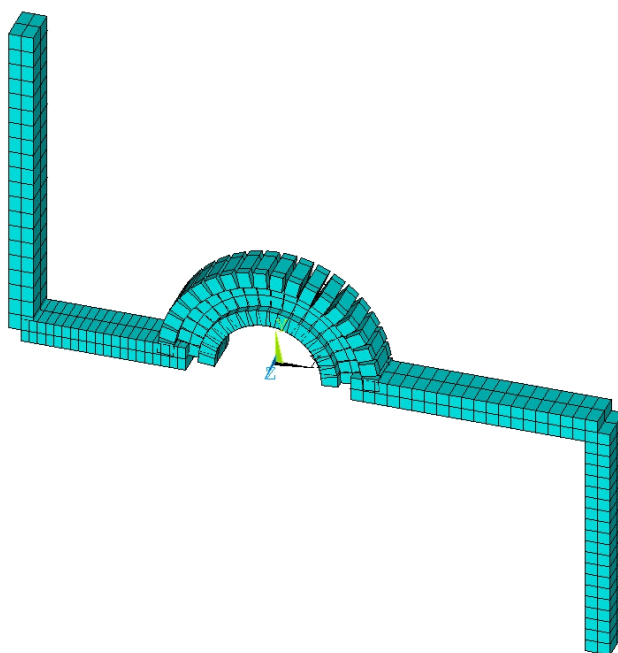


Рис. 6.3.2. КЭ сетка конструкции.

Реализация ГУ: в шестой и восьмой точках необходимо зафиксировать все степени свободы: три перемещения и три угла поворота:

```
DK, 6, , , , 0, All, , , , , ,
```

```
DK, 8, , , , 0, All, , , , , ,
```

Выберем линии балки CD, а затем все узлы, им принадлежащие:

```
CMSEL, S, ArcL  
NSEL, ALL  
NSLL, R, 1
```

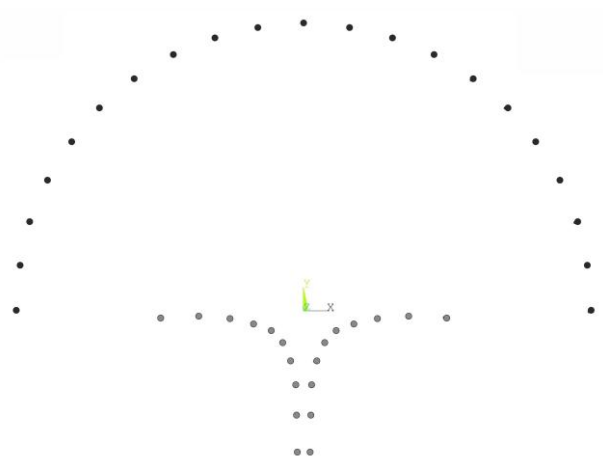


Рис. 6.3.3. Узлы.

На рис. 6.3.3 показаны выбранные узлы. Стоит отметить узлы, отмеченные серым (находящиеся в нижней полуплоскости) – это узлы К, необходимые для ориентации локальной СК элементов. Если приложить после этого ко всей выборке силы, то и к этим узлам будут приложены нагрузки, которых в рассчитываемой конструкции нет. Этого можно избежать, сделав выборку необходимых узлов по координатам, например, в цилиндрической СК:

```
CSYS, 1  
NSEL, S, LOC, X, 10, 10  
CSYS, 0
```

Запишем в nNodes с помощью команды **\*GET** количество узлов и введём переменную nd, с начальным значением равным нулю:

```
nd=0  
*GET, nNodes, NODE, 0, COUNT
```

Откроем цикл, в котором i будет изменяться от 1 до nNodes:

```
*DO, i, 1, nNodes
```

Идея заключается в том, что в цикле на каждом шаге будет выбираться только один узел. Для этого узла будет определяться угол  $\alpha$  и нагрузка (в зависимости от величины угла).

На каждом шаге цикла в переменную nd запишем номер следующего узла:

```
nd=NDNEXT (nd)
```

Теперь надо определить и записать угол  $\alpha$ . Используем цилиндрическую СК и команду **\*GET**:

```
CSYS, 1  
*GET, Angle, NODE, nd, LOC, Y  
CSYS, 0
```

Нагрузка прикладывается в узел по закону:

```
F, nd, FY, -Load*Sin (Angle*Pi/180)
```

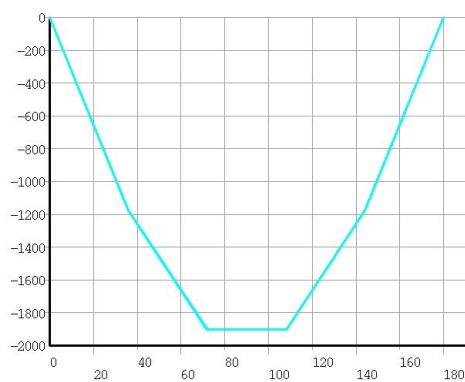
После этого можно закрыть цикл и выбрать всю расчетную область:

```
*ENDDO  
ALLSEL, ALL
```

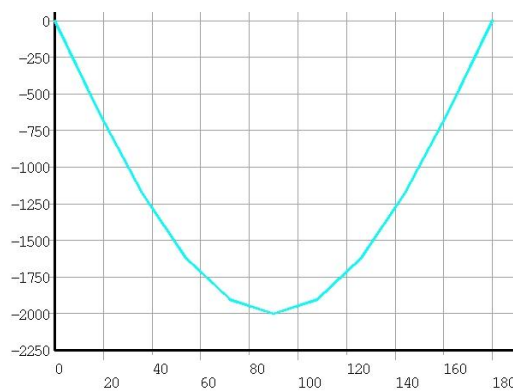
*Другой способ* задания нагрузки – использование табличной функции. Для этого сначала необходимо задать табличную функцию по переменной Y цилиндрической СК:

```
Nt=15  
dA=180/Nt  
*DIM, sinLoad, TABLE, Nt+1, , , Y, , , 1  
*DO, i, 0, Nt  
    sinLoad(i+1, 0)=i*dA  
    sinLoad(i+1, 1)=-Load*Sin(i*dA*PI/180)  
*ENDDO
```

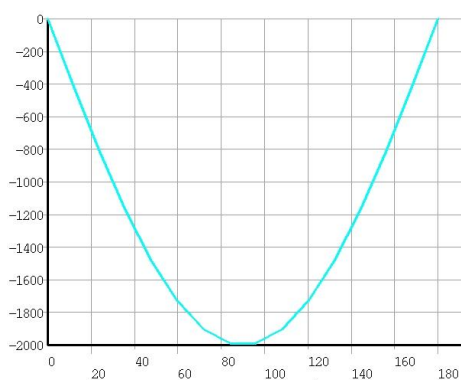
Здесь Nt – количество интервалов для табуляции функции.



а



б



в

Рис. 6.3.4. Табуляция функции нагрузки: а – 5 интервалов, б – 10 интервалов, в – 15 интервалов.

Приложение нагрузки в этом случае можно реализовать в виде:

```
F,ALL,FY,%sinLoad%
```

Зададим тип решаемой задачи (задача статики), и запустим процесс решения:

```
/SOL
/STATUS,SOLU
SOLVE
```

Построим эпюры изгибающих моментов относительно оси  $Oz$ :

```
/POST1
ETABLE,MzI,SMISC, 3
ETABLE,MzJ,SMISC, 16
PLLS,MzI,MzJ,1,0
```



и относительно оси  $Oy$ :

```
ETABLE,MyI,SMISC, 2  
ETABLE,MyJ,SMISC, 15  
PLLS,MyI,MyJ,1,0
```

Стоит отметить, что при выводе эпюры изгибающих моментов относительно оси  $Oz$  дуга будет без данных (см. рис. 6.3.5а), а при выводе эпюры изгибающих моментов относительно оси  $Oy$ , наоборот, величины будут только на дуге (см. рис. 6.3.5б).

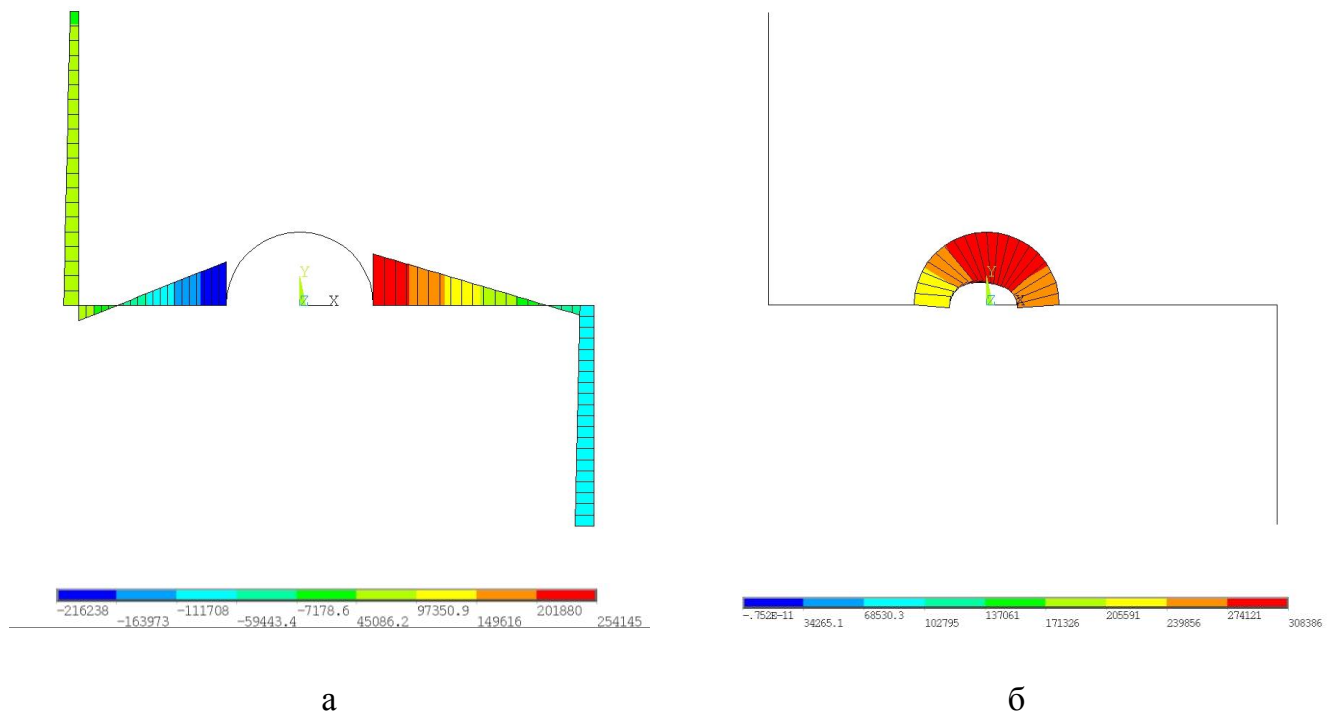


Рис. 6.3.5. Эпюры изгибающих моментов: а – относительно оси  $z$ , б – относительно оси  $y$ .

Это связано с тем, что на дуге локальная СК элемента была повернута и, соответственно, индексация моментов изменилась, хотя все КЭ конструкции изгибаются в плоскости чертежа.

#### *Полный листинг программы*

```
PI=ACOS(-1)  
Nmesh=20  
NStep=10  
Load=2000
```

```

! Приложение силы с помощью таблицы [
!Nt=15
!dA=180/Nt
!*DIM,sinLoad,TABLE,Nt+1,,,Y,,,1
!*Do,i,0,Nt
!   sinLoad(i+1,0)=i*dA
!   sinLoad(i+1,1)=-Load*Sin(i*dA*PI/180)
!*ENDDO
! ] Приложение силы с помощью таблицы

```

```

/PREP7
K,, 0, 0, 0
K,, 10, 0, 0
K,, 0, 10, 0
K,, -10, 0, 0
K,, 40, 0, 0
K,, 40,-30, 0
K,, -30, 0, 0
K,, -30, 40, 0

```

```

LARC, 2, 3, 1, 10
LARC, 3, 4, 1, 10
LSTR,2,5
LSTR,5,6
LSTR,4,7
LSTR,7,8

```

```

! КЭ
ET,1,BEAM188
KEYOPT,1,3,2
! зададим свойства материала
MPTEMP,1,0
MPDATA,EX,1,,210000
MPDATA,PRXY,1,,0.3

```

```

! зададим тип и размеры поперечных сечений
! Двутавр
SECTYPE, 1, BEAM, I, , 0
SECOFFSET, CENT
SECDATA,5,5,7,2,2,3,0,0,0,0,0,0

```

```

! Прямоугольник
SECTYPE, 2, BEAM, RECT, , 0

```

```
SECOFFSET, CENT  
SECDATA, 3, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
! Компоненты  
! Арка  
LSEL, S, , , 1, 2  
CM, ArcL, Lines  
! Остальное  
LSEL, S, , , 3, 6  
CM, BeamL, Lines
```

```
! Разметка  
! проблема с узлами K у арки  
CMSEL, S, ArcL  
LESIZE, All, , , Nmesh/2, , , , 1  
LATT, 1, , 1, , 1, 1, 1
```

```
CMSEL, S, BeamL  
LESIZE, All, , , Nmesh, , , , 1  
LATT, 1, , 1, , , , 2
```

```
ALLSEL, ALL  
LMESH, All
```

```
DK, 6, , , , 0, All, , , , ,  
DK, 8, , , , 0, All, , , , ,
```

```
CMSEL, S, ArcL  
NSEL, ALL  
NSLL, R, 1
```

```
CSYS, 1  
NSEL, S, LOC, X, 10, 10  
CSYS, 0
```

```
! Приложение силы без таблицы [  
nd=0  
*GET, nNodes, NODE, 0, COUNT  
*Do, i, 1, nNodes  
nd=NDNEXT(nd)  
CSYS, 1  
*GET, Angle, NODE, nd, LOC, Y  
CSYS, 0
```

```
F,nd,FY,-Load*Sin(Angle*Pi/180)
*ENDDO
! ] Приложение силы без таблицы
```

```
! Приложение силы с помощью таблицы [
! F,ALL,FY,%sinLoad%
! ] Приложение силы с помощью таблицы
```

```
ALLSEL,ALL
```

```
/SOL
/STATUS,SOLU
SOLVE
```

```
/POST1
! Эпюра моментов Mz
ETABLE,MzI,SMISC, 3
ETABLE,MzJ,SMISC, 16
PLLS,MzI,MzJ,1,0
! Эпюра моментов My
ETABLE,MyI,SMISC, 2
ETABLE,MyJ,SMISC, 15
PLLS,MyI,MyJ,1,0
```

#### Задача 4. Определение массово-прочностных характеристик конструкции

Рассмотрим кольцо D, опёртое на три балки:  $B_1C_1$ ,  $B_2C_2$ ,  $B_3C_3$  (рис. 6.4.1). В точках В балки жестко защемлены. На кольцо установлена некоторая деталь заданной массы: в модели заменим влияние детали распределенным давлением, которое эквивалентно массе детали и может быть рассчитано по формуле:

$$p = \frac{F}{\pi \cdot D \cdot A_p},$$

где:  $p$  – внешнее давление,  $F$  – вес детали,  $D$  – диаметр кольца,  $A_p$  – площадь опирания детали.

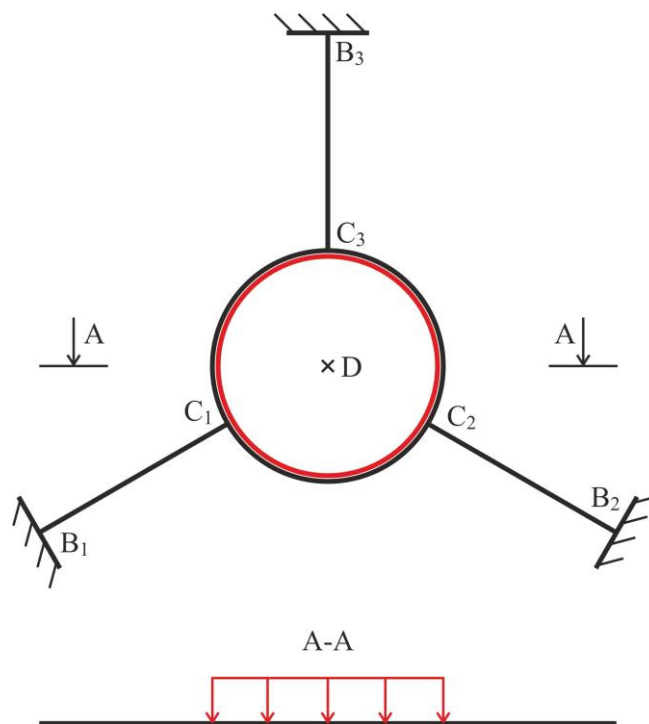


Рис. 6.4.1. Расчетная схема.

Поперечное сечение балок и кольца – круг площадью 2.5 мм. Необходимо определить максимальные напряжения и вес конструкции при различных

диаметрах кольца в диапазоне от 20 мм до 100 мм. Механические свойства материала: модуль Юнга – 210 ГПа, коэффициент Пуассона – 0.3.

#### *Ввод исходных данных*

Зададим параметры: число  $\pi$ , изначальную длину балок  $L$ , диаметр кольца  $D$ , количество КЭ на линии  $N$ , площадь сечения  $Area$  и величину нагрузки  $pValue$ :

```
PI=ACOS (-1)
L=100
D=20
N=10
Area=2.5
pValue=1000
```

Введем параметр  $h$  – шаг изменения диаметра кольца:

```
h=10
```

Далее определим тип КЭ и свойства материала. Так как линейные размеры были заданы в миллиметрах, то модуль Юнга введём в МПа:

```
ET, 1, BEAM188
KEYOPT, 1, 3, 0
MPTEMP, 1, 0
MPDATA, EX, 1, , 210000
MPDATA, PRXY, 1, , 0.3
```

Зададим тип и размеры поперечного сечения:

```
SECTYPE, 1, BEAM, CSOLID, Cir, 0
SECOFFSET, CENT
SECDATA, Area, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

#### *Тело цикла*

Как уже было отмечено выше, диаметр кольца будет изменяться в диапазоне от 20 мм до 100 мм с шагом 10 мм, поэтому введём еще один параметр и реализуем цикл:

```
NDIM= ( (100-20) /h) +1
*DO, j, 1, NDIM
/PREP7
```

В теле цикла определяются максимальные напряжения и веса конструкции. Параметр цикла  $j$  меняется в пределах от 1 до NDM.

Перейдем к построению геометрии: для этого зайдём в препроцессор и создадим четыре точки. Первая точка соответствует началу СК (точка D), вторая, третья и четвертая – точкам  $C_1$ ,  $C_2$  и  $C_3$  (рис. 6.4.1) соответственно. Затем создаём окружность с центром в точке 1 и радиусом  $D/2$ :

```
K, , 0, 0, 0
K, , 0, L, 0
K, , -L*Cos (Angle) , -L*Sin (Angle) , 0
K, , L*Cos (Angle) , -L*Sin (Angle) , 0
CIRCLE, 1, D/2
```

Соединяем точки 2, 3, 4 линиями с точкой, находящейся в начале СК:

```
LSTR, 1, 2
LSTR, 1, 3
LSTR, 1, 4
```

Разделяем линии 6 и 7 линиями 3 и 4. Первоначальные линии 6 и 7 сохраняем.

```
LSBL, 6, 3, , Keep, Keep
LSBL, 7, 4, , Keep, Keep
```

Затем разделяем линии 5, 3, 4 по точкам 6, 9, 10 соответственно:

```
LDIV, 5, , 6
LDIV, 3, , 9
LDIV, 4, , 10
```

Удаляем ненужные линии и объединяем все геометрические примитивы, совпадающие друг с другом:

```
LDELE, 5, , , 1
LDELE, 6, , , 1
LDELE, 7, , , 1
LDELE, 8, , , 1
LDELE, 10, , , 1
NUMMRG, ALL
```

Зададим на линиях 9, 11, 12 N (количество КЭ). Остальные линии будут разбиты автоматически:

```

LESIZE, 9, , , N, , , , 0
LESIZE, 11, , , N, , , , 0
LESIZE, 12, , , N, , , , 0
LMESH, ALL

```

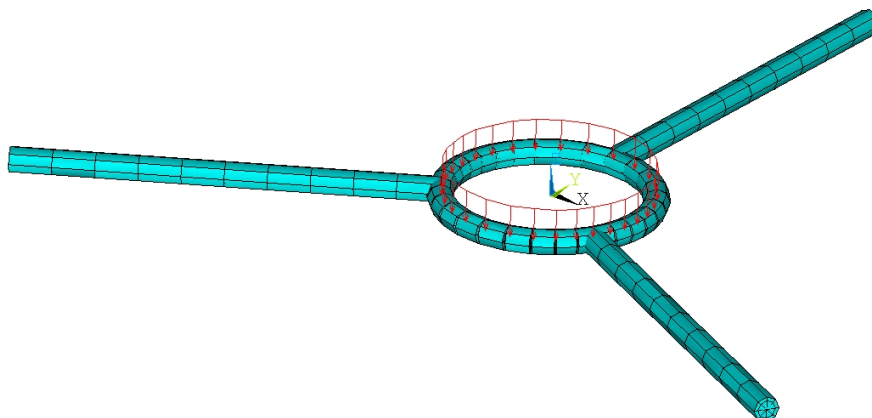


Рис. 6.4.2. КЭ модель.

Реализация ГУ (шарнирная опора): во второй, третьей и четвертой точках необходимо зафиксировать все степени свободы:

```

DK, 2, , , , 0, ALL
DK, 3, , , , 0, ALL
DK, 4, , , , 0, ALL

```

Формула для пересчёта действующего давления примет вид:

$$\frac{pValue}{\pi \cdot D \cdot Area}.$$

Выберем линии, образующие кольцо, и приложим к ним давление:

```

LSEL, S, , , 1, 4, 1, 0
LSEL, A, , , 13, 14, 1, 0
ESEL, ALL
ESLL, R
SFBEAM, ALL, 1, PRES, pValue/PI/D/Area, , , , , 0
ALLSEL, ALL
FINISH

```

Зададим тип решаемой задачи (задача статики) и запустим процесс решения.



```
/SOL  
/STATUS,SOLU  
SOLVE  
FINISH  
/POST1
```

После того как задача определения НДС решена, необходимо найти максимальное напряжение в конструкции и её вес. Так как у всех деталей конструкции плотность и поперечное сечение одинаковые, то достаточно найти общую длину деталей.

### *Определение максимальных напряжений*

Через команду **\*GET** запишем в ElemNum количество всех КЭ:

```
*GET, ElemNum, ELEM, 0, COUNT
```

Затем открываем цикл, в теле которого будет происходить определение напряжения в КЭ, а затем и определение наибольшего напряжения во всей конструкции. Параметр цикла *i* меняется в пределах от 1 до ElemNum.

Создадим массив StrTEMP размерностью 6x1, в который будем записывать напряжения (изгибные, осевые, нормальные) КЭ в *i*-ом и *j*-ом узле:

```
*DIM, StrTEMP, ARRAY, 6, 1, 1, , ,  
*DO, i, 1, ElemNum
```

Введем счётчик *k*, который на каждой итерации будет перезаписываться следующим номером КЭ. Перед циклом необходимо обнулить значение счётчика.

```
k = ELNEXT(k)
```

С помощью команды **\*GET** записываем в массив изгибные и осевые напряжения в *i*-ом и *j*-ом узле. Для реализации необходимо обратиться к документации команды и элемента BEAM188:

```
*GET, StrTEMP(1), ELEM, 1, SMISC, 32  
*GET, StrTEMP(2), ELEM, 1, SMISC, 37  
*GET, StrTEMP(3), ELEM, 1, SMISC, 34  
*GET, StrTEMP(4), ELEM, 1, SMISC, 39
```

Суммируем осевые и изгибные напряжения в *i*-ом и *j*-ом узле:

```
StrTEMP(5)=StrTEMP(1)+StrTEMP(3)  
StrTEMP(6)=StrTEMP(2)+StrTEMP(4)
```

Определим наибольшие (по абсолютному значению) нормальные напряжения:

```
*DO, ii, 5, 6  
  *IF, StressMax, ABLT, StrTEMP(ii), THEN  
    StressMax = StrTEMP(ii)  
  *ENDIF  
*ENDDO
```

Цикл перебора всех КЭ закрывается:

```
*ENDDO
```

### *Определение длины деталей конструкции*

Для определения суммарной длины деталей конструкции воспользуемся командой **\*GET**. Переменная LenTEMP будет содержать сумму всех длин, а LenTEMPG – длину одной линии.

Сначала присвоим LenTEMP значение ноль. Длину линии 9 запишем в LenTEMPG. Пересчитаем LenTEMP как сумму его прошлого значения с LenTEMPG. Аналогично повторяются данные действия для линии 11 и 12.

```
LenTEMP=0  
*GET, LenTEMPG, LINE, 9, LENG  
LenTEMP = LenTEMP + LenTEMPG  
*GET, LenTEMPG, LINE, 11, LENG  
LenTEMP = LenTEMP + LenTEMPG  
*GET, LenTEMPG, LINE, 12, LENG  
LenTEMP = LenTEMP + LenTEMPG
```

Для учета длины окружности кольца прибавим к LenTEMP значение  $\pi \cdot D$ :

```
LenTEMP = LenTEMP + Pi*D
```

Создадим массив OUTData размерностью NDIM x 2. В него будут записываться результаты расчета на каждой итерации.

```
OUTData(j,1)=LenTEMP  
OUTData(j,2)=StressMax
```

После записи результатов выбирается вся модель, удаляется КЭ-сетка и вся геометрия. Значение диаметра увеличивается на параметр  $h$ . Цикл, который открывался перед построением геометрии, закрывается.

```
/PREP7  
ALLSEL, ALL  
LCLEAR, ALL  
LDELETE, All, , , 1  
D = D + h
```

### Макросы

Для упрощения кода разумно вынести в отдельный макрос команды, связанные с построением геометрии, сетки, решением и сбором результатов. Для этого создадим файл *geometry.mcr* и скопируем туда необходимые команды. Тогда основной цикл примет вид:

```
*DO, j, 1, NDIM  
  /INPUT, geometry, mcr  
  D = D + h  
*ENDDO
```

Для записи результатов в отдельный txt-файл воспользуемся командой **\*VWRITE**. Команда **\*CFOPEN** открывает для записи файл *output.txt*. Команда **\*CFCLOS** закрывает файл. Указанные команды необходимо оформить в виде макроса, который можно создать средствами APDL:

```
*CREATE, OutSave  
*CFOPEN, 'output', 'txt',  
*VWRITE, OUTData(1,1), OUTData(1,2)  
(F9.3, F9.3)  
*CFCLOS  
*END
```

Для записи информации в файл следует вызвать созданный макрос:

```
/INPUT, OutSave
```

При выборе КЭ была задана линейная функция формы (KEYOPT(3)=0), ниже на рис. 6.4.3 приведены результаты расчётов для этого случая; а также квадратичной (KEYOPT(3)=2) и кубической (KEYOPT(3)=3) функций формы.

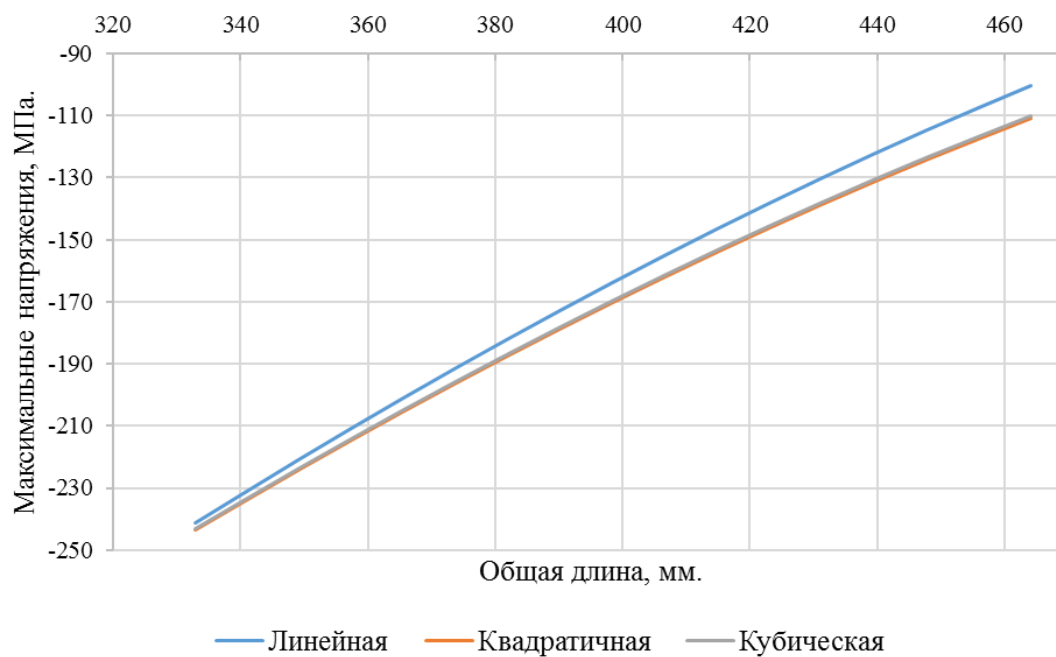


Рис. 6.4.3. Результаты расчетов.

### *Полный листинг программы*

Основной макрос:

```
/CWD, <DirName>
```

```
*CREATE, OutSave
*CFOPEN, 'output', 'txt',
*VWRITE, OUTData(1,1), OUTData(1,2)
(F9.3, F9.3)
*CFCLOS
*END
```

```
/PREP7
! Параметры задачи
PI=ACOS(-1)
L=100
D=20
N=10
Area=2.5
Angle=30*PI/180
pValue=1000
```

```
h=10
NDIM=((100-20)/h)+1
*DIM, StrTEMP, ARRAY, 6, 1, 1, , ,
*DIM, OUTData, ARRAY, NDIM, 2, 1, , ,
```

```
! КЭ
ET, 1, BEAM188
!KEYOPT, 1, 3, 0
KEYOPT, 1, 3, 2
```

```
! зададим свойства материала
MPTEMP, 1, 0
MPDATA, EX, 1, , 210000
MPDATA, PRXY, 1, , 0.3
```

```
! зададим тип и размеры поперечного сечения
SECTYPE, 1, BEAM, CSOLID, Cir, 0
SECOFFSET, CENT
SECDATA, Area, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
*DO, j, 1, NDIM
  /INPUT, geometry, mcr
  D = D + h
*ENDDO
```

```
/INPUT, OutSave
```

Дополнительный макрос (*geometry.mcr*):

```
/PREP7

K, , 0, 0, 0
K, , 0, L, 0
K, , -L*cos(Angle), -L*sin(Angle), 0
K, , L*cos(Angle), -L*sin(Angle), 0

CIRCLE, 1, D/2

LSTR, 1, 2
LSTR, 1, 3
LSTR, 1, 4

LSBL, 6, 3, , Keep, Keep
LSBL, 7, 4, , Keep, Keep
```

```
LDIV, 5, , 6
LDIV, 3, , 9
LDIV, 4, , 10
```

```
LDELE, 5, , , 1
LDELE, 6, , , 1
LDELE, 7, , , 1
LDELE, 8, , , 1
LDELE, 10, , , 1
```

```
NUMMRG, ALL
```

```
! Разметка сетки
LESIZE, 9, , , N, , , , , 0
LESIZE, 11, , , N, , , , , 0
LESIZE, 12, , , N, , , , , 0
LMESH, ALL
```

```
! Опоры
DK, 2, , , , 0, ALL
DK, 3, , , , 0, ALL
DK, 4, , , , 0, ALL
```

```
! Приложение нагрузок
LSEL, S, , , 1, 4, 1, 0
LSEL, A, , , 13, 14, 1, 0
ESEL, ALL
ESLL, R
SFBEAM, ALL, 1, PRES, pValue/PI/D/Area, , , , , 0
ALLSEL, ALL
FINISH
```

```
/SOL
/STATUS, SOLU
SOLVE
```

```
FINISH
/POST1
```

```
*GET, ElemNum, ELEM, 0, COUNT
k=0
StressMax_EL=0
```

```

StressMax=0
*DO,i,1,ElemNum
    k = ELNEXT(k)
    *GET,StrTEMP(1),ELEM,1,SMISC,32
    *GET,StrTEMP(2),ELEM,1,SMISC,37
    *GET,StrTEMP(3),ELEM,1,SMISC,34
    *GET,StrTEMP(4),ELEM,1,SMISC,39
    StrTEMP(5)=StrTEMP(1)+StrTEMP(3)
    StrTEMP(6)=StrTEMP(2)+StrTEMP(4)

    *DO,ii,5,6
        *IF, StressMax,ABLT,StrTEMP(ii),THEN
            StressMax = StrTEMP(ii)
        *ENDIF
    *ENDDO
*ENDDO

LenTEMP=0
*GET, LenTEMPG, LINE, 9, LENG
LenTEMP = LenTEMP + LenTEMPG
*GET, LenTEMPG, LINE, 11, LENG
LenTEMP = LenTEMP + LenTEMPG
*GET, LenTEMPG, LINE, 12, LENG
LenTEMP = LenTEMP + LenTEMPG
LenTEMP = LenTEMP + Pi*D

OUTData(j,1)=LenTEMP
OUTData(j,2)=StressMax

/PREP7
ALLSEL,ALL
LCLEAR,ALL
LDELETE, All, , , 1

```

### Задача 5. Расчёт фермы из разномодульного материала

Рассмотрим ферму (рис. 6.5.1):  $L=100$  мм. Площадь поперечного сечения фермы –  $20 \text{ мм}^2$ . В точке А действует сила  $F = (1000, -1000) \text{ Н}$ . В крайних точках жёсткая заделка. Ферма сделана из разномодульного материала: модуль Юнга на растяжение –  $210 \text{ ГПа}$ , модуль Юнга на сжатие –  $90 \text{ ГПа}$ , коэффициент Пуассона –  $0.3$ . Цель расчёта – определить НДС фермы.

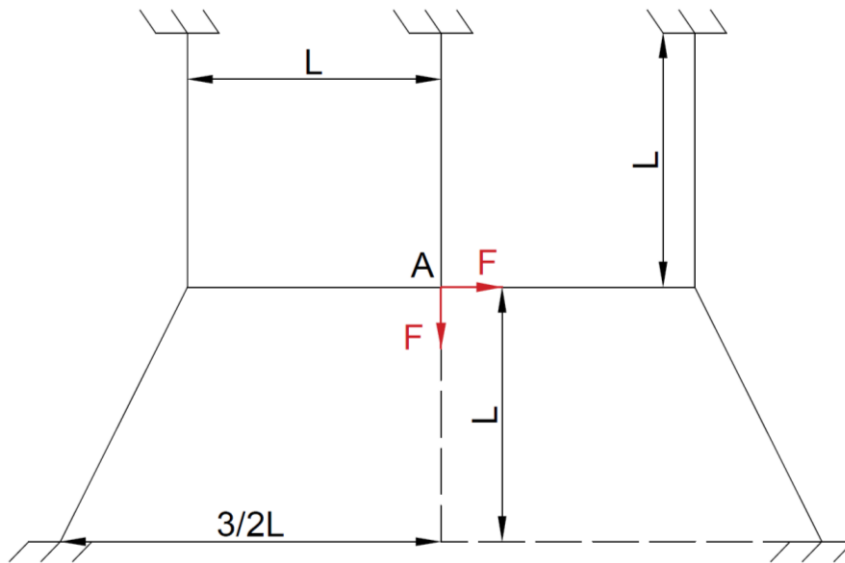


Рис. 6.5.1. Схема нагружения фермы.

#### *Ввод исходных данных*

Зададим параметр  $L = 100$  мм, после чего зайдём в препроцессор и создадим точки:

```
L=100
/prep7
K,,0,L,0
K,,L,L,0
K,,-L,L,0
K,,3*L/2,0,0
K,,-3*L/2,0,0
K,,0,2*L,0
K,,L,2*L,0
K,,-L,2*L,0
```



Соединим данные точки линиями:

```
L, 1, 6  
L, 3, 8  
L, 2, 7  
L, 1, 3  
L, 1, 2  
L, 5, 3  
L, 4, 2
```

Следующим шагом определяется тип КЭ и свойства материалов. Так как нас интересует стержневая система, то тип КЭ будет LINK180. Для реализации разномодульности создадим два материала (модуль Юнга вводится в МПа.).

```
ET, 1, LINK180
```

```
MPTEMP, 1, 0  
MPDATA, EX, 1, , 210000  
MPDATA, PRXY, 1, , 0.3
```

```
MPTEMP, 1, 0  
MPDATA, EX, 2, , 90000  
MPDATA, PRXY, 2, , 0.3
```

Зададим площадь поперечного сечения стержней:

```
R, 1, 20, 0, 0
```

Зададим на линиях необходимое количество КЭ для разбиения:

```
LESIZE, ALL, , , 1, , , , 1  
LMESH, ALL
```

Реализация ГУ: выберем точки с четвёртой по восьмую и зафиксируем все степени свободы: три перемещения и три угла поворота:

```
KSEL, S, , , 4, 8  
DK, ALL, , , 0, ALL, , , , ,  
ALLSEL, ALL
```

Нагрузку прикладываем в первой точке:

```
FK, 1, FY, -1000  
FK, 1, FX, 1000
```

После этого зададим тип решаемой задачи (задача статики) и запустим процесс решения:

```
/SOL  
ANTYPE, 0  
SOLVE
```

Для оценки результата построим эпюру растягивающих/сжимающих напряжений. Создадим две таблицы, содержащие значение нормальных напряжений в I-ом и J-ом узлах КЭ. Согласно документации, для КЭ LINK180 это параметр Sxx и его значения хранятся в LS 1 и 2. После того как эти результаты сохранены в таблицах, выведем их значения на экран (на рис. 6.5.2 приведена иллюстрация ожидаемого результата, величины нормальных напряжений имеют размерность МПа).

```
/POST1  
ETABLE, , LS, 1  
ETABLE, , LS, 2  
PLLS, LS1, LS2, 1, 0
```

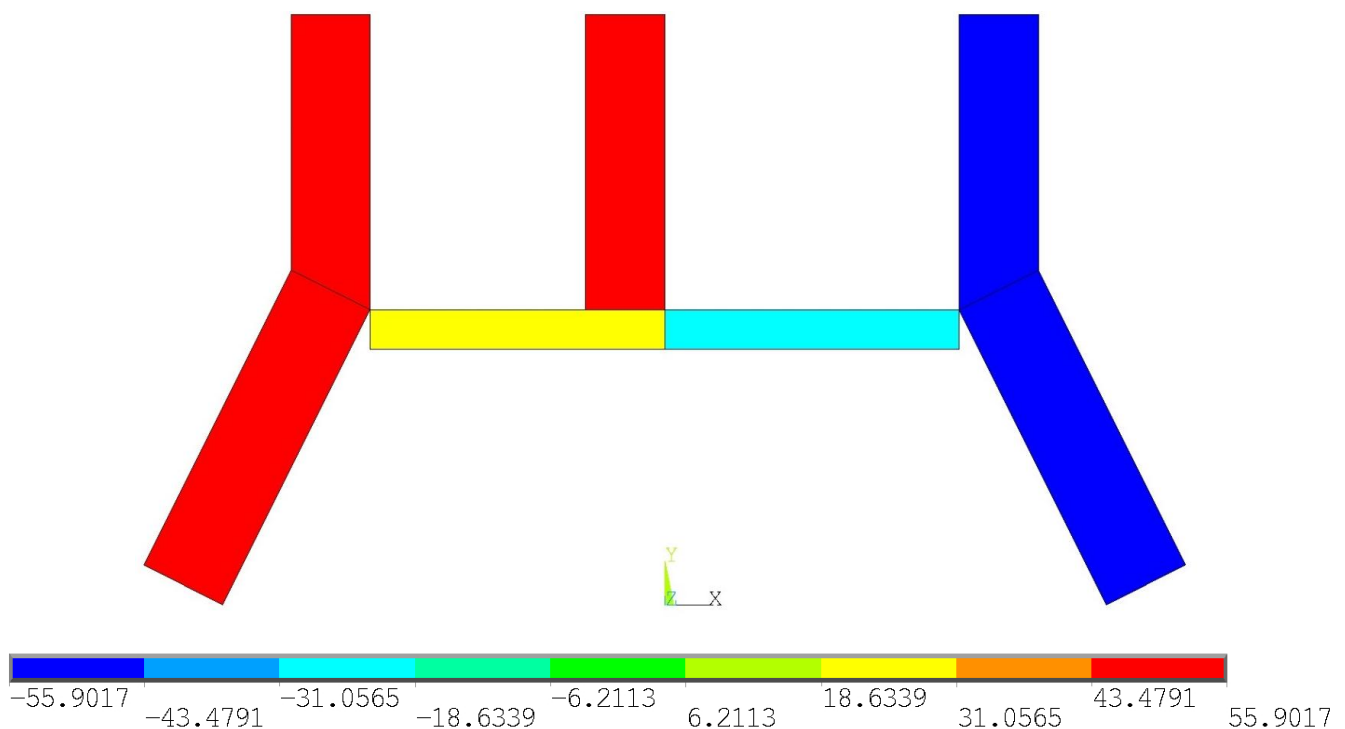


Рис. 6.5.2. Эпюра нормальных напряжений для фермы.

Для замены модуля упругости в стержнях необходимо определить стержни, которые подвергаются сжатию. С этой целью необходимо перебрать все стержни и оценить в них усилия. Для перебора стержней воспользуемся циклом, параметр которого будет изменяться от 1 до nELEM, где nELEM – общее количество КЭ. В теле цикла необходимо определить сжимающиеся стержни и заменить у них материал.

Выберем все КЭ и с помощью команды **\*GET** запишем в nELEM их количество:

```
ESEL,S,, ,ALL  
*GET,nELEM,ELEM,0,COUNT
```

#### *Тело цикла*

Открываем цикл и вводим счётчик ne, который на каждой итерации будет перезаписываться следующим номером КЭ. Перед циклом значение ne надо обнулить:

```
ne=0  
*DO,i,1,nELEM  
  ne=ELNEXT(ne)
```

Запишем с помощью команды **\*GET** в forceAx осевое напряжение в КЭ:

```
*GET,forceAx,ELEM,ne,SMISC,1
```

Затем с помощью условного оператора необходимо оценить знак forceAx: в случае отрицательного значения производится замена материала в КЭ с помощью команды **EMODIF**:

```
*IF,forceAx,LT,0,THEN  
  /PREP7  
  EMODIF,ne,MAT,2,  
  /POST1  
*ENDIF
```

Закрываем тело цикла и перезапускаем процесс решения:

```
*ENDDO  
/SOL  
SOLVE
```

Обновим результаты в таблицах и выведем их значения на экран:

```
/POST1
ETABLE, REFL
PLLS, LS1, LS2, 1, 0
```

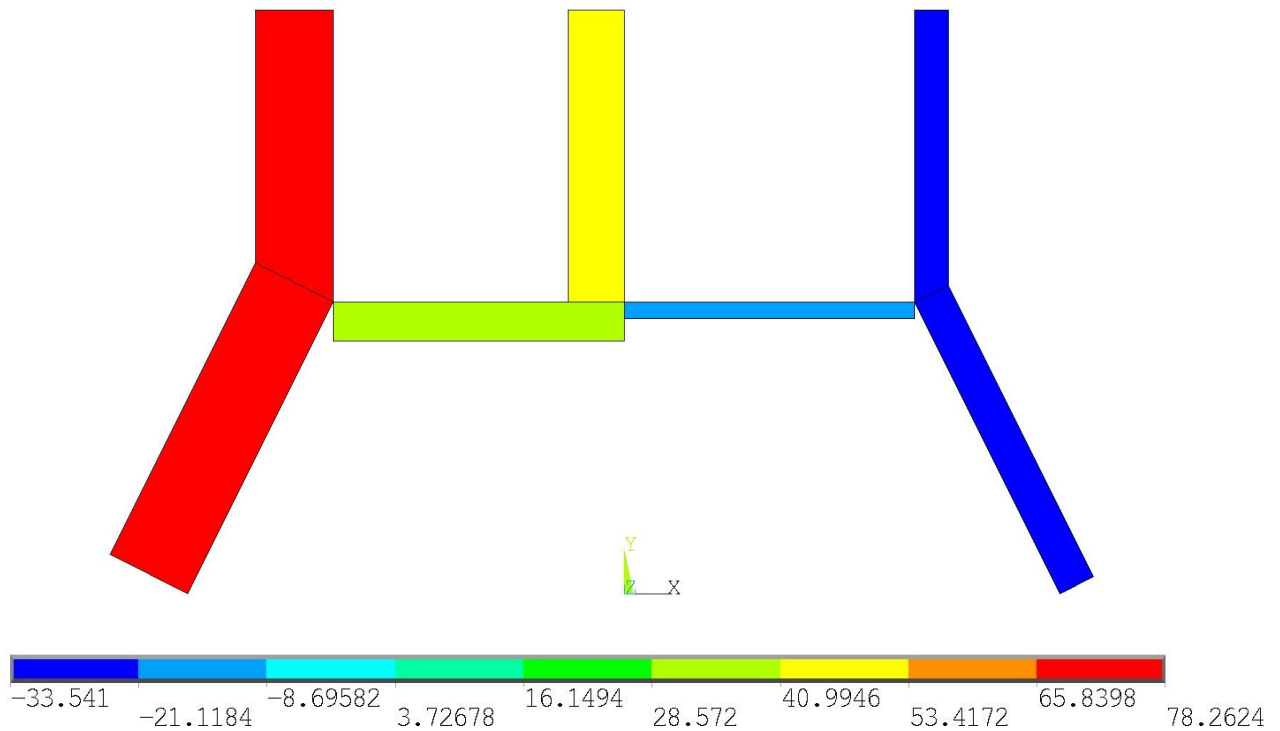


Рис. 6.5.3. Эпюра осевых напряжений для разномодульной фермы.

### *Полный листинг программы*

```
/PREP7
```

```
L=100
```

```
K,,0,L,0
K,,L,L,0
K,,-L,L,0
K,,3*L/2,0,0
K,,-3*L/2,0,0
K,,0,2*L,0
K,,L,2*L,0
K,,-L,2*L,0
```

```
L,1,6
```

```
L,3,8  
L,2,7  
L,1,3  
L,1,2  
L,5,3  
L,4,2
```

```
MPTEMP,1,0  
MPDATA,EX,1,,210000  
MPDATA,PRXY,1,,0.3
```

```
MPTEMP,1,0  
MPDATA,EX,2,,90000  
MPDATA,PRXY,2,,0.3
```

```
ET,1,LINK180
```

```
R,1,20,0,0
```

```
LESIZE,ALL,,1,,1  
LMESH, ALL
```

```
KSEL,S,,4,8  
DK,ALL,,0,ALL,,  
ALLSEL,ALL
```

```
/SOL  
ANTYPE,0
```

```
FK,1,FY,-1000  
FK,1,FX, 1000  
SOLVE
```

```
/POST1  
ETABLE, ,LS, 1  
ETABLE, ,LS, 2  
PLLS,LS1,LS2,1,0
```

```
ESEL,S,,ALL  
*GET,nELEM,ELEM,0,COUNT  
ne=0  
*DO,i,1,nELEM
```

```
ne=ELNEX (ne)
*GET, forceAx, ELEM, ne, SMISC, 1
*IF, forceAx, LT, 0, THEN
  /PREP7
  EMODIF, ne, MAT, 2,
  /POST1
*ENDIF
*ENDDO
```

```
/SOL
SOLVE
```

```
/POST1
ETABLE, , LS, 1
ETABLE, , LS, 2
PLLS, LS1, LS2, 1, 0
```

### Задача 6. Расчёт рамы с трением в скользящей опоре

Рассмотрим раму (см. рис. 6.6.1):  $AC = BC = 80$  мм. Поперечное сечение рамы – прямоугольник со сторонами 5 мм и 8 мм. В точке С действует сила  $F = (150, -1500)$  Н. В точке А жёсткая заделка, а в точке В скользящая опора с коэффициентом трения 0.1. Механические свойства материала: модуль Юнга – 210 ГПа, коэффициент Пуассона – 0.3.

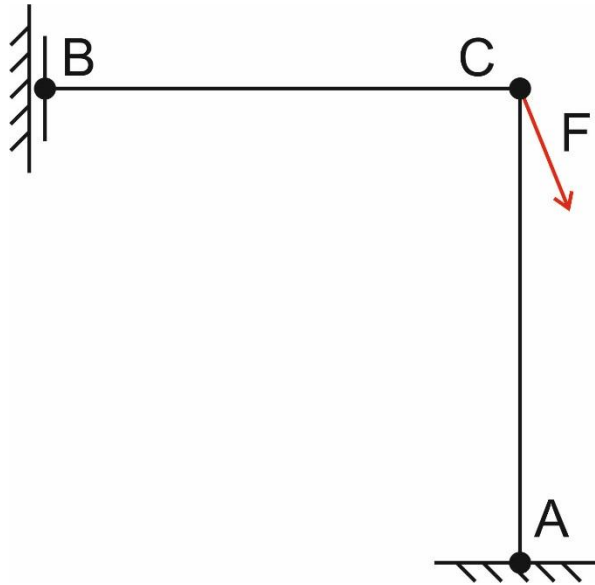


Рис. 6.6.1.Схема нагружения рамы.

Для решения этой задачи реализуем следующий итерационный процесс. На первом шаге решим задачу без учёта трения в опоре – на основании этого определим направление перемещения и нормальную реакцию в точке В. Затем рассчитаем силу трения как величину нормальной реакции, умноженную на коэффициент трения и направленную в противоположное перемещению точки В направление. Прикладывая эту силу как активную, пересчитаем задачу и будем повторять это процесс, пока изменение нормальной реакции в скользящей опоре не станет незначительным.

*Ввод исходных данных*

В первую очередь зададим параметры, определяющие геометрию рамы, величину нагрузки Load, коэффициент трения Fric, количество КЭ на линии Nmesh:

```
L=80  
a=5  
b=8  
Load=1500  
Fric=0.1  
Nmesh=10
```

Перейдём к построению геометрии: для этого зайдём в препроцессор и создадим три точки. Первая соответствует точке А (рис.6.6.1), вторая и третья – точкам С и В соответственно.

```
/PREP7  
K,,L,0,0  
K,,L,L,0  
K,,0,L,0
```

Соединим данные точки двумя линиями:

```
LSTR,1,2  
LSTR,2,3
```

Следующим шагом определяется тип КЭ и свойства материала. Так как линейные размеры были заданы в миллиметрах, то модуль Юнга введём в МПа:

```
ET,1,BEAM188  
MPTEMP,1,0  
MPDATA,EX,1,,210000  
MPDATA,PRXY,1,,0.3
```

Зададим тип и размеры поперечного сечения балки:

```
SECTYPE,1,BEAM,RECT,,0  
SECOFFSET,CENT  
SECDATA,a,b,0,0,0,0,0,0,0,0,0,0
```

Зададим на линиях необходимое количество КЭ, материал и поперечное сечение:

```
TYPE,1  
MAT,1
```



```
LESIZE, All, , , Nmesh, , , , , 0  
SECNUM, 1  
LMESH, All
```

Реализация ГУ: в первой точке необходимо зафиксировать все степени свободы: три перемещения и три угла поворота; в третьей точке необходимо зафиксировать перемещения по осям  $Ox$  и  $Oz$  и три угла поворота:

```
DK, 1, , , , 0, All, , , , ,  
DK, 3, , , , 0, UX, , UZ, ROTX, ROTY, ROTZ,
```

Нагрузка прикладывается во второй точке:

```
FK, 2, FY, -Load  
FK, 2, FX, Load/10
```

Как было отмечено выше, задачу будем решать итерационно. Силу трения будем находить с помощью нормальной реакции и трактовать как активную силу. Блок-схема приведена на рис. 6.6.2: для таких процессов удобно использовать условный цикл.

В теле цикла будет происходить пересчёт и определение силы трения  $FLoad$  до тех пор, пока горизонтальная реакция в скользящей опоре на  $(i+1)$ -ой итерации не совпадёт с заданной погрешностью с горизонтальной реакцией на  $i$ -ой итерации. Для итерационного решения введём дополнительные параметры:

***toll*** – скалярная величина, необходимая для логического выражения условного цикла: хранит в себе отклонение погрешности нормальных реакций двух шагов от заданной точности;

***iter*** – счётчик итераций;

***FLoad*** – переменная, хранящая величину силы трения;

***RForce0*** и ***RForce1*** – горизонтальная реакция в скользящей опоре на предыдущей и актуальной итерациях соответственно;

***eps*** – величина погрешности.

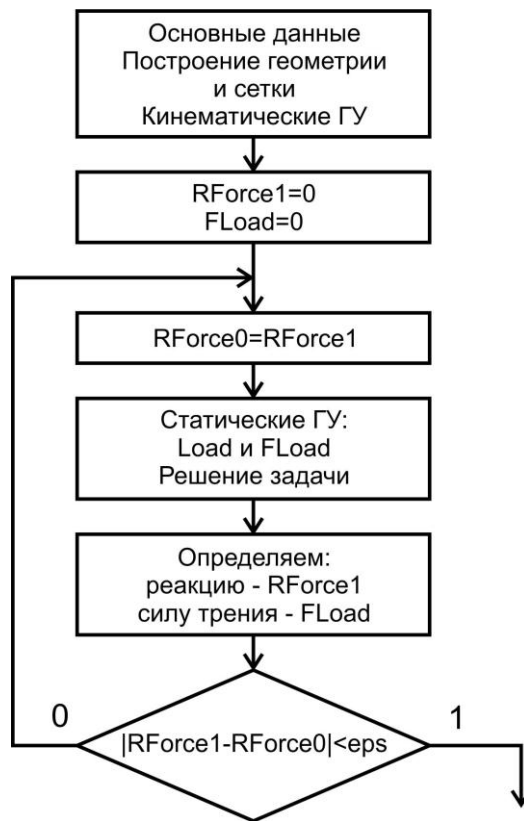


Рис. 6.6.2. Блок-схема итерационного процесса.

Начальные значения переменных:

```

toll=1
iter=0
FLoad=0
RForce0=0
RForce1=0
eps=0.0001

```

Приложим силу трения в третью точку. На первой итерации эта сила будет равняться нулю.

```
FK, 3, FY, FLoad
```

После этого зададим тип решаемой задачи (задача статики) и запустим процесс решения:

```

/SOL
/STATUS, SOLU
SOLVE

```

После определения НДС рамы, необходимо определить горизонтальную реакцию в скользящей опоре. Для этого выберем узел, принадлежащий третьей точке:

```
/POST1  
KSEL,S,,3  
NSEL,ALL  
NSLK,R
```

С помощью команды **\*GET** запишем в RForce1 горизонтальную реакцию, возникающую в скользящей опоре. Для определения направления действия реакции необходимо установить перемещение в точке В – это можно сделать вручную, оценив деформированное состояние рамы, или автоматически командой **\*GET**:

```
*GET, RForce1, NODE, NDNEXT(0), RF, FX  
ALLSEL,ALL
```

Теперь заходим в препроцессор и определяем актуальное значение силы трения (переменная FLoad). Сила трения равна произведению коэффициента трения Fric и горизонтальной реакции RForce1:

```
/prep7  
FLoad=Fric*ABS(RForce1)
```

Далее удаляем все активные силы в третьей точке и переопределяем значение нормальной реакции (переменная RForce0):

```
FKDELE,3,ALL  
RForce0=RForce1
```

Определяем величину отклонения изменения нормальной реакции от заданной точности (переменная toll):

```
toll=ABS(RForce1-RForce0)-eps
```

Теперь можно заключить часть операторов в условный цикл – условием его остановки будет отрицательное значение переменной toll:

```
*DOWHILE, toll  
<тело цикла>  
*ENDDO
```

Для удобства после цикла можно продублировать набор команд, связанных с решением, чтобы после того, как процесс сошелся, можно было получить результат для определенной силы трения:

```
FK, 3, FY, FLoad  
/SOL  
/STATUS, SOLU  
SOLVE
```

На рис. 6.6.3 приведена величина горизонтальной реакции в точке В на различных итерациях. Так как для хранения данных использовался массив типа TABLE, то значения между итерациями аппроксимированы.

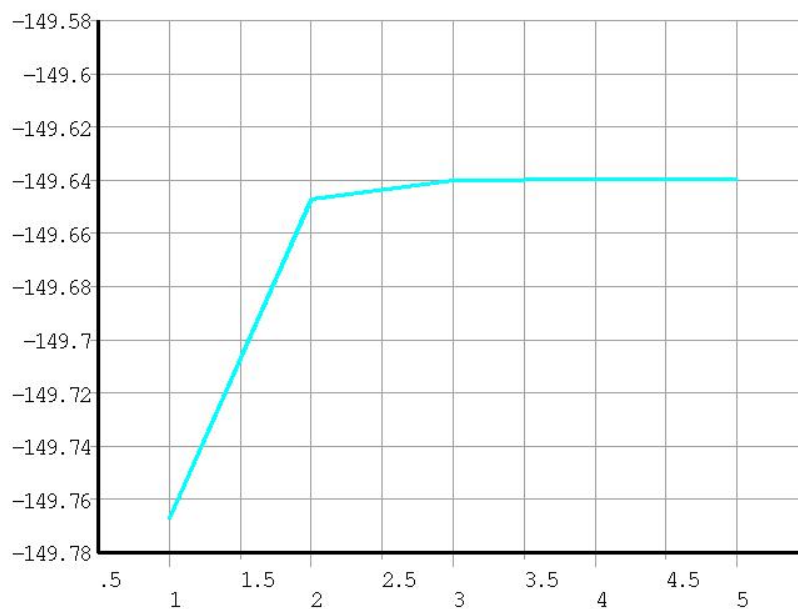
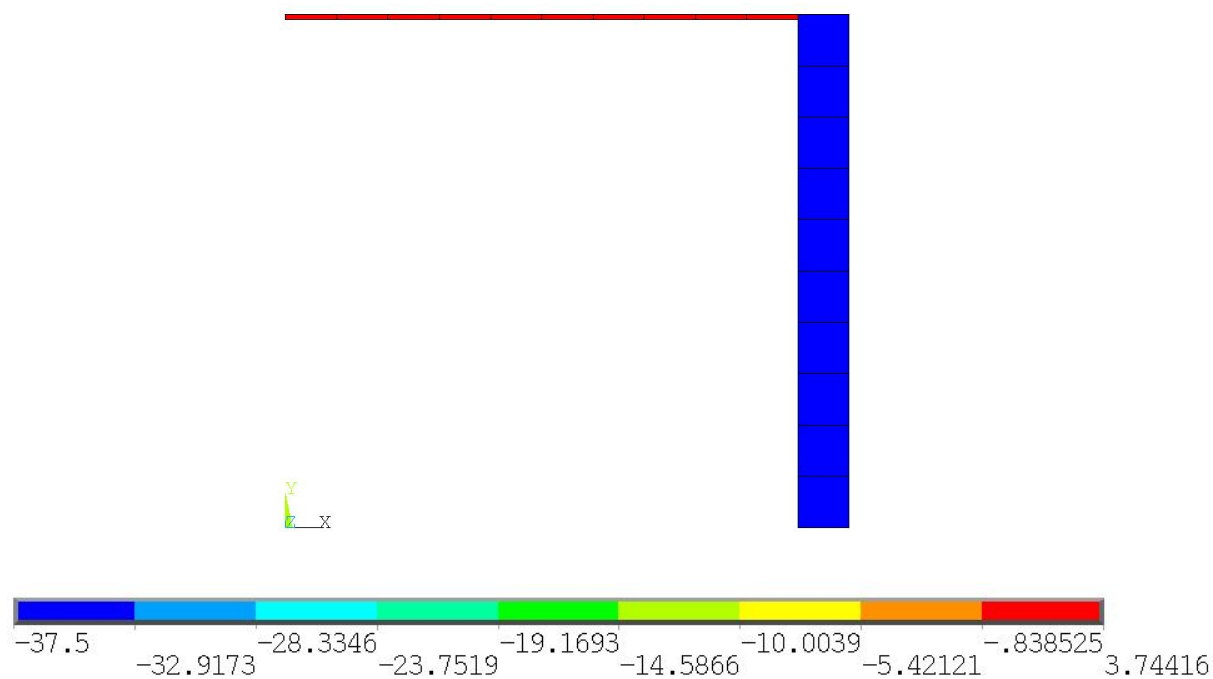


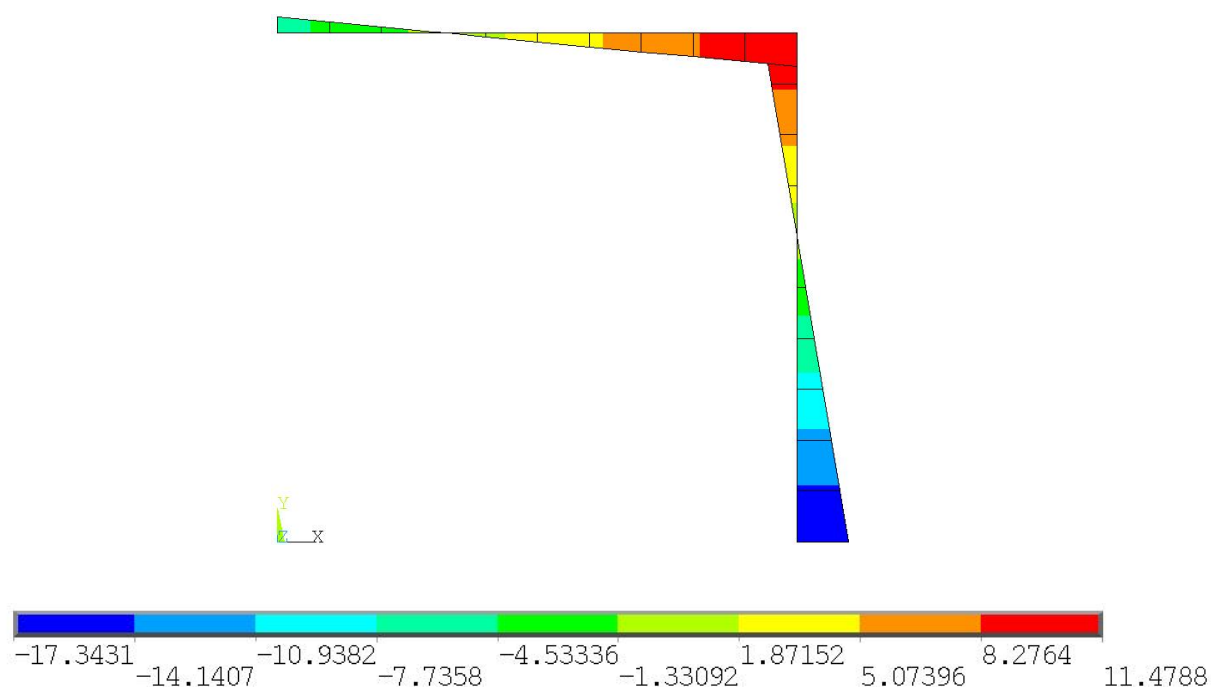
Рис. 6.6.3. Горизонтальной реакции в зависимости от итерации.

```
/POST1  
! Эпюра моментов Mz  
ETABLE, MzI, SMISC, 3  
ETABLE, MzJ, SMISC, 16  
PLLS, MzI, MzJ, 1, 0
```

На рис. 6.6.4 приведены эпюры изгибающих моментов относительно оси  $Oz$ : на рис. 6.6.4а приведена эпюра без учёта сил трения, а на рис. 6.6.4б – с учётом сил трения.



а



б

Рис. 6.6.4. Эпюра изгибающих моментов относительно оси  $Oz$ .

## Полный листинг программы

```
/PREP7
```

```
L=80
```

```
a=5
```

```
b=8
```

```
Load=1500
```

```
FLoad=0
```

```
RForce0=0
```

```
RForce1=0
```

```
Fric=0.1
```

```
eps=0.0001
```

```
Nmesh=10
```

```
K,,L,0,0
```

```
K,,L,L,0
```

```
K,,0,L,0
```

```
LSTR,1,2
```

```
LSTR,2,3
```

```
! КЭ
```

```
ET,1,BEAM188
```

```
! зададим свойства материала
```

```
MPTEMP,1,0
```

```
MPDATA,EX,1,,210000
```

```
MPDATA,PRXY,1,,0.3
```

```
! зададим тип и размеры поперечных сечений
```

```
SECTYPE, 1, BEAM, RECT, , 0
```

```
SECOFFSET, CENT
```

```
SECDATA,a,b,0,0,0,0,0,0,0,0,0,0
```

```
TYPE, 1
```

```
MAT, 1
```

```
LESIZE, All, , , Nmesh, , , , 0
```

```
SECNUM, 1
```

```
LMESH, All
```

```
DK,1, , , , 0, All, , , , ,
```

```
DK,3, , , ,0, UX, , UZ, ROTX, ROTY, ROTZ,
```

```
FK,2,FX, Load/10
```

```
FK,2,FY,-Load
```

```
toll=1
```

```
iter=0
```

```
*DOWHILE, toll
```

```
    iter=iter+1
```

```
    FK,3,FY, FLoad
```

```
    /SOL
```

```
    /STATUS,SOLU
```

```
    SOLVE
```

```
    /POST1
```

```
    KSEL,S, , ,3
```

```
    NSEL,ALL
```

```
    NSLK,R
```

```
    *GET, RForce1, NODE, NDNEXT(0), RF, FX
```

```
    ALLSEL,ALL
```

```
    /prep7
```

```
    FLoad=Fric*ABS(RForce1)
```

```
    !Conver(iter,1)=RForce1
```

```
    FKDELE,3,ALL
```

```
    toll=ABS(RForce1-RForce0)-eps
```

```
    RForce0=RForce1
```

```
*ENDDO
```

```
FK,3,FY, FLoad
```

```
/SOL
```

```
/STATUS,SOLU
```

```
SOLVE
```

```
/POST1
```

```
! Эпюра моментов Mz
```

```
ETABLE,MzI,SMISC, 3
```

```
ETABLE,MzJ,SMISC, 16  
PLLS,MzI,MzJ,1,0
```



## РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Голованов А. И. Метод конечных элементов в механике деформируемых твердых тел / А. И. Голованов, Д. В. Бережной. – Казань: ДАС, 2001. – 301 с.
2. Крылов О. В. Метод конечных элементов и его применение в инженерных расчетах: Учеб. пособие для вузов. – М.: Радио и связь, 2002. – 104 с.: ил.
3. Конюхов А. В. Основы анализа конструкций в Ansys: учебное пособие. – Казань: КГУ, 2001. – 102 с.: ил.
4. Басов К. А. Ansys: справочник пользователя. – М.: ДМК Пресс, 2005. – 640 с.: ил.
5. Бруяка В. А. Инженерный анализ в Ansys Workbench: учебное пособие / В. А. Бруяка, В. Г. Фокин, Е. А. Солдусова, Н. А. Глазунова, И. Е. Адеянов. – Самара: Самар. гос. техн. ун-т, 2010. – 271 с.: ил.

## ПРИЛОЖЕНИЯ

### Приложение 1. Жестко защемленная балка

```
/prep7
CSYS,0
! зададим параметры: L - длина балки;
! a, b - размеры поперечного сечения
! N - количество элементов
N=5
L=100
a=10
b=5
! Зададим кейпоинты и проведем линию
K,,0,0,0
K,,L/2,0,0
K,,L,0,0
LSTR,1,2
LSTR,2,3
!выберем тип элемента и его настройки
ET,1,BEAM188
! Линейная функция формы
KEYOPT,1,3,0
! зададим свойства материала
MPTEMP,1,0
MPDATA,EX,1,,210000
MPDATA,PRXY,1,,0.3
! зададим тип и размеры поперечного сечения
SECTYPE, 1, BEAM, RECT, MyName, 0
SECOFFSET, CENT
SECDATA,10,5,3,3
! привяжем тип элемента, материал, поперечное сечение
! разметим линию для разбиения и разобьем ее
TYPE, 1
MAT, 1
SECNUM, 1
ESYS, 0
LESIZE, All, , , N, , , , , 0
!LMESH, All
LMESH, ALL
! приложим граничные условия
DK,1, , , , 0, ALL, , , , ,
DK,3, , , , 0, ALL, , , , ,
```

```
FK,2,FY,-100
```

```
/SOLU  
/STATUS,SOLU  
SOLVE
```

```
/POST1  
ETABLE, MzI, SMISC, 3  
ETABLE, MzJ, SMISC, 16  
PLLS,MZI,MZJ,1,0
```

## Приложение 2. Шарнирно опертая балка

```
/CWD, 'C:\Users\OASachenkov\Downloads'
```

```
/prep7  
CSYS,0  
! зададим параметры: L - длина балки;  
! a, b - размеры поперечного сечения  
! N - количество элементов  
N=5  
L=100  
a=10  
b=5  
! Зададим кейпоинты и проведем линию  
K,,0,0,0  
K,,L/2,0,0  
K,,L,0,0  
LSTR,1,2  
LSTR,2,3  
!выберем тип элемента и его настройки  
ET,1,BEAM188  
! Линейная функция формы  
KEYOPT,1,3,0  
! Квадратичная функция формы  
! необходимо снять комментарий для работы  
!KEYOPT,1,3,2  
! зададим свойства материала  
MPTEMP,1,0  
MPDATA,EX,1,,210000  
MPDATA,PRXY,1,,0.3  
! зададим тип и размеры поперечного сечения  
SECTYPE, 1, BEAM, RECT, MyName, 0  
SECOFFSET, CENT
```

```

SECDATA,10,5,3,3
! привяжем тип элемента, материал, поперечное сечение
! разметим линию для разбиения и разобьем ее
TYPE, 1
MAT, 1
SECNUM, 1
ESYS, 0
LESIZE, All, , , N, , , , , 0
!LMESH, All
LMESH, ALL
! приложим граничные условия
DK,1, , , , 0, UX, UY, UZ, ROTX, ROTY, ,
DK,3, , , , 0, UX, UY, UZ, ROTX, ROTY, ,
FK,2, FY, -100

/SOLU
/STATUS, SOLU
SOLVE

/POST1
ETABLE, MzI, SMISC, 3
ETABLE, MzJ, SMISC, 16
PLLS, MZI, MJ, 1, 0

```